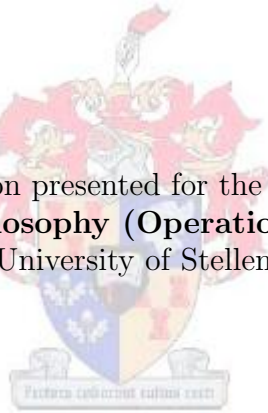
Multi-objective optimisation of water distribution systems design using metaheuristics

by

Darian Nicholas Raad



Dissertation presented for the degree of
Doctor of Philosophy (Operations Research)
at the University of Stellenbosch

Promoter: Prof Jan H van Vuuren
Department of Logistics
Faculty of Economic and Management Sciences
Co–promoter: Dr Alexander Sinske

March 2011

Declaration

By submitting this dissertation electronically, I declare that the entirety of the work contained therein is my own, original work, that I am the owner of the copyright thereof (unless to the extent explicitly otherwise stated) and that I have not previously in its entirety or in part submitted it for obtaining any qualification.

March 1, 2011

Abstract

The design of a *water distribution system* (WDS) involves finding an acceptable trade-off between cost minimisation and the maximisation of numerous system benefits, such as *hydraulic reliability* and surplus capacity. The primary design problem involves cost-effective *specification of a pipe network layout and pipe sizes* (which are typically available in a discrete set of commercial diameters) *in order to satisfy expected consumer water demands within required pressure limits*. The problem may be extended to consider the design of additional WDS components, such as reservoirs, tanks, pumps and valves. Practical designs must also cater for the *uncertainty of demand*, the requirement of *surplus capacity for future growth*, and the hydraulic reliability of the system under different demand and potential failure conditions.

A detailed literature review of exact and approximate approaches towards single-objective (minimum cost) WDS design optimisation is provided. Essential topics which have to be included in any modern WDS design paradigm (such as demand estimation, reliability quantification, tank design and pipe layout) are discussed. A number of formative concepts in multi-objective evolutionary optimisation are also reviewed (including a generic problem formulation, performance evaluation measures, comparative testing strategies, and suitable classes of metaheuristics).

The two central themes of this dissertation are conducting *multi-objective WDS design optimisation using metaheuristics*, and a critical examination of *surrogate measures used to quantify WDS reliability*. The aim in the first theme is to compare numerous modern metaheuristics, including several *multi-objective evolutionary algorithms*, an *estimation of distribution algorithm* and a recent *hyperheuristic* named AMALGAM (an evolutionary framework for the simultaneous incorporation of multiple metaheuristics applied here for the first time to a real-world problem), in order to determine which approach is most capable with respect to WDS design optimisation. Several novel metaheuristics are developed, as well as a number of new variants of existing algorithms, so that a total of twenty-three algorithms were compared.

Testing with respect to eight small-to-large-sized WDS benchmarks from the literature reveals that the four top-performing algorithms are mutually non-dominated with respect to the various performance metrics. These algorithms are NSGA-II, TAMALGAMJ$_{\text{ndu}}$, TAMALGAM$_{\text{ndu}}$ and AMALGAMS$_{\text{ndp}}$ (the last three being novel variants of AMALGAM). However, when these four algorithms are applied to the design of a very large real-world benchmark, the AMALGAM paradigm outperforms NSGA-II convincingly, with AMALGAMS$_{\text{ndp}}$ exhibiting the best performance overall. As part of this study, a novel multi-objective greedy algorithm is developed by combining several heuristic design methods from the literature in order to mimic the design strategy of a human engineer. This algorithm functions as a powerful local search. However, it is shown that such an algorithm cannot compete with modern metaheuristics, which employ advanced strategies in order to uncover better solutions with less computational effort.

The second central theme involves the comparison of several popular WDS reliability surrogate measures (namely the *Resilience Index*, *Network Resilience*, *Flow Entropy*, and a novel

mixed surrogate measure) in terms of their ability to produce designs that are robust against pipe failure and water demand variation. This is the first systematic study on a number of WDS benchmarks in which regression analysis is used to compare reliability surrogate measures with *probabilistic reliability* typically derived via simulation, and failure reliability calculated by considering all single-pipe failure events, with both reliability types quantified by means of *average demand satisfaction*. Although no single measure consistently outperforms the others, it is shown that using the Resilience Index and Network Resilience yields designs that achieve a better positive correlation with both probabilistic and failure reliability, and while the Mixed Surrogate measure shows some promise, using Flow Entropy on its own as a quantifier of reliability should be avoided. Network Resilience is identified as being a superior predictor of failure reliability, and also having the desirable property of supplying designs with fewer and less severe size discontinuities between adjacent pipes. For this reason, it is recommended as the surrogate measure of choice for practical application towards design in the WDS industry.

AMALGAMS$_{\text{ndp}}$ is also applied to the design of a real South African WDS design case study in Gauteng Province, achieving savings of millions of Rands as well as significant reliability improvements on a preliminary engineered design by a consulting engineering firm.

Uittreksel

Die ontwerp van *waterverspreidingsnetwerke* (WVNe) behels die soeke na 'n aanvaarbare afruiling tussen koste-minimering en die maksimering van 'n aantal netwerkvoordele, soos *hidroliese betroubaarheid* en surpluskapasiteit. Die primêre ontwerpsprobleem behels 'n koste-doeltreffende *spesifikasie van 'n netwerkuitleg en pypgroottes* (wat tipies in 'n diskrete aantal kommersiële deursnedes beskikbaar is) *wat aan gebruikersaanvraag binne sekere drukspesifikasies voldoen.* Die probleem kan uitgebrei word om die ontwerp van verdere WVN-komponente, soos opgaardamme, opgaartenks, pompe en kleppe in ag te neem. Praktiese WVN-ontwerpe moet ook voorsiening maak vir *onsekerheid van aanvraag*, genoegsame *surpluskapsiteit vir toekomstige netwerkuitbreidings* en die hidroliese betroubaarheid van die netwerk onder verskillende aanvraag- en potensiële falingsvoorwaardes.

'n Omvattende literatuurstudie word oor eksakte en benaderde oplossingsbenaderings tot enkeldoelwit (minimum koste) WVN-ontwerpsoptimering gedoen. Sentrale temas wat by hedendaagse WVN-ontwerpsparadigmas ingesluit behoort te word (soos aanvraagvooruitskatting, die kwantifisering van betroubaarheid, tenkontwerp en netwerkuitleg), word uitgelig. 'n Aantal basiese konsepte in meerdoelige evolusionêre optimering (soos 'n generiese probleemformulering, werkverrigtingsmaatstawwe, vergelykende toetsingstrategieë, en sinvolle klasse metaheuristieke vir WVN-ontwerp) word ook aangeraak.

Die twee sentrale temas in hierdie proefskrif is *meerdoelige WVN-ontwerpsoptimering deur middel van metaheuristieke*, en 'n kritiese evaluering van verskeie *surrogaatmaatstawwe vir die kwantifisering van netwerkbetroubaarheid.* Die doel in die eerste tema is om 'n aantal moderne metaheuristieke, insluitend verskeie *meerdoelige evolusionêre algoritmes* en die onlangse hiperheuristiek AMALGAM ('n evolusionêre raamwerk vir die gelyktydige insluiting van 'n aantal metaheuristieke wat hier vir die eerste keer op 'n praktiese probleem toegepas word), met mekaar te vergelyk om sodoende 'n ideale benadering tot WVN-ontwerpoptimering te identifiseer. Verskeie nuwe metaheuristieke sowel as 'n aantal nuwe variasies op bestaande algoritmes word ontwikkel, sodat drie en twintig algoritmes in totaal met mekaar vergelyk word.

Toetse aan die hand van agt klein- tot mediumgrootte WVN-toetsprobleme uit die literatuur dui daarop dat die vier top algoritmes mekaar onderling ten opsigte van verskeie werkverrigtingsmaatstawwe domineer. Hierdie algoritmes is NSGA-II, TAMALGAMJ$_{\text{ndu}}$, TAMALGAM$_{\text{ndu}}$ en AMALGAMS$_{\text{ndp}}$, waarvan laasgenoemde drie nuwe variasies op AMALGAM is. Wanneer hierdie vier algoritmes egter vir die ontwerp van 'n groot WVN-toetsprobleem ingespan word, oortref die AMALGAM-paradigma die NSGA-II oortui-gend, en lewer AMALGAMS$_{\text{ndp}}$ die beste resultate. As deel van hierdie studie is 'n nuwe meerdoelige gulsige algoritme ontwerp wat verskeie heuristiese ontwerpsmetodologieë uit die literatuur kombineer om sodoende die ontwerpstrategie van 'n ingenieur na te boots. Hierdie algoritme funksioneer as 'n kragtige lokale soekprosedure, maar daar word aangetoon dat die algoritme nie met moderne metaheuristieke,

wat gevorderde soekstrategieë inspan om beter oplossings met minder berekeningsmoeite daar te stel, kan meeding nie.

Die tweede sentrale tema behels die vergelyking van 'n aantal gewilde surrogaatmaatstawwe vir die kwantifisering van WVN-betroubaarheid (naamlik die *elastisiteitsindeks*, *netwerkelastisiteit*, *vloei-entropie* en 'n *gemengde surrogaatmaatstaf*) in terme van die mate waartoe hul gebruik kan word om WVNe te identifiseer wat robuust is ten opsigte van pypfaling en variasie in aanvraag. Hierdie proefskrif bevat die eerste sistematiese vergelyking deur middel van regressie-analise van 'n aantal surrogaatmaatstawwe vir die kwantifisering van WVN-betroubaarheid en *stogastiese betroubaarheid* (wat tipies via simulasie bepaal word) in terme van 'n aantal toetsprobleme in die literatuur. Alhoewel geen enkele maatstaf as die beste na vore tree nie, word daar getoon dat gebruik van die elastisiteitsindeks en netwerkelastisiteit lei na WNV-ontwerpe met 'n groter positiewe korrelasie ten opsigte van beide stogastiese betroubaarheid en falingsbetroubaarheid. Verder toon die gebruik van die gemengde surrogaatmaatstaf potensiaal, maar die gebruik van vloei-entropie op sy eie as kwantifiseerder van betroubaarheid behoort vermy te word. Netwerkelastisiteit word as 'n hoë-gehalte indikator van falingsbetroubaarheid geïdentifiseer en het ook die eienskap dat dit daartoe instaat is om ontwerpe met 'n kleiner aantal diskontinuïteite sowel as van 'n minder ekstreme graad van diskontinuïteite tussen deursnedes van aangrensende pype daar te stel. Om hierdie rede word netwerkelastisiteit as die surogaatmaatstaf van voorkeur aanbeveel vir toepassings van WVN-ontwerpe in die praktyk.

AMALGAM word ook ten opsigte van 'n werklike Suid-Afrikaanse WVN-ontwerp gevallestudie in Gauteng toegepas. Hierdie toepassing lei na die besparing van miljoene rande asook noemenswaardige verbeterings in terme van netwerkbetroubaarheid in vergeleke met 'n aanvanklike ingenieursontwerp deur 'n konsultasiefirma.

Acknowledgements

I wish to thank my promoter, Prof Jan van Vuuren, for his invaluable support, dedication to excellence, and patience. It has been a fascinating and at times bewildering journey, complicated by a topic change in my first year due to data unavailability, a six month 'research' trip to Canada, and a wildly unexpected, but warmly welcomed, opportunity to upgrade my MSc thesis to a PhD disseration. However, despite the difficulties, the end product has proven very rewarding. My boyish idealism has been suitably crushed, but there is still a sober optimist lurking in the depths. I have matured academically under Prof Van Vuuren's tutelage, and was afforded the opportunity to attend some incredible conferences, including the 18^{th} Triennial International IFORS conference held in Sandton in 2008.

I would also like to thank my co-promoter, Dr Alexander Sinske, for providing me with the opportunity to investigate a fascinating subject under the guidance of a professional in the field of hydraulic engineering. His practical advice provided an additional layer of substance to my work. I would further like to thank him for the opportunity to attend the International Water Distribution Systems Analysis conference held in the Kruger National Park in 2008, an enriching experience where I was able to speak directly to many of the top researchers in the field of WDS design optimisation.

This study would not have been possible without the funding provided by the Harry Crossly Foundation, as well as financial support from Stellenbosch University.

I wish to thank the staff at the Applied Mathematics Division of the Department of Mathematical Sciences and at the Department of Logistics for their kind advice, and facilitation of my work. Also, thanks to those people who were involved in proof-reading my work.

I wish to thank my parents and brother for their continual moral and financial support throughout my studies. I am also grateful for the presence of some amazing work colleagues and friends who, during the course of my research, provided a highly stimulating and supportive environment in which to work and live. In order of how much I love them, I would like to make special mention of Shawn Bergh, Jacques du Toit, Carina Joubert, Dirk Keen, Martin Kidd, Thomas Lawrie, Dillon Marsh, Jessica Moll, Ingrid Mostert, Dennis Moss, Kieka Mynhardt, Frank Ortmann, Eddie Raad, Rhonda Raad, Tayne Ruddock, Dora Scott, Alan Smith, Ian Stuart, Bani van der Merwe, De Villiers Venter, and Lieschen Venter, for making my life much, much better. Finally, I want to thank the universe for its kindness in providing the formative elements and circumstances which brought about my existence. Truly, this is a bizarre and exceptional state to find oneself in. And truly, it would be a waste not to have some fun with it.

Terms of Reference

The author was first introduced to the topic of water distribution systems design towards the end of the first year of his MSc studies in 2006. An old university colleague named Ben Harper mentioned in passing that one of the directors of the company he worked for needed assistance in developing optimization software for their WDS design package Wadiso^{TM}. This led the author to contacting Dr Alexander Sinske at the company GLS Consulting (Pty) Ltd, based in Technopark near Stellenbosch. Prior to that the author had been dabbling in the modelling of the natural water systems of the Western Cape, focussing on the Theewaterskloof dam catchment area, a problem for which he was struggling to obtain the necessary data. After an interview with Dr Sinske, the author decided to investigate metaheuristics towards the design of water distribution systems instead, since the scope of the project was far better defined, and the outcomes could hopefully be implemented in practise. Dr Sinske was eventually appointed co-supervisor for the study, since his extensive industry experience was invaluable in shaping the direction of the research, ultimately yielding a more pragmatic model. In particular, Dr Sinske was able to provide industry design guidelines, and access to real-world design projects, such as the R21 Corridor WDS development project that was used as a case study in this dissertation. The course of 2007 was primarily spent coming to grips with the concepts and mathematics of hydraulic modelling and evolutionary metaheuristics, and in developing the initial software framework for hydraulic modelling and optimization.

In 2008 the author was fortunate enough to participate in the Canadian Graduate Students Exchange Programme, which saw him flying off for six months to the University of Victoria on beautiful Vancouver Island. His Canadian supervisor was Prof Kieka Mynhardt, who helped him to expand his graph theory repertoire, which is extremely useful in understanding network reliability. The author continued work on his MSc thesis, and produced some excellent results, which were used to coauthor papers for two international conferences. On returning to South Africa in July 2008, the author attended the triennial conference of the International Federation of Operations Research Societies (IFORS) held in Sandton, where he submitted his work in the form of a paper towards the OR in Development Competition. Much to his delight, this garnered second place in the competition. Consequently, this paper was published as

- Raad DN, Sinske A & Van Vuuren JH, 2009. *Robust multi-objective optimization for water distribution system design using a meta-meta-heuristic.* International Transactions in Operational Research, **16**(5), 595–626.

A second conference paper was submitted for the 10th annual International Water Distributions Systems Analysis conference held on August 2008 in the Kruger National Park,

and published in the conference proceedings as

- Raad DN, Sinske AN & Van Vuuren JH, 2008. *Jumping genes for water distribution system design optimization*, Proceedings of 10th Annual Water Distribution Systems Analysis Conference (WDSA 2008), held in the Kruger National Park, South Africa and organised by the American Society of Civil Engineers, 437–449.

At this conference the author was fortunate enough to meet many of the leading researchers in the field of WDS analysis, which provided him with many ideas on strengthening his work. The author submitted his MSc Thesis in November of 2008.

At the authors MSc defence in February of 2009, he was given the opportunity to upgrade his masters to a PhD, since the work was judged of such a quality to be worthy of the honour. The author gladly accepted and set his life down a new path. He was able to redesign the entire software framework to make it substantially more generic, and address many of the shortcomings of the MSc research, such as the lack of consideration of temporal demand correlation and pressure-driven analysis. The author also greatly strengthened the experimental method and developed many new algorithms for WDS design optimization. Over the course of 2009 and 2010, two additional papers were coauthored, namely

- Raad DN, Sinske A & Van Vuuren JH, 2010. *Multiobjective optimisation for water distribution system design using a hyperheuristic*, Journal of Water Resources Planning and Management, **136**(5), 592–596, and

- Raad DN, Sinske A & Van Vuuren JH, 2010. *Comparison of four reliability surrogate measures for water distribution systems design*, Water Resources Research, **46**, Paper W05524 (no page numbers).

In September of 2010, the author attended the Operations Research Society of South Africa (ORSSA) conference held near Tzaneen in Limpopo province. Here he was presented with the prestigious Tom Rozwadowski medal, awarded annually for the best OR paper in an international journal for the ITOR article. This was an unexpected highlight of his academic career. The author submitted his PhD dissertation in November 2010.

Table of Contents

List of Figures

List of Tables

List of Algorithms

List of Acronyms

Acronym	Definition
ACO	Ant Colony Optimisation
ADS	Average Demand Satisfaction
ADSU	Average Demand Satisfaction under Uncertain demands
ADSF	Average Demand Satisfaction under Failure conditions
ADMOEA	Another Dynamic Multi-objective Evolutionary Algorithm
ANOVA	Analysis of Variance
APR	Average Probabilistic Reliability
AFR	Average Failure Reliability
EAS	Epsilon Archive Size
CS	Control Surface
CSIR	Council for Scientific and Industrial Research (South African)
CV	Control Volume
DE	Differential Evolution
DDA	Demand Driven Analysis
DR	Dominance Rank
EDA	Estimation of Distribution Algorithm
EGL	Energy Grade Line
EI	Explicit Integration
FDV	Fraction of Delivered Volume
FDD	Fraction of Delivered Demand
FDQ	Fraction of Delivered Quality
FGN	Fixed Grade Node
FR	Failure Reliability
GA	Genetic Algorithm
GIS	Geographic Information Systems
HGL	Hydraulic Grade Line
ISO	Insurance Services Office (USA)
IWA	International Water Association
LHS	Latin Hypercube Sampling
MCS	Monte Carlo Simulation
MOA	Multi-objective Algorithm
MOEA	Multi-objective Evolutionary Algorithm
MOO	Multi-objective Optimisation
MOPSO	Multi-objective Particle Swarm Optimisation
NHV	Normalised Hypervolume
NR	Network Resilience
NRV	Non-return Valve
NSGA-II	Non-dominated Sorting Genetic Algorithm II
NYTUN	New York Tunnels
PDA	Pressure Driven Analysis
PEM	Partial Enumeration Method
PR	Probabilistic Reliability

Acronym	Definition
PRV	Pressure Reducing Valve
PSO	Particle Swarm Optimisation
PUMDA	Partitioned UMDA
RI	Resilience Index
RNSGA-II	Robust NSGA-II
SBX	Simulated Binary Crossover
SD	Standard deviation
SDD	Sum of Diameter Differences
SQDD	Sum of Squared Diameter Differences
SPEA-II	Strength Pareto Evolutionary Algorithm II
SSDM	Source Share Deviation from Mean
RSM	Reliability Surrogate Measures
TCV	Throttle Control Valve
TLN	Two Loop Network
TRP	Two Reservoir Problem
UN	United Nations
UMD/A	Univariate Marginal Distribution / Algorithm
WDS	Water Distribution System
WDSDO	Water Distribution System Design Optimisation

List of Symbols

The following font conventions are applicable in this dissertation. Unless defined otherwise in the table of symbols, vectors are represented by lower case symbols in bold font, matrices are represented with upper case Roman alphabetic letters in bold font, sets are represented using uppercase calligraphic symbols, and scalar variables are denoted by lower case symbols. Variables are typically represented by cursive math font, whilst standard font may be used as a subscript or superscript to differentiate variables.

Symbol	Description	Units
a	Wave speed	m/s
$\mathbf{a}$	Acceleration	m/s^2
A	Area	m^2
$\mathbf{A}$	Area vector	m^2
α	Pheromone learning exponent (ACO)	—
b	Intensive property	—
B	Extensive property	—
β	Local cost exponent (ACO)	—
c	Emitter / leakage coefficient	—
C	Cost	R (Rands)
C_{hw}	Hazen-Williams pipe coefficient	—
$\boldsymbol{d}$	Vector of nodal demands ($n \times 1$)	m^3/s
$\mathcal{D}$	Set of demand loading condition vectors	m^3/s
$\boldsymbol{d}_i$	The i-th member of $\mathcal{D}$	m^3/s
d_i	Demand at the i-th node	m^3/s
d_{R}	Dominance rank quantifier ([148])	—
d^{c}	Dominance count	—
D	Diameter of pipe	m
e	Energy per unit mass	kJ
E	Energy	kJ
e_{k}	Kinetic energy per unit mass	kJ
e_{p}	Potential energy per unit mass	kJ
e_{u}	Internal energy per unit mass	kJ
E_{k}	Kinetic energy	kJ
E_{m}	Elastic modulus of pipe wall	N/m^2
E_{p}	Potential energy	kJ
E_{u}	Internal energy	kJ
E_{v}	Bulk modulus of elasticity	N/m^2
$\mathcal{E}$	Entropy	—
$\mathcal{E}_{\text{f}}$	Flow entropy	—
f	Darcy-Weisbach friction factor	—
$\boldsymbol{F}, F$	Force	N (= kg.m/s^2)
$\hat{F}$	Failure event function for a component set	—
$\mathcal{F}$	Set of failure events	—
$\mathcal{F}^i_{\boldsymbol{x},k}$	Event of failure of the i-th instance of k specific components	—

Symbol	Description	Units
g	Gravitational acceleration constant (= 9.81)	J/kg.m or m/s^2
$G_{\max}$	Maximum number of search generations	—
γ	Specific weight	N/m^3
h	Pressure head	m
$\boldsymbol{h}$	Vector of nodal heads ($n \times 1$)	m
$h_{\min}, h_{\max}$	Minimum and maximum nodal heads	m
h_{f}	Head loss due to pipe friction	m
h_{p}	Head supplied by a pump	m
h_{t}	Head lost to a turbine	m
h_{L}	Head loss	m
$\boldsymbol{h}_{\min}, \boldsymbol{h}_{\max}$	$n \times 1$ vectors of minimum and maximum nodal heads	m
H	Heat	J
h_{R}	Reservoir levels	m
h_{R}^0	Initial reservoir levels	m
i, j	Ordinarily denotes pipe connecting nodes i and j	—
k_{s}	Average height of pipe roughness elements	m
K	Loss coefficient	—
K_{en}	Minor loss coefficient for sudden expansion at entrance	—
K_{ex}	Minor loss coefficient for gradual expansion	—
K_{co}	Minor loss coefficient for gradual contraction	—
K_{e}	Equivalent loss coefficient	—
κ	Dimensionality of $\boldsymbol{x}$	—
L	Length of pipe	m
ł	Number of modifiable components in a WDS	—
m	Mass	kg
M	Number of optimisation objectives	—
μ	Dynamic viscosity	cP (= N.s/m^2)
n	Number of nodes in network	—
n_{f}	Number of fixed grade nodes in network	—
n_{L}	Number of primary loops in network	—
n_{p}	Number of pipes in the WDS	—
n_{pmp}	Number of pumps in the WDS	—
n_{R}	Number of reservoirs in the WDS	—
n_{T}	Number of tanks in the WDS	—
N	Population size (population-based algorithms)	—
N_{C}	Number of constraints	—
η	Head loss flow exponent	—
ν	Kinematic viscosity	m$^2/s$
p_{f}	Penalty factor	—
p_{c}	Probability of crossover (GA)	—
p_{m}	Probability of mutation (GA)	—
$\widehat{P}(\cdot)$	Penalty function	—
p	Pressure	Pa (= N/m^2)
$P(\cdot), p(\cdot)$	Probability density function	—
P_{r}	Power	W (= J/s)
P_{ru}	Pipe unitary power	W (= J/s)
$\mathbf{p}$	Momentum	kg.m/s
$\Phi(\cdot)$	Standard normal (Gaussian) cumulative distribution function	—
q	Discharge (volume flow rate)	m^3/s
$q_{i,j}^{\mathrm{p}}$	Flow in pipe connecting nodes i and j	m^3/s
Ψ	Pheromone reward factor (ACO)	—
$\boldsymbol{q}$	Vector of nodal outflows	m^3/s
$\boldsymbol{q}_{\mathrm{p}}$	Vector of pipe flows ($n_{\mathrm{p}} \times 1$)	m^3/s
r	Radius	m

Symbol	Description	Units
R	Reliability	% or other
R_e	Reynolds number	—
$\mathcal{R}$	Set of reservoirs	—
ρ	Density	$\mathrm{kg/m^3}$
ρ_e	Pheromone persistence factor (ACO)	—
s	Distance along the length of a pipe	m
$\mathbf{s}$	Cartesian coordinate vector	m
S	Specific gravity	—
$\acute{S}$	WDS design feasibility function	—
σ_h	Variance of pressure head at node	m
t	Time / discrete time (iteration counter)	s / —
T	Temperature	$^o\mathrm{C}$
T_D	Sum total of demands	$\mathrm{m^3/s}$
τ	Sheer stress	$\mathrm{N/m^2}$
$\widehat{\tau}$	Pheromone concentration (ACO)	
u	Uniform random variable (typically $\in [0, 1]$)	—
$\mathbf{v}$	Velocity	m/s
v	Velocity (magnitude)	m/s
$\overline{v}$	Velocity (mean magnitude)	m/s
V	Volume	$\mathrm{m^3}$
$\mathbf{v}_e$	Unit velocity	m/s
υ	Number of uncertain WDS parameters	—
φ	Number of optimisation parameters	—
ϑ	Safety factor	—
w	Weighting coefficient / inertial weight	—
W	Work	kJ
W_s	Shaft work	kJ
W_f	Flow work	kJ
W_t	Turbine work	kJ
W_p	Pump work	kJ
ω	Number of values in a discrete set (*e.g.* a range of pipe diameters)	
y, z	Height / elevation	m
$\boldsymbol{x}$	Decision variable vector	—
$\boldsymbol{x}^\mathrm{d}$	Discrete decision variable vector	—
$\boldsymbol{x}^\mathrm{c}$	Continuous decision variable vector	—
$\mathcal{X}$	Discrete set of decision variable options	—
$\boldsymbol{\chi}$	Set of uncertain WDS parameters	—
ξ_h	Mean pressure head at node	m
$\boldsymbol{z}$	Instance of nodal demands $\boldsymbol{d}_j$ and WDS failure condition $\mathcal{F}_k^i$	$\langle \mathrm{m^3/s}, - \rangle$
ζ	Heuristic function favoring low cost options (ACO)	—

Chapter 1

Introduction

> "Water is life, sanitation is dignity." (South African Strategic Framework for Water Services [207])

Water — that most precious substance, essential for the survival of all life on earth. One cannot but pause a moment and ponder in reverence at its simple purity and extreme necessity. Safe, effective water storage and delivery systems are amongst mankind's greatest feats of engineering, and they present some of the most compelling challenges in this dire age of overpopulation and global warming.

The subject matter of this dissertation is the design optimisation of urban water distribution systems (WDSs). Indeed, there is immense potential for reducing costs and building better, more reliable water systems, considering a broad range of objectives. The primary formative elements of these water networks are pipes, reservoirs, tanks, pumps and valves. A simple water distribution network is shown in Figure 1.1, including a reservoir with a pump, a balancing tank, five numbered junctions connected by pipes, and a valve. The looped layout of the pipes is a common feature of WDS, as loops provide alternative flow pathways enabling the disconnection of pipes during times of system maintenance or failure. The primary goal of WDS design optimisation is to minimize installation (or rehabilitation) and operating costs, whilst satisfying flow and pressure requirements throughout the system. However, in recent years there has been an increasing focus on obtaining information on the trade-off between system cost and benefits (often expressed in terms of system *reliability*). This has led to the use of multi-objective optimisation techniques that obtain a Pareto-optimal set of solutions in cost–benefit space. In this dissertation, various algorithms for the multi-objective design optimisation of WDSs are analyzed, considering both cost and surrogate measures of reliability. These algorithms are applied to the design of real WDSs, in order to prove them practical for real-world engineering.

1.1 Water Distribution System Design Optimisation

A water network is typically represented as a two-dimensional plan of existing and/or potential pipelines. Individual pipes are linked together to form pipelines, which may meet at *nodes* (or *junctions*). Although water may exit the pipeline at any point along its length via service lines, demand is usually grouped at the nodes (also known as *demand nodes*). Pipelines may also be connected to tanks, pumps, and valves. Pipes are ordinarily straight and cylindrical, both because this is the easiest and most reliable way to manufacture them, and because a cylindrical section is best suited to handle fluid pressure and makes the most economical use

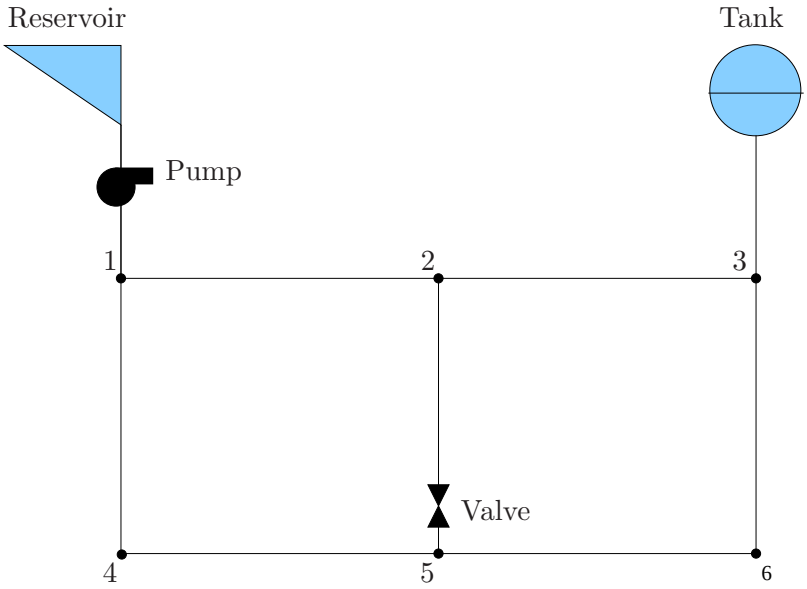


Figure 1.1: *A simple water distribution network.*

of pipe material. Each component in a WDS is associated with an elevation. Demand nodes are also associated with lumped demand quantities (or *loads*), expressed in terms of flow out of the network. Reservoirs and tanks (*sources*) are associated with a volume of water in storage, a potential energy expressed in units of pressure that depends on the source elevation, and a maximum flow rate at which they can feed the network. Historically, the physical network layout was designed first by an engineer, so that the design optimisation problem was merely that of choosing the component characteristics for a static layout of pipelines and other components at fixed locations. Recently, there have been several attempts at incorporating some form of layout design and/or component placement in the optimisation, which complicates the problem considerably [4].

Optimisation proceeds by considering alternative sizes for, and operations on, pipelines and other system elements and, for each network configuration, calculating the hydraulic properties of the network such as flow and pressure values. Calculating hydraulic properties is commonly known as 'balancing the network', and is itself a challenging problem. The system has a feasible configuration if the hydraulic properties satisfy the constraints set on them. However, it is also possible to consider infeasible WDS designs depending on the extent of the constraint violation. In an exhaustive search of configurations, each system element would take on each of its possible attribute values, generating multiple combinations. Combinations grow exponentially as the number of network elements increase (the search complexity is $O(\omega^{\kappa})$, where κ is the number of formative elements and ω is the number of design options for each element).

The size of a new pipeline may typically be one of a discrete set of commercially available pipe sizes, each associated with a different cost. In addition, existing pipes in a system under rehabilitation may undergo several operations including, cleaning (which reduces internal pipe roughness), parallel pipe installation (duplication), replacement, or removal. It is important to note that each pipe size has its own pressure rating, which may complicate the constraints of the problem, though typically they are safely in excess of the system pressure performance constraints. Tanks are used as balancing agents to provide additional flow during times of high demand, and fill during periods of low demand. Tanks also assist in providing consistent pressures across the WDS (also known as *pressure equalization*). They may be placed at multiple

points in a system, and come in a variety of sizes and installation costs. It is desirable that tanks fill and empty over their operational ranges during their demand cycle (*e.g.* daily / weekly), in order to avoid overflow and water stagnation. Pumps may also be placed throughout a system to add energy where necessary, though this is typically near the water source in the form of a pump station. There are various pump types, each with different installation and running costs. Pumps are also associated with operating curves with different wire-to-water efficiencies at different pressure and flow conditions. Valves are used to change flow profiles between pipes at links in the network, often for reducing pressure between different parts of a system. Optimisation typically applies to a steady-state system (flow velocity in each part of the system is static) during peak flow conditions, although several different demand loads may be analyzed (the loads may be incorporated as constraints in the optimisation process). It is assumed that a fixed inflow and outflow to the system is known in advance. If a system can satisfy peak demand, then it will obviously also be able to cater for reduced demand, but care must be taken to respect maximum pressure limitations, justifying the need for a static zero-flow simulation. If tanks are to be designed, then it is essential to conduct an extended period analysis, simulating the tank inflows and outflows, in order to design for effective tank operation [248].

Given a computer representation of a hydraulic network, several public-domain source code libraries exist which are able to calculate its hydraulic state properties. The highly popular EPANET 2 [203] dynamic-linked-library will be used for this purpose in this dissertation. The calculated state variables must satisfy the pressure and flow requirements for each demand node, otherwise the network will not fulfil its supply objectives.

1.2 Motivation for Research Topic

Water distribution systems are essential to modern civilization, and their inadequacy places absolute limitations on economic growth, social development and health. The World Health Organization / UNICEF 2010 report *Progress on Sanitation and Drinking Water* indicates that while 87% of the global population obtains their water from improved sources, there are many regions with extremely poor access. In particular, only 60% of the populations of Sub-Saharan Africa and 50% of those in Oceania have access to treated water supply. The situation is most dire in rural areas, where in Sub-Saharan Africa coverage is only 47% in rural areas compared to 83% in urban areas, and in Oceania where coverage is 37% in rural areas compared to 92% in urban areas [258]. An urgent need exists to develop this critical infrastructure in developing countries, as highlighted in the UN Millennium Development goal of halving the proportion of the population without sustainable access to safe drinking water and basic sanitation by 2015 [233].

In many developing countries, 30 to 40% of water (or more) is lost due to water leakages and illegal tapping [235], a situation which is exacerbated as systems age, unless they are properly maintained. This highlights the design goal of planning for leakage abatement, long-term performance considerations and correctly sizing pipes according to their pressure and velocity ratings.

Furthermore, the rapid pace of urban development and the steady rise of global warming all place pressure on our basic resources, especially water, creating enormous planning and management challenges for now and the future [132].

Another important concept often neglected is designing with a long-term outlook, considering

the entire life-cycle of the engineered system. In particular, there is typically a trade-off between initial investment costs and maintenance / operation costs over the system lifespan. The present value of these costs should be included in the model. Secondly, one of the major problems in developing communities is that although there may be availability of funds to install new infrastructure, there is a severe shortage of resources and skill to maintain that infrastructure. This calls for the installation of more robust systems which require less maintenance over their lifespan [234].

One of the underlying goals in this dissertation is to investigate techniques that enable automatic design optimisation with minimal user input. This is important in real design situations, as many optimisation models contain a plethora of parameters which are unfathomable to the average engineer. This is probably the main reason why design optimisation for engineering has not become more mainstream.

WDSs are extremely costly to install and maintain [248], and it is often the case that optimisation can achieve dramatic cost savings, as shall be demonstrated in the South African case study towards the end of this dissertation. Any methodology which makes WDS design easier and more comprehensive is worth earnest consideration, especially if it can produce designs which are both cheaper and more reliable.

Finally, at the time of writing there was no commercial software product for WDS design which offers the possibility of multi-objective optimisation [208]. It is the goal of the author to rectify this shortcoming by producing a dynamic linked-library which may easily be incorporated into any commercial package, using a familiar input format already widely used in the industry (*i.e.* that of EPANET §2.2.9).

1.3 Research Scope and Objectives

The objectives of this dissertation are:

1. To *provide* a review of the hydraulics theory necessary for WDSs analysis, furnishing the terminology necessary to formulate a WDS design optimisation (WDSDO) model.

2. To *provide* a broad introduction to the problem of WDS design, including the practical engineering perspective and various mathematical formulations of the problem.

3. To *conduct* an extensive literature survey on the topic of design optimisation for WDS, focussing on the problem of component sizing and placement. This shall be addressed in two phases:

   (a) The problem of least-cost design of a fixed layout network under a single water demand scenario with given hydraulic and pressure constraints.

   (b) The multi-objective problem considering objectives of minimizing total costs and maximizing system reliability, providing some scope for layout modification.

4. To *provide* a self-contained introduction to the topic of multi-objective optimisation using *multi-objective evolutionary algorithms* (MOEAs) and other population-based metaheuristics.

5. To *formulate* a pragmatic model for the multi-objective WDSDO problem, and to produce software for the model implementation.

6. To *compare* several existing and new population-based metaheuristics for multi-objective optimisation of WDS designs in a systematic manner on multiple benchmark systems. The algorithms compared should include the *Non-dominated Sorting Genetic Algorithm 2* [61], the *Strength Pareto Evolutionary Algorithm 2* [277], a *Differential Evolution* algorithm [152], a *Particle Swarm Optimisation* algorithm [231], a novel *Greedy Engineering Heuristic*, an *estimation of distribution algorithm* (EDA) based on the *Univariate Marginal Distribution* (UMD) [185] and a novel variant of UMD named *Partitioned UMD*, an adapted *Cellular Dynamic Multi-objective Evolutionary Algorithm* [271], a novel self-adaptive evolutionary algorithm named *ANIMA*, and a recent *hyperheuristic* named *AMALGAM* [244].

7. To study several alternative formulations of the AMALGAM hyperheuristic, addressing shortcomings uncovered in the existing algorithm.

8. To *compare* two different constraint handling techniques, the first incorporating a penalty term, and the second using a recent method called *constrained domination* [190].

9. To *study* different numeric indicators of WDS reliability (reliability surrogate measures) for use during multi-objective optimisation. This will include the *Reliability Index* measure [227], the *Network Resilience* measure [194], and the *Flow Entropy* measure [195]. These measures should be compared in terms of their ability to produce solutions that are robust in terms of uncertain demands and pipe failure conditions.

10. To *implement* the optimisation model in a widely used programming language, yielding a software library which may easily be linked to any commercial WDS design software package.

1.4 Dissertation Layout

Chapter 2 constitutes a literature survey of fluid mechanics for WDSs providing the foundation required to understand pipe hydraulics and hydraulic network simulation theory. Chapter 3 deals with the problem of least-cost WDS design optimisation, including a thorough investigation of existing single-objective optimisation algorithms. In Chapter 4, essential topics in WDS design are presented, including demand estimation, tank design and reliability quantification. Chapter 5 contains an overview of multi-objective optimisation and algorithms used in this context. It also contains a multi-objective formulation of the WDS design problem. In Chapter 6, the actual optimisation model implementation used in this study is developed. The test results of the optimisation procedure for the competing algorithms on nine benchmark systems are presented in Chapter 7. Chapter 8 comprises a reliability analysis study comparing reliability surrogate measures to their stochastic counterparts. Chapter 9 contains an application to a recent South African WDS case study, for which substantial cost savings were found. Chapter 10 is a conclusion in which a summary of the findings, contributions of the dissertation, and an appraisal of the contributions areprovided. Chapter 11 describes possible avenues for further research.

1.5 Technical Notes

All units of measurement in this dissertation are in SI units (*International System of Units*), except where particular WDS benchmarks are formulated in alternative unit systems. When

references to equations are made in the body of the dissertation, these will appear with the equation number in round brackets, whereas external references to the literature will appear with the bibliographic entry number in square brackets. There are five appendices. Appendix A contains a brief summary of basic fluid mechanics theory with examples and hydraulic equation derivations. Appendix B contains a summary of prerequisite, miscellaneous mathematical theory. Appendix C contains some illustrative examples of important algorithmic concepts. Appendix D contains a discussion on the use of the optimisation software developed as part of the work towards this dissertation. Finally, Appendix E provides a brief description of the contents of the CD accompanying this dissertation.

Chapter 2

Fluid Mechanics for WDS Analysis

Water distribution systems (WDSs) are designed to transport water from water sources to consumers. The simplicity of this sentence belies a great deal of complexity. Uncertain, time-varying quantities of water must be coaxed to flow to a multitude of heterogenous consumers via a complex pipe network. This must be able to cater for the maximum demand capacity, be delivered within a maximum travel time to avoid quality degradation, be supplied within satisfactory pressure and velocity ranges, and all within the framework of potential system failures and emergency conditions such as fires. As water travels through a distribution system (whose exact hydraulic properties are also uncertain, and change as the system ages) it loses energy (referred to as *head loss*). Furthermore, the natural topography of a service region may vary dramatically from point to point, impacting the effective pressures experienced by WDS users. Care must therefore be taken to ensure that some consumers do not receive very high pressures, whilst others struggle with low pressures. In order to design such a WDS, a sound knowledge of hydraulic behaviour is required. This chapter constitutes a review of essential concepts in fluid mechanics, necessary for hydraulic network analysis. The reader is invited to explore additional hydraulics definitions and derivations in Appendix A.

2.1 Fluid Mechanics Basics

Fluid mechanics is the study of fluids in motion. These fluids contain energy in various forms: chemical energy, kinetic energy and potential energy. In the context of WDSs one considers water at varying degrees of impurity, and one is most interested in kinetic energy in the form of flow (engendered by the force of gravity or pumping), and potential energy embodied in pressure and elevation. Energy is also lost in the form of friction work against the pipe walls. Owing to the negligible changes in density and temperature during ordinary operation, it is common to assume that water is *incompressible* and *isothermal* [170]. However, consideration of internal energy becomes essential when a system experiences extreme temperatures (pipes may burst if water freezes in them). Chemical analysis becomes important when an in-depth water quality study is conducted; however, this is beyond the scope of this dissertation. Some important concepts in hydraulics are presented in this section.

2.1.1 Pressure

Pressure is the force per unit area caused by the weight of the fluid. Pressure, p, is a force, F, acting over an area A, defined as

$$p = \lim_{\Delta A \to 0} \frac{\Delta F}{\Delta A} = \frac{\mathrm{d}F}{\mathrm{d}A}.$$

Pressure at a point is a scalar quantity and is equal in all directions [170]; it is measured in units of Pascal (Pa), where 1 Pa = 1 N/m^2. *Specific weight* γ is the weight per unit volume of the fluid, related to the *fluid density* ρ by $\gamma = \rho g$, where $g = 9.81$ denotes standard gravitational acceleration. For water at 4^oC, its specific weight is 9810 N/m^3 and its density is 1000 kg/m^3. For static fluids, the only variation in hydraulic pressure is with the elevation y in the fluid, that is

$$\frac{dp}{dy} = -\gamma. \tag{2.1}$$

For a static fluid on a horizontal plane, the pressure everywhere on this plane is constant. Considering a constant specific weight, (2.1) may be integrated to obtain $p = -\gamma y + c$, where c is a constant, or

$$\frac{p}{\gamma} + y = \text{constant}. \tag{2.2}$$

The left hand side of (2.2) is known as the *piezometric head*, which is constant throughout any incompressible static fluid. The first term, p/γ, is the *pressure head* (which is what shall be referred to when the word *head* is used in the remainder of this dissertation), and the second term, y, is the elevation above some datum. Piezometric head is also known as *hydraulic head*. It is a measure of the total energy per unit weight above a datum. Piezometric head is measured in units of *height* (m). Pressure and elevation at two different points, 1 and 2, in the fluid must satisfy

$$\frac{p_1}{\gamma} + y_1 = \frac{p_2}{\gamma} + y_2.$$

Hydraulic head may be used to determine a hydraulic gradient between two or more points. Fluid always flows down a hydraulic gradient from a higher to a lower total head (hydraulic head plus velocity head). In a closed hydraulic system, a pressure change produced at one point is transmitted throughout the entire system (this effect is caused by a *pressure wave* and travels at close to the speed of sound). This principle is known as *Pascal's law*. Such a pressure change might be brought on by a pump being switched on or a valve being closed [42].

A *piezometer* is a simple device for measuring pressure, which works by utilizing the change in pressure with elevation. Figure 2.1 shows an example of a piezometer attached to a pipe. The pressure at the exposed surface is that of atmospheric pressure, p_{atm}. Therefore, the *gauge pressure* in the middle of the pipe, at a distance h below the water surface level, is $p = \gamma h$, and the *absolute pressure* is $p_{\text{abs}} = \gamma h + p_{\text{atm}}$.

2.1.2 Flow

Discharge or *flow rate*, q, is the volume rate of flow ($\frac{\mathrm{d}V}{\mathrm{d}t}$) that passes a given section in a flow stream (*e.g.* a pipe section), and has SI units of m^3/s. If flow velocity is constant throughout a section of pipe, then $q = vA$, where v is the velocity and A is the cross-sectional area of the pipe. The same equation may apply if v represents a constant mean velocity. However, the

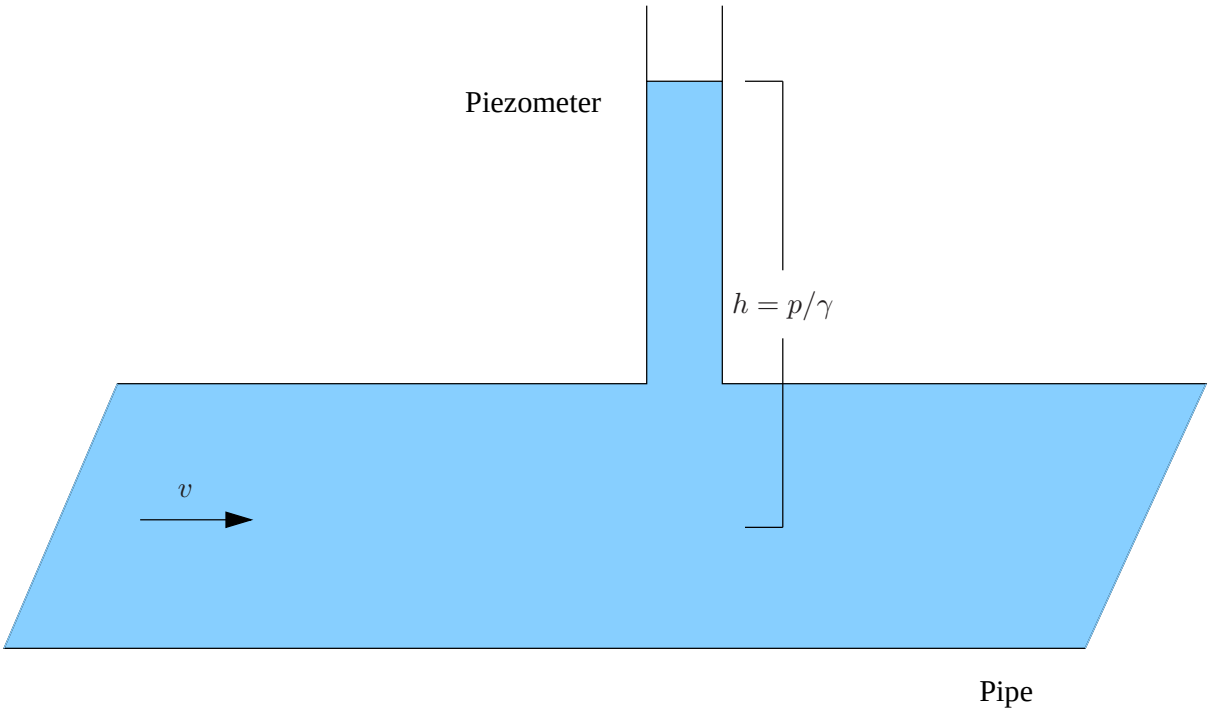


Figure 2.1: *Piezometer attached to a pipe.*

actual flow velocity $\mathbf{v}$ varies across a flow field, as seen in the example of pipe flow in Figure 2.2. Thus, in a real scenario discharge is the integral across the section; that is

$$\frac{\mathrm{d}V}{\mathrm{d}t} = q = \int_A \mathbf{v} \cdot \mathrm{d}\mathbf{A},$$

where $\mathbf{v}$ is the velocity vector for each differential area $\mathrm{d}A$, and $\mathrm{d}\mathbf{A}$ is the area vector oriented normal to $\mathrm{d}A$ with the same magnitude as the area [42].

The *mean velocity*, $\overline{v}$, of a fluid is defined as its discharge divided by the total cross-sectional area, $\overline{v} = \frac{q}{A}$. To simplify pipe-flow analysis, one typically considers only the one-dimensional mean velocity in a pipe, in which case the bar over the velocity may be dropped[1] [42].

Mass flow is simply the incorporation of density, ρ, in the discharge equation, yielding

$$\begin{aligned} \frac{\mathrm{d}m}{\mathrm{d}t} &= \int_A \rho\mathbf{v} \cdot \mathrm{d}\mathbf{A} \\ &= \rho \int_A \mathbf{v} \cdot \mathrm{d}\mathbf{A} \\ &= \rho q. \end{aligned}$$

Hydraulic analysis is often simplified by considering a system in *steady-state*. In this case, the flow itself is *steady* ($\frac{\mathrm{d}\mathbf{v}}{\mathrm{d}t} = 0$), and hence the mass in a control volume is constant over time. Such an analysis is useful for macroscopic planning, and the majority of hydraulic engineering design is conducted on this basis. However, one must keep in mind the existence of flow variation, especially with regards to sudden changes in flow which may cause so-called *transients* or *water*

[1] The symbol v is used throughout this dissertation to denote steady-state mean velocity for one-dimensional flow in a pipe. Velocity is used rather than speed, since flow always occurs in one or the other direction along a pipe.

hammer effects, where strong pressure waves rush through a system. While transient analysis is usually relegated to be a secondary consideration in hydraulic engineering design, it is actually very important. Poorly designed systems may experience extreme transient events with the potential of causing serious component damage. This is also an operational question, since transient problems may usually be avoided by inducing changes slowly [251].

Flow may either be *laminar* (smooth, constant flow profile) or *turbulent* (chaotic flow marked by strong eddies of current), where laminar flow is associated with smooth pipes at slow flow rates and turbulent flow is more likely found at higher velocities in rough pipes. Turbulence is characterized by the so-called *Reynolds number*, R_e (see Appendix A for details). In practice, most flow in WDSs is turbulent. The energy equations used in practice try to account for both flow types [169].

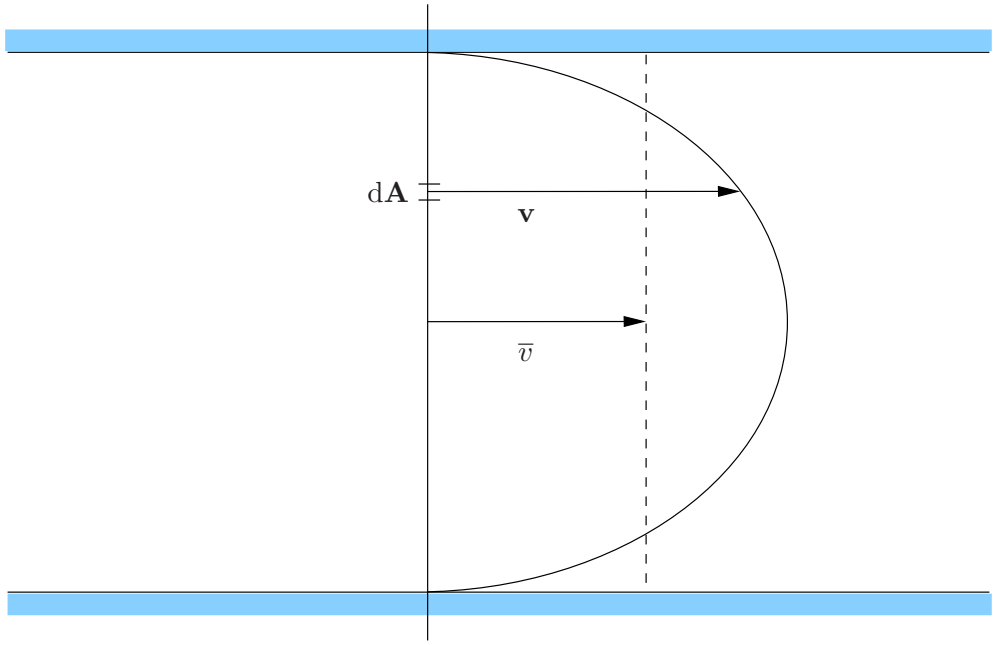


Figure 2.2: *Velocity distribution in a pipe flow.*

2.1.3 Control Volume Approach

The so-called *control volume approach* may be used to develop continuity and energy equations for hydraulic systems. A *control volume* (denoted by CV) is a fixed volumetric region in space through which fluid may flow freely, bounded by a *control surface* (denoted by CS). An example of this might be a pipe segment bounded by the pipe walls extending a short distance in either direction along the pipe. A given body of fluid of constant mass (a *fluid system*) may possess *extensive* properties which apply to the entire fluid system (*e.g.* mass), or *intensive* properties which apply to the fluid unit mass (*e.g.* velocity) [42].

By considering the relationship between a control volume and a fluid system, one is able to develop a general equation for the change of any extensive property, B, of the fluid system in terms of an intensive property, b, of the system (see Appendix A). This yields the general

control volume equation,

$$\frac{\mathrm{d}B_{\mathrm{sys}}}{\mathrm{d}t} = \frac{\mathrm{d}}{\mathrm{d}t}\int_{\mathrm{CV}} b\rho\,\mathrm{d}V + \int_{\mathrm{CS}} b\rho\mathbf{v}\cdot\mathrm{d}\mathbf{A}. \tag{2.3}$$

The first term on the right-hand side of the equation is the instantaneous change over time of B inside the control volume (which is zero for steady flow), and the second term represents the net outflow rate of B from the control volume [42].

2.1.4 Continuity of Flow

The general control volume equation may be used to derive the continuity equation (see Appendix A) for flow with a uniform velocity across the flow section and constant density,

$$\sum_{\mathrm{CS}} \mathbf{v}\cdot\mathbf{A} = -\frac{\mathrm{d}}{\mathrm{d}t}\int_{\mathrm{CV}} \mathrm{d}V.$$

For constant density, steady, one-dimensional flow, such as water flowing in a conduit, the equation becomes

$$\sum_{\mathrm{CS}} vA = 0.$$

This equation expresses *continuity of flow*, one of the most important conservation laws in hydraulics. Considering a control volume between locations 1 and 2 in a pipe, the continuity equation delivers the relationship

$$q_1 = v_1A_1 = v_2A_2 = q_2.$$

2.1.5 Hydraulic Energy

The control volume equation may be combined with the *first law of thermodynamics* to develop the energy equation for fluid flow in hydraulic processes. This energy balance forms an accounting of the energy inputs and outputs to and from a system [42, 170]. The first law of thermodynamics states that the rate of change of energy with time is the rate at which heat is transferred into the fluid, $\mathrm{d}H/\mathrm{d}t$, less the rate at which the fluid performs work on its surroundings, $\mathrm{d}W/\mathrm{d}t$, expressed as $\frac{\mathrm{d}E}{\mathrm{d}t} = \frac{\mathrm{d}H}{\mathrm{d}t} - \frac{\mathrm{d}W}{\mathrm{d}t}$. The total energy of a fluid system is the sum of the internal energy E_{u}, the kinetic energy E_{k}, and the potential energy E_{p}, with unit mass equivalents, $e_{\mathrm{u}}, e_{\mathrm{k}}$, and e_{p}. Total energy is therefore $E = E_{\mathrm{u}} + E_{\mathrm{k}} + E_{\mathrm{p}}$. The kinetic energy per unit mass is the total kinetic energy of the mass with velocity v divided by the mass m, $e_{\mathrm{k}} = \frac{mv^2/2}{m} = \frac{v^2}{2}$. The potential energy per unit mass is the weight of the fluid, γV, multiplied by the centroid elevation z divided by the mass m, producing $e_{\mathrm{p}} = \frac{\gamma Vz}{m} = \frac{\gamma Vz}{\rho V} = gz$.

Combining the first law of thermodynamics and the expression for total energy in the one-dimensional version of the general control volume equation yields the *general energy equation for unsteady variable-density uniform flow*,

$$\begin{aligned}
\frac{\mathrm{d}E}{\mathrm{d}t} = \frac{\mathrm{d}H}{\mathrm{d}t} - \frac{\mathrm{d}W}{\mathrm{d}t} &= \frac{\mathrm{d}}{\mathrm{d}t}\int e\rho\,\mathrm{d}V + \sum_{\mathrm{CS}} e\rho\mathbf{v}\cdot\mathbf{A} \\
&= \frac{\mathrm{d}}{\mathrm{d}t}\int (e_{\mathrm{u}} + e_{\mathrm{k}} + e_{\mathrm{p}})\rho\,\mathrm{d}V + \sum_{\mathrm{CS}} (e_{\mathrm{u}} + e_{\mathrm{k}} + e_{\mathrm{p}})\rho\mathbf{v}\cdot\mathbf{A} \\
&= \frac{\mathrm{d}}{\mathrm{d}t}\int (e_{\mathrm{u}} + \frac{v^2}{2} + gz)\rho\,\mathrm{d}V + \sum_{\mathrm{CS}} (e_{\mathrm{u}} + \frac{v^2}{2} + gz)\rho\mathbf{v}\cdot\mathbf{A},
\end{aligned}$$

which for steady flow, reduces to

$$\frac{\mathrm{d}H}{\mathrm{d}t} - \frac{\mathrm{d}W}{\mathrm{d}t} = \sum_{\mathrm{CS}} (e_{\mathrm{u}} + \frac{v^2}{2} + gz)\rho \mathbf{v} \cdot \mathbf{A}. \tag{2.4}$$

The work done by a system on its surroundings may further be divided into flow work, W_{f}, and shaft work, W_{s} (see Appendix A). Flow work per unit volume is the work done by the pressure force in order to move the fluid a distance $v\Delta t$. This may be incorporated into (2.4) as

$$\frac{\mathrm{d}W_{\mathrm{f}}}{\mathrm{d}t} = \sum_{\mathrm{CS}} p\mathbf{v} \cdot \mathbf{A} = \sum_{\mathrm{CS}} \frac{p}{\rho}\rho \mathbf{v} \cdot \mathbf{A}.$$

The general energy equation for steady, uniform flow now becomes

$$\frac{\mathrm{d}H}{\mathrm{d}t} - \frac{\mathrm{d}W_{\mathrm{s}}}{\mathrm{d}t} = \sum_{\mathrm{CS}} (\frac{p}{\rho} + e_{\mathrm{u}} + \frac{v^2}{2} + gz)\rho \mathbf{v} \cdot \mathbf{A}. \tag{2.5}$$

2.1.6 Pressurised Pipe Flow

Hydraulics is the study of liquid flow in pipes and open channels. *Pressurised pipe flow* refers to the flow of a fluid under pressure in a pipe, where the interior walls of the pipe are in contact with the fluid at all times (*i.e.* there is no free surface). Open channel flow, where there is a free surface, may also occur in pipes, when a pipe is not full. This is often applicable to gravity systems such as sewerage networks.

Using the general energy equation for steady flow (2.5), and considering pipe flow between sections 1 and 2, the energy equation for pipe flow may be expressed as

$$\frac{\mathrm{d}H}{\mathrm{d}t} - \frac{\mathrm{d}W_{\mathrm{s}}}{\mathrm{d}t} = \sum_{\mathrm{A}_2} \left(\frac{p_2}{\rho} + e_{\mathrm{u}_2} + \frac{v_2^2}{2} + gz_2 \right) \rho \mathbf{v}_2 \cdot \mathbf{A}_2 - \sum_{\mathrm{A}_1} \left(\frac{p_1}{\rho} + e_{\mathrm{u}_1} + \frac{v_1^2}{2} + gz_1 \right) \rho \mathbf{v}_1 \cdot \mathbf{A}_1,$$

which may be modified to

$$\begin{aligned} \frac{\mathrm{d}H}{\mathrm{d}t} - \frac{\mathrm{d}W_{\mathrm{s}}}{\mathrm{d}t} \quad = \quad & \sum_{\mathrm{A}_2} \left(\frac{p_2}{\rho} + e_{\mathrm{u}_2} + gz_2 \right) \rho \mathbf{v}_2 \cdot \mathbf{A}_2 - \sum_{\mathrm{A}_1} \left(\frac{p_1}{\rho} + e_{\mathrm{u}_1} + gz_1 \right) \rho \mathbf{v}_1 \cdot \mathbf{A}_1 \\ & + \sum_{\mathrm{A}_2} \frac{\rho v_2^3}{2} A_2 - \sum_{\mathrm{A}_1} \frac{\rho v_1^3}{2} A_1. \end{aligned} \tag{2.6}$$

Flow is uniform between sections 1 and 2, so that hydrostatic conditions prevail across each individual section. Therefore $p/\rho + e_{\mathrm{u}} + gz$ is constant and may be taken outside the summation. Also, the term $\rho v A = \dot{m}$ is the mass flow rate, so that $\sum_A (\rho v^3/2)\,\mathrm{d}A = (\rho v^3/2)A = \dot{m}(v^2/2)$. Equation (2.6) now becomes

$$\frac{\mathrm{d}H}{\mathrm{d}t} - \frac{\mathrm{d}W_{\mathrm{s}}}{\mathrm{d}t} = (\frac{p_2}{\rho} + e_{\mathrm{u}_2} + gz_2)\dot{m} - (\frac{p_1}{\rho} + e_{\mathrm{u}_1} + gz_1)\dot{m} + \dot{m}\frac{v_2^2}{2} - \dot{m}\frac{v_1^2}{2}.$$

This may be divided through by $\dot{m}$ and rearranged to yield

$$\frac{1}{\dot{m}} \left(\frac{\mathrm{d}H}{\mathrm{d}t} - \frac{\mathrm{d}W_{\mathrm{s}}}{\mathrm{d}t} \right) + \frac{p_1}{\rho} + e_{\mathrm{u}_1} + gz_1 + \frac{v_1^2}{2} = \frac{p_2}{\rho} + e_{\mathrm{u}_2} + gz_2 + \frac{v_2^2}{2}. \tag{2.7}$$

At this stage it is convenient to separate shaft work W_s into work expended on a turbine (W_t) or work done by a pump (W_p). Shaft work may be expressed as

$$\frac{\mathrm{d}W_\mathrm{s}}{\mathrm{d}t} = \frac{\mathrm{d}W_\mathrm{t}}{\mathrm{d}t} - \frac{\mathrm{d}W_\mathrm{p}}{\mathrm{d}t}. \tag{2.8}$$

Substituting (2.8) into (2.7) and dividing through by g yields

$$\frac{1}{\dot{m}g}\frac{\mathrm{d}W_\mathrm{p}}{\mathrm{d}t} + \frac{p_1}{\gamma} + z_1 + \frac{v_1^2}{2g} = \frac{1}{\dot{m}g}\frac{\mathrm{d}W_\mathrm{t}}{\mathrm{d}t} + \frac{p_2}{\gamma} + z_2 + \frac{v_2^2}{2g} + \left[\frac{e_{\mathrm{u}_2} - e_{\mathrm{u}_1}}{g} - \frac{1}{\dot{m}g}\frac{\mathrm{d}H}{\mathrm{d}t}\right]. \tag{2.9}$$

The following terms are identified from (2.9): The head supplied by the pumps is

$$h_\mathrm{p} = \frac{1}{\dot{m}g}\frac{\mathrm{d}W_\mathrm{p}}{\mathrm{d}t}.$$

The head given up to the turbines is

$$h_\mathrm{t} = \frac{1}{\dot{m}g}\frac{\mathrm{d}W_\mathrm{t}}{\mathrm{d}t}.$$

The head loss (mechanical energy loss due to viscous stress) is

$$h_\mathrm{L} = \left[\frac{e_{\mathrm{u}_2} - e_{\mathrm{u}_1}}{g} - \frac{1}{\dot{m}g}\frac{\mathrm{d}H}{\mathrm{d}t}\right]. \tag{2.10}$$

The term $(e_{\mathrm{u}_2} - e_{\mathrm{u}_1})/g$ represents the finite increase in internal energy of the flow system, because a portion of the mechanical energy is converted into thermal energy through the viscous action between the fluid particles. The term $-\frac{1}{\dot{m}g}\frac{\mathrm{d}H}{\mathrm{d}t}$ represents the heat generated through energy dissipation that escapes the system. Equation (2.9) may now be expressed as

$$\frac{p_1}{\gamma} + z_1 + \frac{v_1^2}{2g} + h_\mathrm{p} = \frac{p_2}{\gamma} + z_2 + \frac{v_2^2}{2g} + h_\mathrm{t} + h_\mathrm{L}. \tag{2.11}$$

This is the full form of the famous *Bernoulli equation* for energy conservation between two points in a conduit under steady flow. This equation is expressed with the velocity representing the mean velocity. Here p/γ is the *pressure head*, $v^2/2g$ is the *velocity head* and z is the potential energy associated with the elevation at the centre of the pipe relative to some datum [42, 93, 170].

2.1.7 Hydraulic and Energy Grade Lines

In a graphical representation of a hydraulic system, a side-view section is often used. The energy terms in (2.11) all have units of height. Therefore, it is useful to draw a *hydraulic grade line* (HGL) and an *energy grade line* (EGL) on the same scale drawing of a hydraulic system. The HGL is essentially the line p/γ above the centre of a pipe, which is the distance the water would rise in a piezometer tube attached to the pipe (see piezometer 1 in Figure 2.3). The EGL is a distance of $v^2/2g$ above the HGL. Piezometer 2 in the figure measures the velocity pressure as well, so that the water rises to meet the EGL. In Figure 2.3 it is assumed that there is negligible initial velocity in the reservoir, so that the total energy head at the water surface is simply z_1. At the second measured position, where the height above the datum is z_2, the difference between the energy head (EGL) and the initial energy is the head loss, h_L, caused by the friction force of the fluid travelling in the pipe. Note that the grade lines exhibit a constant

negative gradient along the length of the pipe where the physical characteristics of the pipe are uniform ($\frac{\mathrm{d}}{\mathrm{d}s}(p/\gamma + z) = -c$), and become steeper when the fluid encounters a new pipe with different hydraulic properties. The positive effect of a pump on the hydraulic and energy grade lines is shown in Figure 2.4. Just as the pump causes the system to gain energy, a turbine or valve may engender a sudden drop of the grade lines. Also demonstrated in Figure 2.4 is the increase in velocity head due to a reduced diameter pipe [42, 93].

2.1.8 Head Losses

Expression (2.10) defines the head loss variable, h_{L}. Head losses in pressurised pipe flow may occur by various means. Calculating these losses is a major part of performing a hydraulic simulation on a network.

Shear-Stress Distribution of Flow in Pipes

Shear-stress is the force tangential and opposite to the flow direction caused by friction against the pipe walls. Shear stress is expressed as

$$\tau = \mu \frac{\mathrm{d}v}{\mathrm{d}y}, \tag{2.12}$$

where the proportionality factor μ is called the *dynamic viscosity* of the fluid and y is the distance from the pipe wall (see Appendix A for a detailed discussion). The velocity distribution in a pipe is directly linked to the shear-stress distribution. Consider steady flow in a pipe of uniform cross-section, forming a cylindrical element of fluid with cylinder length Δs, radius r_0 and cross-sectional area A (as shown in Figure 2.5). Let the flow be in a direction from pipe section 1 to pipe section 2 (the two ends of the system). In general, this pipe may be at any angle θ with respect to the horizontal plane. Under uniform flow, the general form of the integral momentum equation in the s-direction (in the plane of the pipe's axis) is expressed by

$$\sum F_s = \frac{\mathrm{d}}{\mathrm{d}t} \int_{\mathrm{CV}} v_s \rho \, \mathrm{d}V + \sum_{\mathrm{CS}} v_s \rho \mathbf{v} \cdot \mathbf{A},$$

where $\frac{\mathrm{d}}{\mathrm{d}t} \int_{\mathrm{CV}} v_s \rho \, \mathrm{d}V = 0$ because the flow is steady and where $\sum_{\mathrm{CS}} v_s \rho \mathbf{v} \cdot \mathbf{A} = 0$ because there is no net flow of momentum through the control surface. Therefore $\sum F_s = 0$ (because the pressure across any section of the face of the fluid will be hydrostatically distributed). The forces acting on the system are the *pressure force*, the *gravitational force*, and the *shearing force*. The pressure forces are $F_{p1} = pA$ and $F_{p2} = (p + \mathrm{d}p/\mathrm{d}s\Delta s)A$, acting at the lower and higher ends of the system. The gravity force is $F_g = \gamma A \Delta s \sin\theta$. The shearing force is $F_\tau = \tau(2\pi r \Delta s)$, where τ is the shear stress. The sum of the forces is

$$F_{p1} - F_{p2} - F_g - F_\tau = 0. \tag{2.13}$$

The negative signs indicate that these forces are in the opposite direction to that of the flow. Equation (2.13) may therefore be written as

$$pA - \left(p + \frac{\mathrm{d}p}{\mathrm{d}s}\Delta s\right) A - \gamma A \Delta s \sin\theta - \tau(2\pi r \Delta s),$$

which simplifies to

$$-\left(\frac{\mathrm{d}p}{\mathrm{d}s}\Delta s\right) A - \gamma A \Delta s \sin\theta - \tau(2\pi r \Delta s) = 0.$$

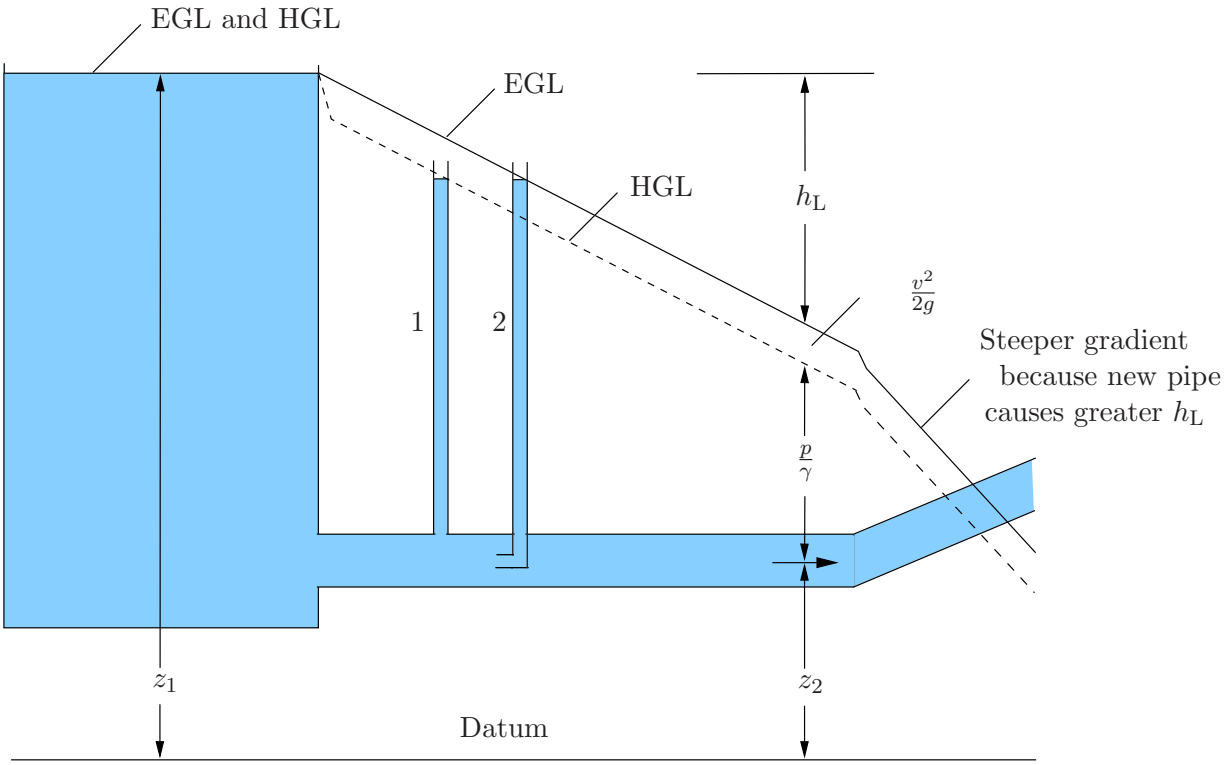


Figure 2.3: *Energy and hydraulic grade lines for a reservoir and pipe system.*

Setting $\mathrm{d}z = \sin\theta\,\mathrm{d}s$, shear stress may be solved for, yielding

$$\tau = \frac{r}{2}\left[-\frac{\mathrm{d}}{\mathrm{d}s}(p+\gamma z)\right]. \tag{2.14}$$

Expression (2.14) indicates that τ is zero at the centre of the pipe, where $r = 0$, and increases linearly to a maximum at the pipe wall. Furthermore, $p + \gamma z$ is constant across a cross-section, because the streamlines are straight and parallel in a uniform flow so that there is no acceleration of fluid normal to the streamline. The gradient $\mathrm{d}(p+\gamma z)/\mathrm{d}s$ is negative and constant across the section for uniform flow (see §2.1.7) [42, 93, 170].

Laminar Flow

Adapting (2.12) by setting $y = r$, one obtains $\tau = \mu\,\mathrm{d}v/\mathrm{d}r$. Substituting this into (2.14) yields

$$\tau = \mu\frac{\mathrm{d}v}{\mathrm{d}r} = \frac{r}{2}\left[-\frac{\mathrm{d}}{\mathrm{d}s}(p+\gamma z)\right].$$

This expression may be integrated over a cross-section by separation of variables, using the boundary condition $v = 0$ when $r = r_0$. This yields

$$v = \frac{r_0^2 - r^2}{4\mu}\left[-\frac{\mathrm{d}}{\mathrm{d}s}(p+\gamma z)\right]. \tag{2.15}$$

Expression (2.15) indicates that the velocity distribution for laminar pipe flow is parabolic across a section and has a maximum velocity at the pipe centre. Laminar flow in a cylindrical pipe is known as *Hagen-Poiseuille* flow.

It is often desirable to relate the pressure change to the rate of flow or mean velocity, $\overline{v}$, in a conduit. Therefore it is necessary to integrate $\mathrm{d}q = v\,\mathrm{d}A$ over the cross-sectional flow area. That is,

$$\begin{aligned} q &= \int_A v\,\mathrm{d}A \\ &= \int_0^{r_0} \frac{r_0^2 - r^2}{4\mu}\left[-\frac{\mathrm{d}}{\mathrm{d}s}(p+\gamma z)\right](2\pi r)\,\mathrm{d}r \\ &= \frac{\pi}{4\mu}\left[-\frac{\mathrm{d}}{\mathrm{d}s}(p+\gamma z)\right]\int_0^{r_0}(r_0^2 - r^2)2r\,\mathrm{d}r \\ &= \frac{\pi}{4\mu}\left[\frac{\mathrm{d}}{\mathrm{d}s}(p+\gamma z)\right]\frac{\left(r^2 - r_0^2\right)^2}{2}\Bigg|_0^{r_0} \\ &= \frac{\pi r_0^4}{8\mu}\left[-\frac{\mathrm{d}}{\mathrm{d}s}(p+\gamma z)\right]. \end{aligned} \tag{2.16}$$

Dividing (2.16) through by the cross-sectional area yields

$$\overline{v} = \frac{r_0^2}{8\mu}\left[-\frac{\mathrm{d}}{\mathrm{d}s}(p+\gamma z)\right]. \tag{2.17}$$

Comparing (2.17) and (2.15) shows that $\overline{v} = v_{\max}/2$. Substituting pipe diameter $D/2$ for r_0

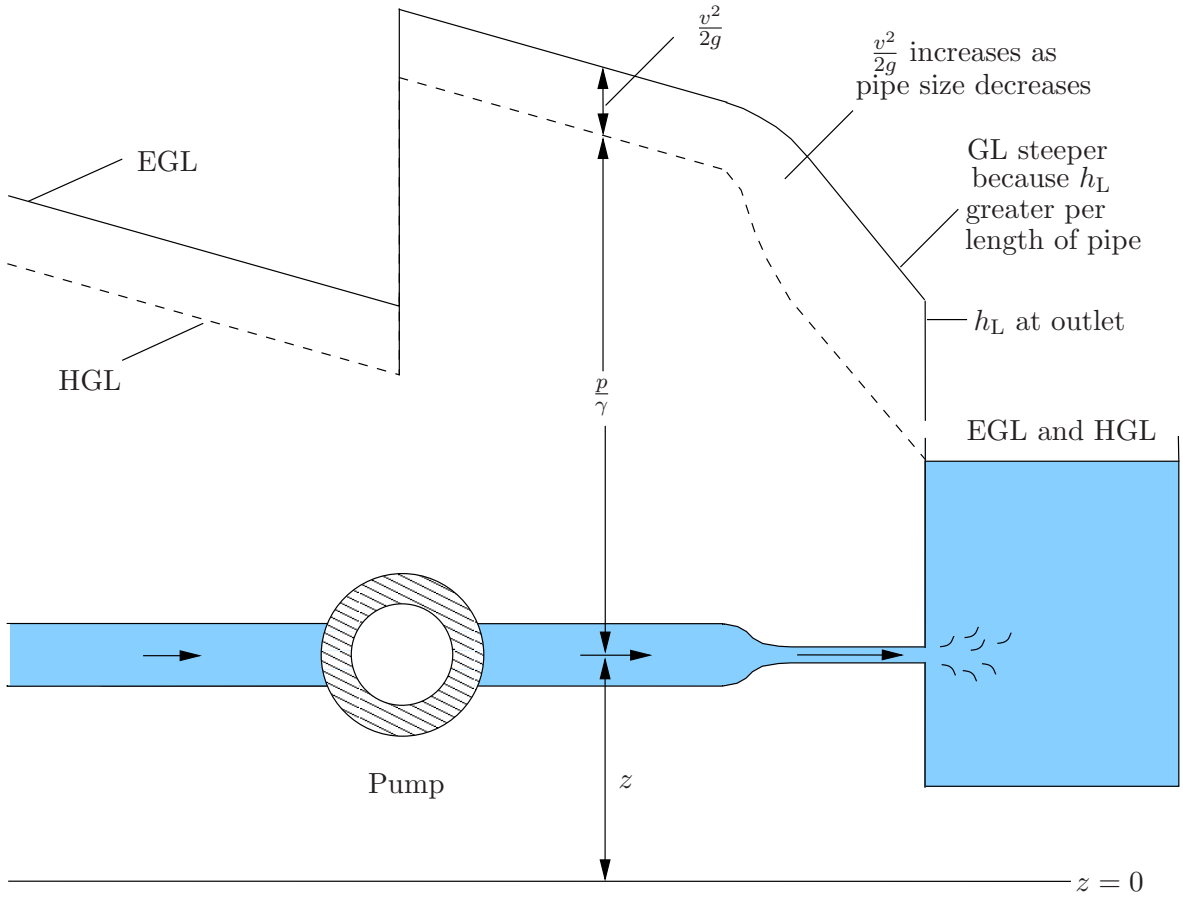


Figure 2.4: *Energy and hydraulic grade lines for a hydraulic system with a pump and a change in pipe diameter.*

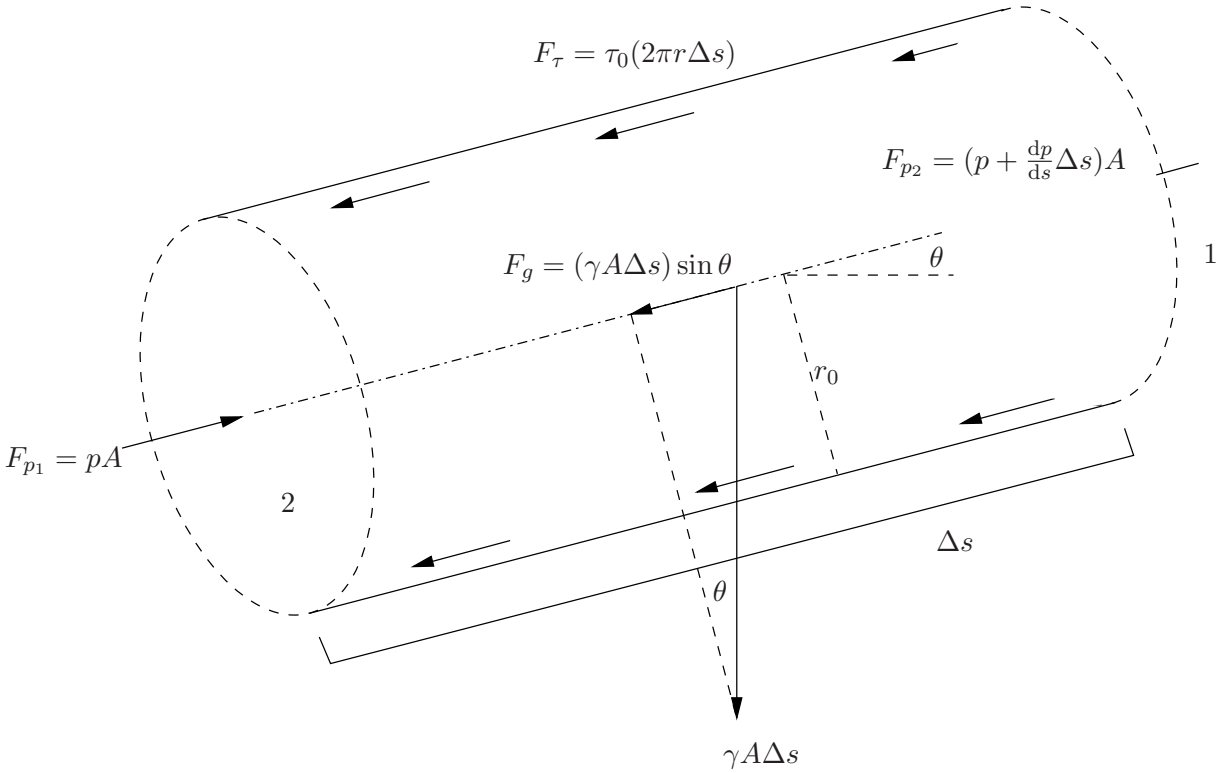


Figure 2.5: *A cylindrical fluid element in a pipe.*

gives

$$\overline{v} = \frac{D^2}{32\mu}\left[-\frac{\mathrm{d}}{\mathrm{d}s}(p + \gamma z)\right]$$

or

$$\frac{\mathrm{d}}{\mathrm{d}s}(p + \gamma z) = -\frac{32\mu\overline{v}}{D^2}. \tag{2.18}$$

Integrating (2.18) along the pipe from section 1 to 2, one obtains

$$p_2 - p_1 + \gamma(z_2 - z_1) = -\frac{32\mu\overline{v}}{D^2}(s_2 - s_1),$$

which may be rewritten as

$$\frac{p_1}{\gamma} + z_1 = \frac{p_2}{\gamma} + z_2 + \frac{32\mu\overline{v}\Delta s}{D^2}.$$

This demonstrates that when the general energy equation for incompressible flow in conduits, (2.11), is reduced to one for uniform laminar flow in a constant diameter pipe, the result is

$$\frac{p_1}{\gamma} + z_1 = \frac{p_2}{\gamma} + z_2 + h_\mathrm{f},$$

where $h_\mathrm{f} = 32\mu\overline{v}\Delta s/D^2$ is used instead of h_L to signify the head loss due to the frictional resistance of the pipe.

Turbulent Flow in Smooth Pipes

Pipes are typically produced to have smooth walls, but degenerate and accumulate dirt with age. One may also clean them to improve smoothness. The following velocity distribution

equations are based on experimental results [43, 170]. If $u_* = \sqrt{\tau_0/\rho}$ is the shear velocity, where τ_0 is the shear stress at the pipe wall, then

$$\frac{v}{u_*} = \frac{u_* y}{\nu} \quad \text{for} \quad 0 < \frac{u_* y}{\nu} < 5$$

and

$$\frac{v}{u_*} = 5.75 \log \frac{u_* y}{\nu} + 5.5 \quad \text{for} \quad 20 < \frac{u_* y}{\nu} < 10^5,$$

where y is the distance from the pipe wall and ν is the kinematic viscosity of the fluid (see Appendix A). The velocity distribution for turbulent flow may also be approximated using power law formulas of the form $v/v_{\max} = (y/r_0)^c$, where $v_{\max}$ is the velocity at the centre of the pipe, r_0 is the pipe radius and c is an exponent that increases along with the Reynolds number (see [43] for typical values of c).

Turbulent Flow in Rough Pipes

Pipes may become rough with age, producing different results depending on the type of pipe and fluid load. Experimental results suggest that the velocity distribution of turbulent flow in rough pipes may be represented by

$$\frac{v}{u_*} = 5.75 \log \frac{y}{k} + \varepsilon,$$

where k is a measure of the height of the roughness elements, and ε is a function of the roughness characteristics. In 1933, Nikuradse [183] did extensive work on turbulent flow, measuring the resistance to flow posed by various pipes with uniform sand grains glued onto their inside walls. Although commercial pipes have some degree of spatial variance in their degree of roughness, they may have the same characteristics as do pipes with a uniform distribution of sand grains of size $k = k_s$ [169]. Through experimentation, Nikuradse was able to determine a value of $\varepsilon = 8.5$. The value of y was taken as the geometric mean of the wall surface, yielding

$$\frac{v}{u_*} = 5.75 \log \frac{y}{k_s} + 8.5.$$

This work revealed that for low Reynolds numbers and small sand grains, the flow resistance is a close approximation to that of smooth pipes. This may be explained by the roughness elements becoming submerged in a viscous sublayer. Also, for high Reynolds numbers, the resistance coefficient is only a function of the relative roughness, k_s/D. Here the viscous sublayer is very thin, so that the roughness elements project into the flow stream causing flow resistance from drag [42, 170].

Head Losses from Pipe Friction

Numerous expressions have been proposed to calculate head losses due to pipe friction, including the *Chezy*, *Manning*, *Scobey*, *Hazen-Williams* and *Darcy Weisbach* expressions. These expressions relate the friction losses to the physical characteristics of the pipe and the flow parameters. The Darcy-Weisbach expression is the most accurate, because it is scientifically based and applies to both laminar and turbulent flow. Given a pipe of length L and diameter D, the Darcy-Weisbach expression for frictional head loss is

$$h_{L_f} = f \frac{L}{D} \frac{\overline{v}^2}{2g},$$

where f is a dimensionless friction factor which is a function of the Reynolds number, R_e, and relative roughness, k_s/D. Here k_s is the average non-uniform roughness of the pipe. This value must often be determined experimentally, although standard tables do exist with roughness values for commercial pipes of different materials and age categories [169]. The Darcy-Weisbach expression has been derived for turbulent flow and can only be applied to laminar flow (usually $R_e < 2000$) using a friction factor of $f = \frac{64}{R_e}$, which is actually equivalent to the *Hagen-Poiseuille* expression. The equation for turbulent flow in a smooth pipe with $R_e > 3000$ is

$$\frac{1}{\sqrt{f}} = 2\log_{10}\left(R_e\sqrt{f}\right) - 0.8, \tag{2.19}$$

and for a rough pipe,

$$\frac{1}{\sqrt{f}} = 2\log_{10}\frac{D}{k_s} + 1.14. \tag{2.20}$$

Equations (2.19) and (2.20) were proposed by Von Karman and Prandtl, based on experimental work done by Nikuradse in 1932 [170, 257]. In 1939, Colebrook and White proposed the semi-empirical formula

$$\frac{1}{\sqrt{f}} = 2\log_{10}\left(\frac{k_s/D}{3.7} + \frac{2.51}{R_e\sqrt{f}}\right).$$

This formula is asymptotic to both (2.19) and (2.20). In 1944 Moody used this equation, in combination with experimental data on commercial pipes, to develop the Moody diagram (see Appendix A). This diagram provides the friction factor f for different values of the Reynolds number and the relative roughness.

The empirical Hazen-Williams equation is also used frequently in practice, with head loss at a particular diameter approximated as

$$h_L = \frac{kL}{C_{hw}^{1.85}D^{4.87}}q^{1.85},$$

where L is the length of the pipe, where C_{hw} is the empirical Hazen-Williams pipe coefficient, and where k is a conversion factor with $k = 4.73$ for imperial units and $k = 10.583$ for metric units.

For ease of explanation, the head loss equation is often presented in the simplified form $h_L = Kq^\eta$, where K is called the *pipe* (*loss*) *coefficient*. For example, in the Darcy-Weisbach expression, $\eta = 2$ and $K = f\frac{2L}{\pi D^3 g}$ [42, 169, 170].

Minor (Form) Losses

Head losses may also be caused by inlets, outlets, bends, fittings, valves, expansions, contractions and other denizens of the hydraulic world. Collectively these losses are known as *minor losses*, *form losses* or *secondary losses*, and are caused by flow separation and the generation of turbulence. An example of flow separation occurring at a sharp pipe inlet is shown in Figure 2.6.

Minor head losses produced may be expressed as

$$h_{L_m} = K\frac{v^2}{2g},$$

where K is the *loss coefficient* which has been documented for various components. In all the formulas which follow v_1 denotes the flow velocity before the pipe change and v_2 denotes the

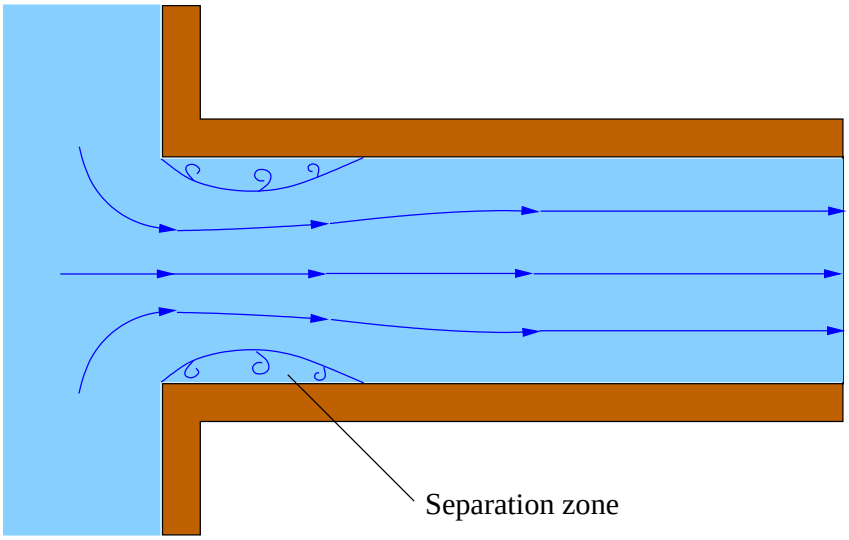


Figure 2.6: *Flow separation at a sharp inlet causing turbulence and head loss.*

velocity after the change. For *sudden expansions* or enlargements of the conduit, head loss may be expressed as

$$h_{\mathrm{L_m}} = \frac{(v_1 - v_2)^2}{2g}.$$

Similarly, for *gradual expansions* (such as *conical diffusers*), the head loss may be expressed as

$$h_{\mathrm{L_m}} = K_{\mathrm{ex}} \frac{(v_1 - v_2)^2}{2g},$$

where K_{ex} is a function of the expansion (cone) angle. The head loss due to a *sudden contraction* may be expressed as

$$h_{\mathrm{L_m}} = K_{\mathrm{co}} \frac{v_2^2}{2g},$$

where the values of K_{co} are a function of the diameter ratios D_2/D_1. *Entrance losses* are calculated by means of the expression

$$h_{\mathrm{L_m}} = K_{\mathrm{en}} \frac{v^2}{2g},$$

where K_{en} depends on the geometry of the entrance. *Exit or discharge losses* from the end of a pipe into a reservoir that has a negligible velocity constitutes the entire velocity head, expressed as

$$h_{\mathrm{L_m}} = \frac{v_2^2}{2g}.$$

Using the concept of *integral roughness*, minor losses may also be accommodated approximately by adapting pipe roughness coefficients. These values are determined experimentally and are often categorized by pipe material [103, 169].

2.1.9 Pipe Flow in Simple Networks

Pipe networks may include pipes in series, parallel pipes and/or branching pipes. These simple configurations may be converted to an *equivalent pipe*. Two pipes are equivalent when they deliver the same rate of flow for the same head loss. This is useful in simplifying and analyzing networks.

Single Pump in a Pipe System

The head produced by a centrifugal pump in a hydraulic system is a function of discharge, q. Such a pump is associated with a pump equation of head versus discharge, specifying the operating range of the pump when running at a specific speed. The resulting characteristic pump curves are usually supplied by the manufacturer. A similar system equation (or curve) exists for any hydraulic system, specifying the possible head values that a system can produce in a particular configuration (connected arrangement of pipes, pumps, valves *etc.*) for different volume throughputs. It is at the intersection of the pump and system curves that operation will occur (known as the *system operating point*). An example of the intersection of the system and pump curves is provided in Figure 2.7. A direct analytical solution of the intersection point is usually not possible, and a solution must be computed via numerical approximation. Consider

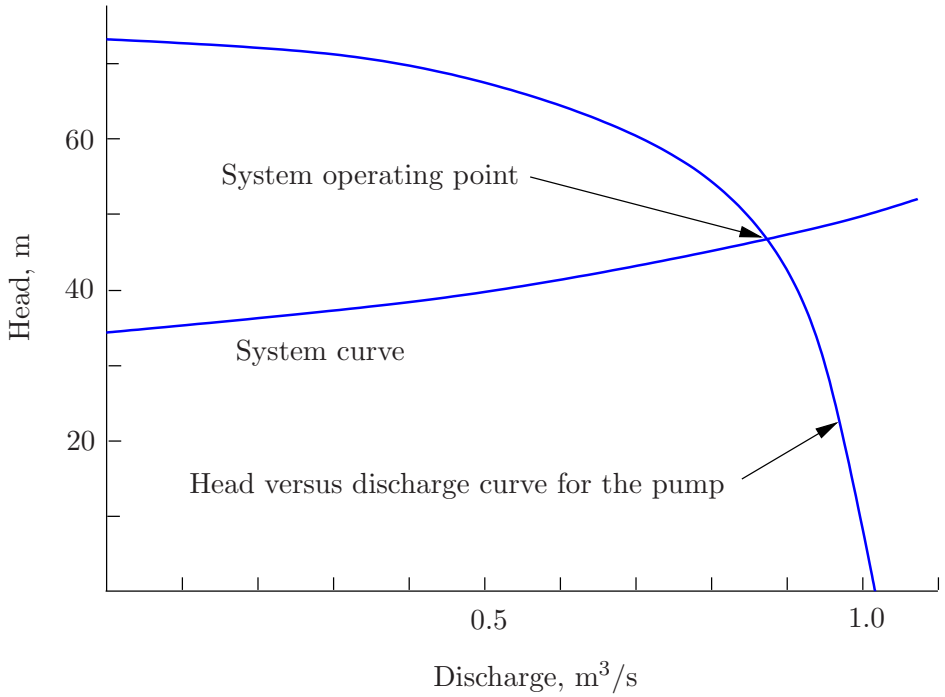


Figure 2.7*: System and pump curves intersecting at the operating point.*

the simple reservoir-pump-reservoir system shown in Figure 2.8. This obeys the energy equation

$$\frac{p1}{\gamma} + z_1 + \frac{v_1^2}{2g} + h_\mathrm{p} = \frac{p2}{\gamma} + z_2 + \frac{v_2^2}{2g} + \sum K\frac{v^2}{2g} + \sum \frac{fLv^2}{D2g}. \tag{2.21}$$

Since the system has an initial velocity of zero and the pipe size is homogeneous, (2.21) simplifies to

$$h_\mathrm{p} = (z_2 - z_1) + \frac{v^2}{2g}\left(1 + \sum K + \sum \frac{fL}{D}\right). \tag{2.22}$$

This means that, for a given discharge, a certain head h_p must be supplied to maintain that flow. The system operating point is where the head produced by the pump is just the amount required to overcome the system head losses and elevation differences. The pump equation of head versus discharge is often approximated by means of an expression of the form $h_\mathrm{p} = aq^2 + bq + c$, or $h_\mathrm{p} = c - dq^\eta$, where a, b and d are constant coefficients, and c represents the maximum head at $q = 0$.

Pumps must consume electricity to produce hydraulic energy. However, pumps are not 100% energy efficient. They are associated with a *wire-to-water* efficiency which may be used to calculate pump power consumption [42].

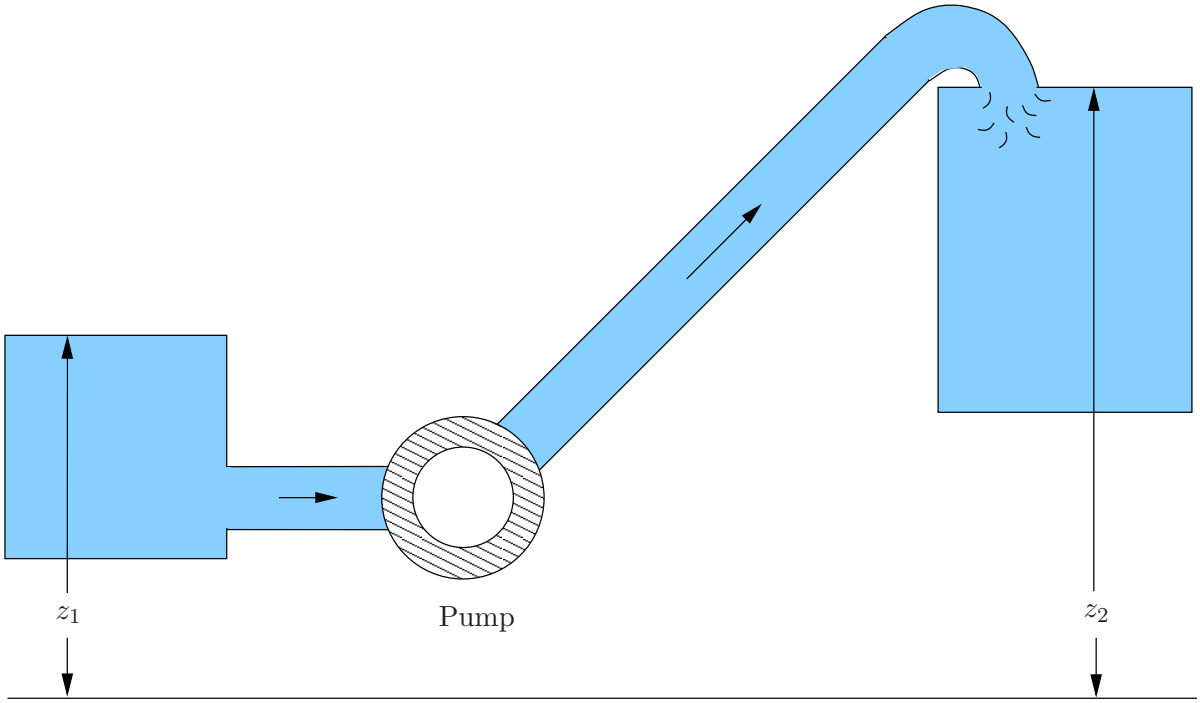


Figure 2.8*: A simple two-reservoir pump system.*

Series Pipe Systems

Consider the series pipe system in Figure 2.9(a). Due to the law of continuity, discharge is equal in each pipe, yielding

$$q = q_1 = q_2 = q_3.$$

The energy equations provide the total head loss as the sum of the head losses in the individual pipes [170],

$$h_{\mathrm{L}} = h_{\mathrm{L}_1} + h_{\mathrm{L}_2} + h_{\mathrm{L}_3} = \sum_i h_{\mathrm{L}_i}.$$

Using the notion of an equivalent pipe and an equivalent loss coefficient K_{e}, the total head loss may be expressed as

$$h_{\mathrm{L}} = K_{\mathrm{e}} q^{\eta_{\mathrm{e}}} = \sum_i K_i q^{\eta_i}.$$

If η_i is the same for each pipe, then $K_{\mathrm{e}} = \sum K_i$ [170].

Parallel Pipe Systems

Consider the parallel pipe system in Figure 2.9(b). By continuity the total discharge is equal to the sum of the discharge from each pipe, giving

$$q = q_1 + q_2 + q_3. \tag{2.23}$$

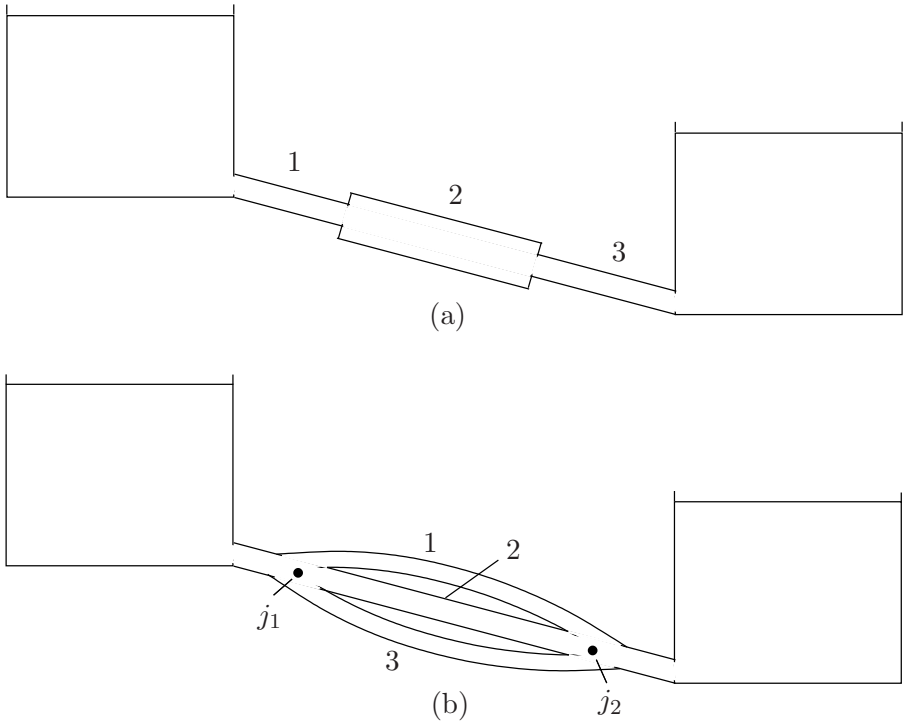


Figure 2.9*: Pipes in (a) series and (b) parallel.*

No matter which pipe one considers, the pressure and elevation difference between the two junction points (j_1 and j_2) are the same. Since $h_{\mathrm{L}} = (p_1/\gamma + z_1) - (p_2/\gamma + z_2)$, it follows that the head loss, h_{L}, between the two junction points is the same in each of the pipes in the parallel pipe system [42, 170]. Therefore $h_{\mathrm{L}} = h_{\mathrm{L}_1} = h_{\mathrm{L}_2} = h_{\mathrm{L}_3}$. Considering pipes 1 and 2, this gives the Darcy-Weisbach formulation as

$$f_1 \frac{L_1}{D_1} \frac{v_1^2}{2g} = f_2 \frac{L_2}{D_2} \frac{v_2^2}{2g},$$

which may be rearranged to yield

$$\frac{v_1}{v_2} = \sqrt{\frac{f_2}{f_1} \frac{L_2}{L_1} \frac{D_1}{D_2}}.$$

If f_1 and f_2 are known, the division of flow may easily be calculated. However, some trial and error analysis might be required if f_1 and f_2 are in the range where they are functions of the Reynolds number [42]. In terms of an equivalent pipe, substituting $q = (h_{\mathrm{L}}/K_e)^{\frac{1}{\eta_{\mathrm{e}}}}$ into (2.23) yields

$$\left(\frac{h_{\mathrm{L}}}{K_{\mathrm{e}}}\right)^{\frac{1}{\eta_{\mathrm{e}}}} = \sum_i \left(\frac{h_{\mathrm{L}}}{K_i}\right)^{\frac{1}{\eta_i}}.$$

When the exponents η_i are all equal, this may be reduced to

$$\left(\frac{1}{K_{\mathrm{e}}}\right)^{\frac{1}{\eta}} = \sum_i \left(\frac{1}{K_i}\right)^{\frac{1}{\eta}}.$$

Branched Pipe Systems

A branched pipe system with its hydraulic grade lines is shown in Figure 2.10. The flow distribution in branched pipe systems may be determined by applying the continuity and energy equations for pipe flow [42, 170]. The continuity equation is $q_3 = q_1 + q_2$. In the vast majority of municipal hydraulic applications, velocity heads are negligible compared to the piezometric heads, and can safely be omitted from calculations [169]. Note that the reservoirs and the outlet pipe are all exposed to atmospheric pressure. The energy equation for the pipe from reservoir surface 1 to the junction at 4 is therefore

$$z_1 = z_4 + \frac{p_4}{\gamma} + h_{\mathrm{L}_1}.$$

From reservoir 2 to 4, the energy equation is

$$z_2 = z_4 + \frac{p_4}{\gamma} + h_{\mathrm{L}_2},$$

and from junction 4 to the outlet at 3, the equation is

$$z_3 + h_{\mathrm{L}_3} = z_4 + \frac{p_4}{\gamma}.$$

If the pipe sizes, lengths and node elevations are known, one may solve these equations for the pressure head, $\frac{p_4}{\gamma}$, at junction 4.

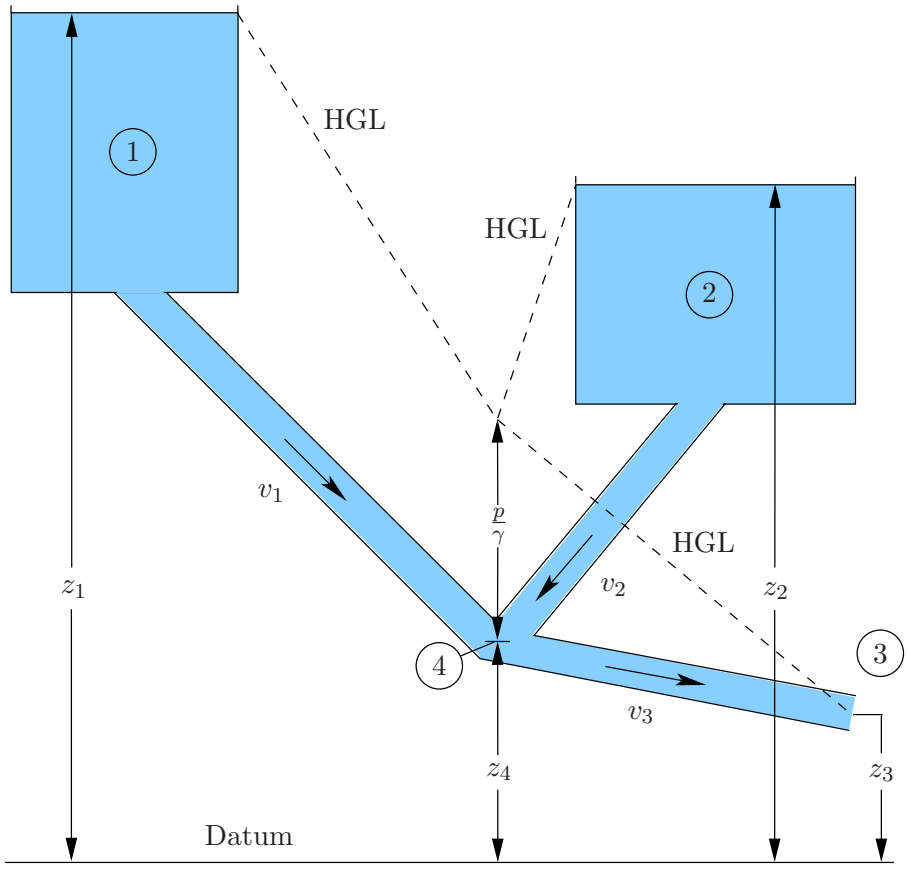


Figure 2.10: *A simple branched pipe system.*

2.1.10 Transient Analysis

Transient events in hydraulic networks occur when there are sudden changes in flow conditions, such as when a valve closes quickly or a pump fails. As mentioned earlier, this may cause large pressure waves (or *water hammer* effects) to rebound through the system at close to the speed of sound, causing much higher pressures than ordinarily found during steady state conditions. This may cause pipes to rupture and other system failures [169, 170].

The underlying reason for these pressure waves is that during a transient event the velocity head is rapidly transformed into pressure head and *vice versa*. During normal hydraulic analysis, the compressibility of water is ignored, but the assumption of compressibility is fundamental in transient analysis. Consider a pipe of length L and radius r with a valve at the downstream end. If the fluid in the pipe is flowing at an average velocity v then the total kinetic energy in the pipe is $E_{\mathrm{k}} = \frac{1}{2}\rho\pi r^2 L v^2$. This quantity may become very substantial as velocity increases. Similarly, pipes may carry large quantities of momentum, consequently requiring large forces to alter flow conditions. Transient analysis is an important component of hydraulic system design which has often been grossly neglected by the engineering community in the past [169].

Consider the simple reservoir system with a pipe and a valve shown in Figure 2.11. Suppose initially the valve is fully open so that water is flowing at an average velocity v from the reservoir, with negligible head loss. Suppose the valve is closed instantly, so that a pressure wave develops which begins to move towards the reservoir at the speed of sound v_c. Now, the water in the pipe between the reservoir and the pressure wave still has the initial velocity v, whilst the water behind the pressure wave has zero velocity. The pressure in the section between the valve and the pressure wave is $p_0 + \Delta p/\gamma$, where p_0 denotes the initial pressure and Δp refers to the increased pressure. Once the pressure wave reaches the reservoir, the pressure imbalance causes the water to flow from the pipe back into the reservoir at velocity v. This causes a new pressure wave that travels in the direction of the valve. These waves continue to rebound until friction forces dampen out the pressure waves. The *acoustic velocity* is the speed of sound in a fluid

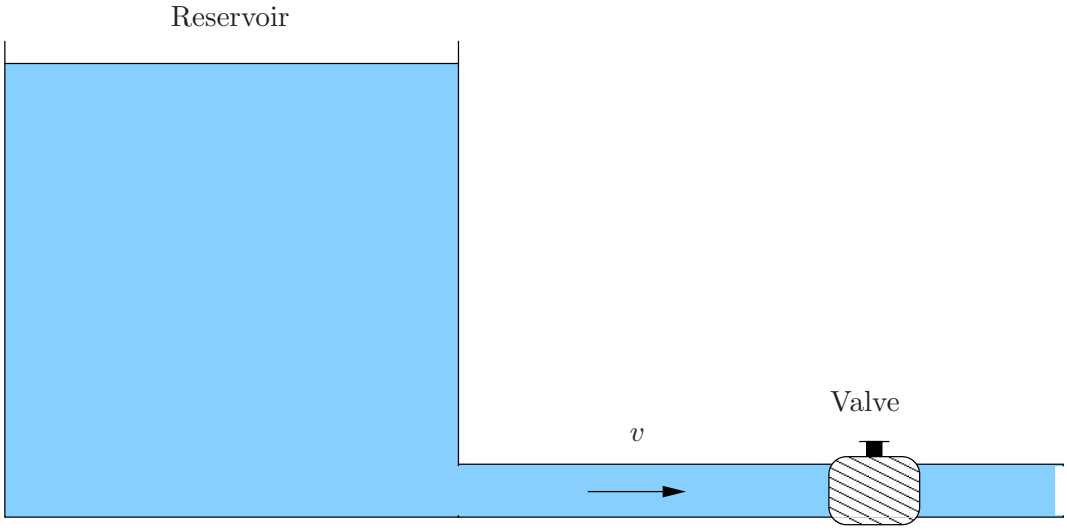


Figure 2.11: *A reservoir system with a distribution pipe and a valve.*

medium. To determine the wave speed a (*celerity*), acoustic velocity in the fluid medium a_0 is modified by the pipe wall elasticity, depending on the elastic properties of the wall material

and the relative wall thickness [169]. The expression for wave speed is

$$a = \frac{\sqrt{E_\mathrm{v}/\rho}}{\sqrt{1+\frac{DE_\mathrm{v}}{eE_\mathrm{m}}}} = \frac{a_0}{\sqrt{1+\frac{DE_\mathrm{v}}{eE_\mathrm{m}}}},$$

where E_v is the *bulk modulus of elasticity* of the fluid, ρ is the density of the fluid, E_m is the *elastic modulus of the pipe wall*, D is the internal diameter of the pipe, and e is the wall thickness. In a very rigid pipe, this reduces to $a = a_0 = \sqrt{(E_\mathrm{v}/\rho)}$.

The *Joukowsky equation* relates water hammer pressure change with velocity change and acoustic velocity, but neglects the effect of head loss and is only valid for sudden changes [169]. It is expressed as

$$\Delta p = -\rho a \Delta v.$$

In terms of head, the *Joukowsky head rise* may be expressed as

$$\Delta h = \frac{\Delta p}{\gamma} = -\frac{\rho a \Delta v}{\rho g} = \frac{a v_0}{g}.$$

2.2 Hydraulic Systems Theory

The mathematical representation and computer simulation of hydraulic networks are the topics of this section. This refers to the problem of computing the hydraulic characteristics of a water system for a given set of initial conditions and demands on the system, commonly known as '*network balancing*' or '*hydraulic network simulation*'. Four different mathematical models are presented in this section, as well as brief discussions of the model implementation and the software used in this dissertation, namely EPANET 2.

2.2.1 Hydraulic Network Simulation

As mentioned earlier in this Chapter, the system pressures and flows are normally calculated for *steady-state* flow conditions. These system-wide characteristics must be calculated for various network configurations under various loading conditions over time. In an *extended period simulation* one considers time variation in the tank elevations and demand loads in discrete time units, effectively performing a steady state simulation for each time unit with the boundary conditions for each consecutive period determined by the previous one [169, 170]. A more advanced simulation technique is *dynamic modelling*, which considers unsteady flow conditions incorporating transient analysis, and thus closer reflects reality. Due to the complexity of this approach it is not yet in wide use. A simplified version of this is *gradually varied conditions* in which it is assumed that a pipe is rigid and changes in flow occur instantly along a pipe. Here, velocity is uniform along a pipe, but may change with time [169].

The calculation of hydraulic state in a network under steady state requires that two conditions be satisfied. The first condition is that continuity must be satisfied for all n junctions in the network. That is, the flow into any junction of the network must equal the flow out of that junction. If $\mathcal{Q}_{i,\mathrm{in}}$ and $\mathcal{Q}_{i,\mathrm{out}}$ are the sets of inflows and outflows respectively for node i, the continuity condition provides the set of equations

$$\sum_{j\in\mathcal{Q}_{i,\mathrm{in}}} q_j - \sum_{k\in\mathcal{Q}_{i,\mathrm{out}}} q_k = \sum d_i, \qquad i = 1,\ldots,n, \tag{2.24}$$

where d_i is the external supply or demand at node i [42].

The second condition is that head loss between any two junctions must be the same, regardless of which path in a series of pipes is taken to arrive at one junction point from another. This requirement results because pressure must be continuous throughout the network, and cannot have two different values at a given point. This condition also requires that the sum of the head losses around a loop must equal zero. In this situation, the sign (positive or negative) for the head loss in a given pipe is bestowed by considering whether the flow is clockwise or counter-clockwise with respect to a loop [42]. A loop is defined as a *primary loop* if it contains no sub-loops (see Figure 2.12 for an example) [169]. For the n_{L} primary loops in a network, there exists a set of conservation of energy equations,

$$\sum_{i,j \in \mathcal{I}_{\mathrm{N}_l}} h_{\mathrm{L}_{i,j}} - \sum_{k \in \mathcal{I}_{\mathrm{P}_l}} h_{\mathrm{p},k} = 0, \qquad l = 1, \ldots, n_{\mathrm{L}}, \tag{2.25}$$

where $\mathcal{I}_{\mathrm{N}_l}$ is the set of nodes in loop l, $h_{\mathrm{L}_{i,j}}$ is the head loss in the pipe connecting nodes i and j, $\mathcal{I}_{\mathrm{P}_l}$ is the set of pumps in loop l, and $h_{\mathrm{p},k}$ is the head produced by the k^{th} pump.

Energy must also be conserved between *fixed-grade nodes* (FGN), which are points of known constant elevation plus pressure head, such as two reservoirs in a network. If the known pressure head difference between two fixed grade nodes is ΔE_{FGN}, then given n_{f} fixed grade nodes in a network, one obtains a set of $n_{\mathrm{f}} - 1$ equations,

$$\sum_{i,j \in \mathcal{I}_{\mathrm{N}_l}} h_{\mathrm{L}_{i,j}} - \sum_{k \in \mathcal{I}_{\mathrm{P}_l}} h_{\mathrm{p},k} = \Delta E_{\mathrm{FGN}_l}, \qquad l = 1, \ldots, n_{\mathrm{f}} - 1, \tag{2.26}$$

where the terms have a similar definition to those in (2.25), except that it applies to path l between two fixed grade nodes instead of to a loop. Fixed grade node pathways are also known as *pseudo-loops* [169].

In total there are now $N_{\mathrm{C}} = n + n_{\mathrm{L}} + n_f - 1$ equations representing the constraints on the water network. Numerous methods exist for solving this system of equations. The original numerical method of solution for hydraulic networks is the *Hardy Cross method* [41].

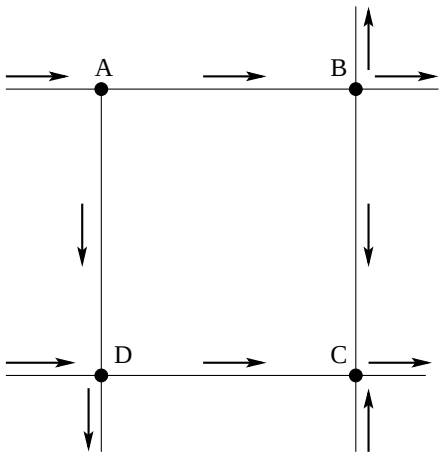


Figure 2.12: *A primary loop subsection in a pipe network.*

2.2.2 Hardy Cross Method

The Hardy Cross method is an adaptation of *Newton's method* (described in Appendix B) [170]. An initial estimate of the flow conditions throughout the network must be established

so that the loads at the various nodes are satisfied. The first estimate of flow conditions will typically not satisfy the head loss condition, therefore corrections are applied [169, 170]. For each loop of the network, a discharge correction is applied to yield a zero net head loss around the loop [42]. Consider the example of an isolated loop in Figure 2.12, noting that no pumps are present and that the arrows indicate the direction of flow. In this loop the loss of head in a clockwise sense is given by $\sum h_{\mathrm{L_c}} = h_{\mathrm{L_{AB}}} + h_{\mathrm{L_{BC}}} = \sum_{\mathrm{c}} K q_{\mathrm{c}}^{\eta}$, and in a counterclockwise sense by, $\sum h_{\mathrm{L_{cc}}} = \sum_{\mathrm{cc}} K q_{\mathrm{cc}}^{\eta}$. In a valid solution, the counterclockwise and clockwise head losses must be equal, that is

$$\begin{aligned} \sum h_{\mathrm{L_c}} &= \sum h_{\mathrm{L_{cc}}}, \\ \sum_{\mathrm{c}} K q_{\mathrm{c}}^{\eta} &= \sum_{\mathrm{cc}} K q_{\mathrm{cc}}^{\eta}. \end{aligned}$$

A discharge correction Δq is applied to satisfy the head loss requirement. If the clockwise head loss is greater than the counterclockwise head loss, then Δq is applied in the counterclockwise sense, and *vice versa*. That is, Δq is subtracted from the clockwise flows and added to the counterclockwise flows [42]. This yields

$$\sum_{\mathrm{c}} K(q_{\mathrm{c}} - \Delta q)^{\eta} = \sum_{\mathrm{cc}} K(q_{\mathrm{cc}} + \Delta q)^{\eta}. \tag{2.27}$$

The summation on either side of (2.27) may be expanded, and all but the two most significant terms excluded, yielding

$$\sum_{\mathrm{c}} K(q_{\mathrm{c}}^{\eta} - \eta q_{\mathrm{c}}^{\eta-1} \Delta q) = \sum_{\mathrm{cc}} K(q_{\mathrm{cc}}^{\eta} + \eta q_{\mathrm{cc}}^{\eta-1} \Delta q).$$

One may now solve for Δq obtaining

$$\Delta q = \frac{\sum K(q_{\mathrm{c}}^{\eta} - q_{\mathrm{cc}}^{\eta})}{\sum \eta K q_{\mathrm{c}}^{\eta-1} + \sum \eta K q_{\mathrm{cc}}^{\eta-1}}. \tag{2.28}$$

If Δq in (2.28) is positive, the correction is applied in a counterclockwise sense. A different Δq is calculated for each loop in the network [42]. Any pipe may be part of at most two primary loops. Consequently certain pipes may have more than one correction applied [169]. Therefore, the first set of flow corrections will usually not yield the desired final result and the solution can only be approached by successive approximations until the corrections are negligible. Experience has shown that multiplying the Δq obtained in (2.28) by 0.6 will result in a faster convergence [42]. To understand why this method may be reduced to Newton's method, consider a notation where the signs of the flows (clockwise or counterclockwise) are implicit in the variable $q_{i,k}$, where this is the ith pipe flow in the kth iteration, and the head loss may be expressed as a function $F(q_k)$. Equation (2.28) may now be rewritten as

$$\Delta q_{k+1} = -\frac{\sum_{i \in \mathcal{I}_L} K_i q_i^{\eta}}{\eta \sum K q_i^{\eta-1}} = \frac{-F(q_k)}{\partial F / \partial q(q_k)},$$

which is exactly equivalent to Newton's Method (see Appendix B).

Although useful for conceptualizing the simulation process, the Hardy Cross method is computationally more intensive than competing methods [169], and is hence not used in this dissertation.

2.2.3 Linear Theory Method

The *linear theory method* solves N_C non-linear energy equations for the unknown pipe flows, q, using an iterative procedure. This method was developed by Wood and Charles [29] in 1972. The energy head loss equations are linearized about $q_{i,k+1}$, where the subscript $k+1$ denotes the current iteration. Here the previous iterations $q_{i,k}$ are taken as known values [169]. For an isolated loop, without the presence of pumps, equations (2.24), (2.25), and (2.26) yield

$$\sum_{i \in \mathcal{P}_j} q_{i,k+1} = \sum d_i,$$

$$\sum_{i,j \in \mathcal{I}_L} K_i q_{i,k}^{\eta-1} q_{i,k+1} = 0,$$

and

$$\sum_{i,j \in \mathcal{I}_L} K_i q_{i,k}^{\eta-1} q_{i,k+1} = \Delta E_{\text{FGN}}$$

respectively. Here, $\mathcal{P}_j$ is the set of pipes connected to junction j and $\mathcal{I}_L$ is the set of pipes in loop or pseudo-loop, L. These equations form a set of linear equations which may be solved for values of $q_{i,k+1}$. The absolute values of the difference in successive flow estimates are compared to a convergence criterion. The process is repeated until these differences become insignificant. Owing to oscillations around the final solution, Woods and Charles recommend using the mean of the previous two iterations as an estimate for the next iteration [169].

2.2.4 Gradient Algorithm

The gradient algorithm does not use loop equations, but rather employs *pipe equations* in which q and h are solved for simultaneously. The pipe equations are formed by writing the conservation of energy equation for the system components in terms of the nodal heads. The energy equation for a pipe is expressed as

$$h_1 - h_2 = K q^{\eta}, \tag{2.29}$$

where h_1 and h_2 are the nodal heads at the upstream and downstream ends of the pipe. Similarly, using a quadratic approximation for the pump head, the energy equation for a pump is

$$h_2 - h_1 = a q^2 + b q + c. \tag{2.30}$$

These equations may be combined with the junction continuity equations (2.24) to form $n + n_s$ equations with unknowns of nodal heads and pipe flows, where n_s is the number of components (pipes and pumps). The gradient algorithm proceeds by linearizing the component energy equations using the previous flow estimates. For example, the equation for pipes becomes

$$K q_k^{\eta-1} q_{k+1} + h_1 - h_2 = 0.$$

The linearized component equations may be expressed in matrix form as

$$\mathbf{A}_{12}\boldsymbol{h} + \mathbf{A}_{11}\boldsymbol{q} + \mathbf{A}_{10}\boldsymbol{h}_0 = \mathbf{0} \tag{2.31}$$

and the linearized continuity equations as

$$\mathbf{A}_{21}\boldsymbol{q} - \boldsymbol{d} = \mathbf{0}, \tag{2.32}$$

where $\mathbf{A}_{12} = \mathbf{A}_{21}^T$ $(n \times n_\mathrm{s})$ is the incidence matrix of zeros and ones corresponding row-wise to the nodes which are connected to each component, and $\mathbf{A}_{10}$ $(n \times n_\mathrm{R})$ identifies all the fixed grade nodes. $\mathbf{A}_{11}$ $(n_\mathrm{s} \times n_\mathrm{s})$ is a diagonal matrix containing the linearization coefficients, $\left|Kq_k^{\eta-1}\right|$ [169]. This produces the overall system of linear equations,

$$\begin{bmatrix} \mathbf{A}_{11} & \mathbf{A}_{12} \\ \mathbf{A}_{21} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \boldsymbol{q} \\ \boldsymbol{h} \end{bmatrix} = \begin{bmatrix} -\mathbf{A}_{10}\boldsymbol{h}_0 \\ \boldsymbol{d} \end{bmatrix}. \tag{2.33}$$

Differentiating this system yields

$$\begin{bmatrix} \mathbf{D}\mathbf{A}_{11} & \mathbf{A}_{12} \\ \mathbf{A}_{21} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathrm{d}\boldsymbol{q} \\ \mathrm{d}\boldsymbol{h} \end{bmatrix} = \begin{bmatrix} \mathrm{d}\mathbf{E} \\ \mathrm{d}\boldsymbol{d} \end{bmatrix}, \tag{2.34}$$

where $\mathrm{d}\mathbf{E}$ and $\mathrm{d}\boldsymbol{d}$ are the residuals of (2.29), (2.30) and (2.24) evaluated at the current solution $\boldsymbol{q}_k$ and $\boldsymbol{h}_k$. $\mathbf{D}\mathbf{A}_{11}$ is a diagonal matrix of the exponents of the pipe equations. The system (2.34) is a set of linear equations in $\mathrm{d}\boldsymbol{q}$ and $\mathrm{d}\boldsymbol{h}$. This may be solved and $\boldsymbol{q}, \boldsymbol{h}$ updated using

$$\boldsymbol{q}_{k+1} = \boldsymbol{q}_k + \mathrm{d}\boldsymbol{q} \tag{2.35}$$

and

$$\boldsymbol{h}_{k+1} = \boldsymbol{h}_k + \mathrm{d}\boldsymbol{h}. \tag{2.36}$$

Iterations continue until convergence is achieved, which is determined by means of $\mathrm{d}\mathbf{E}$ and $\mathrm{d}\boldsymbol{d}$. Todini and Pilati developed an alternative efficient recursive scheme which improves performance. Although the gradient algorithm requires a larger set of equations to be solved than the Hardy Cross or Linear Theory methods, it has been shown to be robust and computationally efficient [169, 192].

2.2.5 Pressure Driven Analysis

While most water simulation models currently in use are based on demand driven analysis (DDA), there has been a campaign to shift towards pressure driven analysis (PDA), which is physically more accurate [100, 228]. DDA fails to take properly into account the relationship between pressure and demand, since demand will only be satisfied provided there is sufficient pressure. Models that assume demand satisfaction and calculate the resulting pressures may produce incongruous results, such as demand being satisfied at nodes with negative pressures, especially when analysing extreme demand conditions and fireflow scenarios [229]. Furthermore, the use of PDA allows one to quantify hydraulic performance using the *degree of demand satisfaction*, and to conduct leakage analysis by means of leakage models which require accurate pressure inputs.

The simplest form of PDA is the use of the standard *emitter* equation to calculate discharge at an orifice. This expresses the flow rate as

$$q = ch^{\mathcal{A}}, \tag{2.37}$$

where c is the constant *emitter / leakage coefficient*, and where $\mathcal{A}$ is the constant *emitter exponent* for the outlet. The emitter coefficient incorporates various flow-independent factors such as the orifice diameter, the secondary loss coefficient and the contraction of the flow path downstream of the orifice. The emitter exponent embodies the sensitivity of the flow rate to pressure. This expression is often used to simulate simple pressure-dependent emitters,

such as sprayers and leaks [204], and has even been used for pressure-dependent discharge at demand nodes [31]. In conventional theory, the rate of flow through a fixed diameter hole in a pipe is proportional to the square root of the pressure, *i.e.* $\mathcal{A} = 0.5$ [239]. However, this method is somewhat simplistic and may not be well suited to general usage for calculating pressure-dependent demand satisfaction or leakage analysis where emitter exponents may vary extensively [228, 239].

In 2003 Todini [228] proposed a more realistic approach to the extended period analysis of WDS incorporating PDA. This yields the more general form of (2.33),

$$\begin{bmatrix} \mathbf{A}_{11} & \mathbf{A}_{12} \\ \mathbf{A}_{21} & \mathbf{A}_{22} \end{bmatrix} \begin{bmatrix} \boldsymbol{q} \\ \boldsymbol{h} \end{bmatrix} = \begin{bmatrix} -\mathbf{A}_{10}\boldsymbol{h}_0 \\ \boldsymbol{d} \end{bmatrix}, \tag{2.38}$$

where $\mathbf{A}_{22}$ is a diagonal $n \times n$ matrix whose elements are either zero if the demand of the relevant node is not head-driven, or nonlinear functions of the pressure at the node. Many formulations exist for such a function, such as the formula

$$\mathbf{A}_{22}(i,i) = \begin{cases} 0 & h_i < z_i \\ \frac{q_i (h_i - z_i)^{-1/2}}{h_i (h_i^* - z_i)^{1/2}} & z_i \leq h_i < h_i^* \\ \frac{q_i}{h_i} & h_i \geq h_i^* \end{cases} \tag{2.39}$$

by Aoki [11], where z_i and h_i^* is the elevation and desired minimum head respectively at node i [228]. Taking the derivative of (2.38) yields

$$\begin{bmatrix} \mathbf{DA}_{11} & \mathbf{A}_{12} \\ \mathbf{A}_{21} & \mathbf{DA}_{22} \end{bmatrix} \begin{bmatrix} \mathrm{d}\boldsymbol{q} \\ \mathrm{d}\boldsymbol{h} \end{bmatrix} = \begin{bmatrix} \mathrm{d}\mathbf{E} \\ \mathrm{d}\boldsymbol{d} \end{bmatrix}, \tag{2.40}$$

where $\mathbf{DA}_{11}(k,k) = \eta K q_k^{\eta - 1}$ and $\mathbf{DA}_{22}$ is a diagonal matrix derived from (2.39) such that

$$\mathbf{DA}_{22}(i,i) = \begin{cases} 0 & h_i < z_i \\ q_i \, \frac{(h_i - z_i)^{1/2}}{2(h_i^* - z_i)^{1/2}} & z_i \leq h_i < h_i^* \\ 0 & h_i \geq h_i^*. \end{cases} \tag{2.41}$$

This system of equations may be solved in an iterative manner, such as in the Gradient Algorithm [228].

2.2.6 Comparison of Network Simulation Methods

In addition to the steady-state solution methods already discussed, there is the popular *Newton-Raphson* method [169, 170]. The Newton-Raphson method converges more quickly than the linear theory method for small systems, but may converge very slowly for large networks. It may also suffer from convergence problems if poor initial conditions are selected [29]. The Linear Theory algorithm is reportedly the best algorithm for the loop equation formulation, does not require initialization of flows, and, according to Woods and Charles [29, 169], always converges rapidly. In 1985, Holloway [124] conducted a comparative study of the Newton-Raphson and Linear theory methods on a 200 pipe system. Both methods converged in 7 to 9 iterations, with the Newton Raphson method requiring the least amount of computation time. In 1987, Salago, Todini and O'Connell [206] compared these two methods to the Gradient algorithm on systems under varying demand loads, and found that the Gradient algorithm outperformed

the others in each scenario. Of the classical methods, the gradient algorithm is reportedly the most computationally efficient, and is the method employed in both the EPANET 2 simulation package [170] and in Watercad^{TM}, a package produced by Haestad Methods [251].

One recent method for extended period simulation, published in 2006 by Van Zyl *et al.* [241], is the *Explicit Integration* (EI) method, which attempts to decouple a network into constituent simple base systems and solve each individually. This is done by integrating their linearized dynamic tank equations explicitly. The results are then used to estimate the dynamic behavior of the full WDS. The accuracy and efficiency of the EI method was shown to be superior to that of the *Euler Integration* method [261] for two sample networks [241]. Another modern trend is to include transient modelling in extended period simulation [169].

2.2.7 Model Calibration

Calibration is the process of adjusting the model parameters so that the simulated results accurately reflect the observed field data for an existing physical system. Calibration is extremely important in WDS simulation. Two of the major difficulties in calibration are estimating the demand loads and pipe-carrying capacities accurately, since these variables may introduce serious errors in calculations. The data for calibration is usually derived from fire-flow pressure measurements, which involves measuring pressures and flow in isolated pipes or pipe sections, although the modern trend is to install automated flow meters with the ability to transmit live data. These measured values are used to adjust pipe friction factors, demand loads and other parameters in a simulation until a good fit is obtained [103, 170].

2.2.8 Model Implementation

In 1988, Clark *et al.* [34] formulated a framework for hydraulic model implementation. This is taken directly from [170] as follows:

1. *Model Selection*: Definition of model requirements and selection of model that fits desired requirements.

2. *Network representation*: Representation of the distribution system components in the model.

3. *Calibration*: Adjustment of model parameters so that predicted results adequately reflect field data.

4. *Verification*: Independent comparison of model and field results to verify the adequacy of the model representation.

5. *Problem definition*: Definition of the specific design or operational problem to be studied and incorporation of the situation into the model.

6. *Model application*: Use of the model to study the specific problem / situation.

7. *Analysis / display of results*: Following the application of the model, the results should be displayed and analyzed to determine the rationality of the results and to translate the results into a solution to the problem.

In this dissertation, only new system designs and existing benchmark systems are examined, in which cases the model parameters are known in advance. Therefore, the calibration and verification steps are not necessary in the application of the optimisation model.

2.2.9 EPANET

Many computer models exist to solve hydraulic network simulation equations. EPANET is one of the more commonly used computer models worldwide. It has the ability to simulate steady-state conditions, perform extended period simulation, and analyze water quality. Other computer models include PIPE2000 [154], Hytran [127], Infowater [179], and WaterCAD [20], all of which are commercial products which additionally support transient analysis. EPANET 2 was used extensively in this dissertation. It was developed by the U.S. Environmental Protection Agency for public domain use [203]. EPANET 2 uses the Gradient Algorithm described previously. A detailed discussion on the use of the software, as well as the underlying theory, is available in [204].

2.3 Chapter Summary

In this chapter the basic theory of WDS hydraulics was presented, in fulfilment of Dissertation Objective 1 in §1.3. This should provide the reader with a foundation for understanding the concepts and terms used in the remainder of this dissertation.

For a comprehensive treatment of fluid mechanics the reader is referred to Crowe and Roberson [43]. For a detailed reference on WDSs the reader may consult Mays [169]. Finally, an excellent reference on WDS modelling is Walski *et al.* [251].

Chapter 3

Single-objective WDS Design Optimisation

This chapter contains a description of the water distribution system (WDS) design optimisation problem, focussing on the single-objective case of constrained least-cost optimisation. It includes a mathematical formulation of the problem, a concise literature survey of research on the problem, numerous WDS design considerations, and a discussion of optimisation methods and metaheuristics which have been applied in an attempt to solve the problem.

3.1 Introduction

Water distribution system design optimisation is a challenging problem [83]. It may involve the design of an original water network, or the expansion / rehabilitation of an existing network. It is desirable that the system be economical to install and maintain, and that it satisfies certain performance criteria, including reliability, water quality, capacity, and pressure quantities [251]. The computational complexity of solving the problem of optimal WDS design problem is extremely high. This problem belongs to the class of NP-hard problems which are intractable when using exact solution methods for large problem instances. Traditional optimisation techniques (such as linear programming) have only been applied to WDS optimisation with numerous simplifying assumptions [83].

Furthermore, WDS design requires significant engineering knowledge and sound judgment. A systematic design optimisation procedure should be followed. A schematic overview of the optimisation *model application* process is provided in Figure 3.1 (a schematic adapted from [251]). It demonstrates that the first requirement is a calibrated model of the WDS which can be used to predict the behaviour of the system under different conditions. This model may not necessarily require calibration with real test data, depending on whether it represents a completely new system or an existing one requiring augmentation. A new system would use cited component characteristic values, such as the quoted internal diameters and roughness characteristics of pipes, as provided by the manufacturer. The *problem statement* in this context is the design optimisation of WDSs (least-cost optimisation is assumed for the moment). If the current model results are not accurate enough to predict real observations, then they cannot be used in decision making, and additional measurements of the real system may be required to recalibrate the model. The technique used to formulate alternative solutions is the actual *optimisation method*, with which this dissertation is primarily concerned. This may constitute a form of non-linear

programming, such as the *generalized reduced gradient method*, an *adaptive search* algorithm, such as the *genetic algorithm*, or one of numerous alternative metaheuristics in the operational research literature. The current solution(s) identified by the optimisation technique must then be tested. This includes conducting a hydraulic simulation in order to calculate the system-wide flows and pressures, and to determine whether a particular solution is feasible in terms of its constraints. Depending on the optimisation strategy used, the hydraulic simulation may be the most computationally intensive subroutine, yet the optimisation method has a direct influence over how many such simulations must be performed. The *cost analysis* is an evaluation of one or more objective function values. These values are then used by the optimisation algorithm to determine how to proceed with its search through solution space. Once a sufficient quantity of quality solutions have been generated, a decision may be made with respect to which solution to implement [251]. Initial optimisation methods, which aimed at determining only a minimum

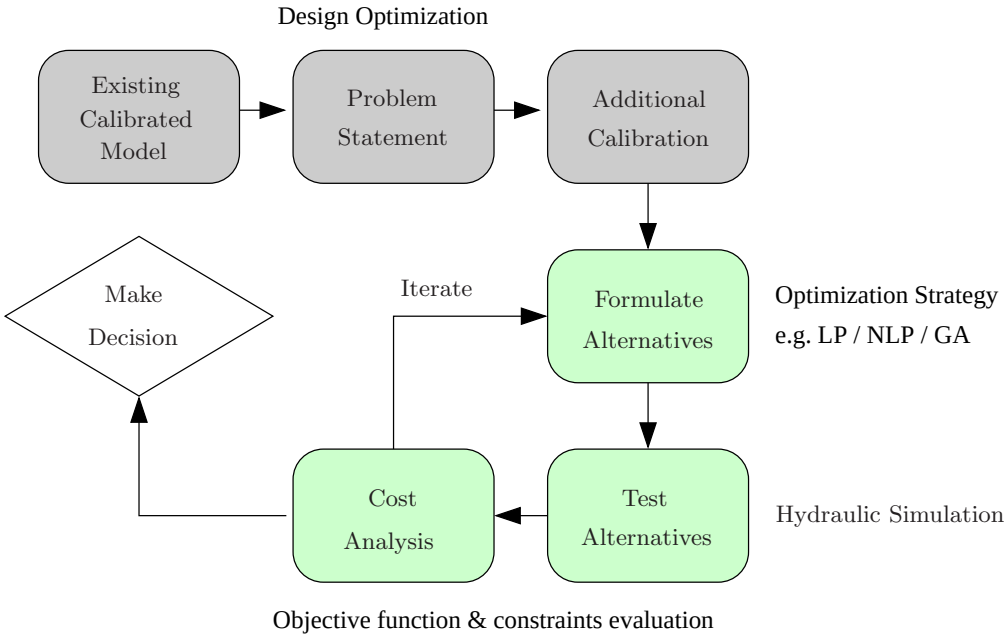


Figure 3.1*: Overview of optimisation model application [251].*

cost design solution, have been superseded by multi-objective optimisation techniques which generate a Pareto-optimal set[1] of alternative solutions. A discussion of objectives in addition to capital costs is included in Chapter 4, while an in-depth discussion of multi-objective optimisation and the issues relating to its application in the context of WDS design optimisation is included in Chapter 5. Additionally, traditional optimisation techniques, such as linear and non-linear programming, which tend to oversimplify the problem or become trapped in local optima (WDS design problems are typically highly non-linear), have given way to more successful adaptive search techniques, such as genetic algorithms and simulated annealing [251]. In particular, evolutionary algorithms are more suited to the multi-objective paradigm, in the sense that they may be able to find multiple members of a Pareto-optimal set in a single run. Optimisation should not be viewed as an automated process by which a single optimal solution is identified. It should provide alternative solutions offering a range of costs and benefits, fulfilling the role of a decision support tool for design engineers [251].

[1]The members of a Pareto-optimal solution set represent a trade-off between different objectives. Moving from one solution to another will result in the degradation of one objective function value, but the improvement of another. In the context of WDS design, one might have a trade-off between reliability and cost, and for each reliability objective value a solution is found which achieves that reliability at the minimum cost.

The typical formulation of the optimisation problem is a minimisation of costs subject to hydraulic feasibility, demand satisfaction and pressure constraints (as described in some detail in this chapter), but additional considerations may include reasonable levels of redundancy and reliability, budgetary constraints, trade-offs between competing objectives such as fire-flow versus water quality, minimizing running costs (pumping costs, pipe replacement costs), increasing component homogeneity, minimizing excavation costs, the uncertainty of future requirements, ensuring spare water storage capacity for emergencies, and ease of maintenance [250, 251].

WDS optimisation methods employed in the literature have included linear and nonlinear programming [9, 223], dynamic programming [212], mixed-integer programming [145], gradient search algorithms [159], enumeration methods [99], genetic algorithms [51, 209, 218, 262], simulated annealing [46], memetic algorithms [83], ant colony optimisation [168, 274], and others [251].

Several benchmark water systems have appeared in the literature for optimisation model testing. The following nine benchmark WDSs are analyzed in this dissertation: the *New York Tunnel* problem (NYTUN), proposed by Schaake and Lai [212] in 1969, the *Two-loop Network* (TLN), introduced by Alperovits and Shamir [9] in 1977, the *Hanoi Network* (HANOI), first presented by Fujiwara and Khang [94] in 1990, the *Two Reservoir Problem* (TRP), proposed by Simpson *et al.* [218] in 1994, the *Exeter Water Network* (EXNET), by the Centre for Water Systems in Exeter, UK [84], and the *Blacksburg* (BLACK), *Fossolo* (FOSS), *Pescara* (PESCA), and *Modena* (MOD) networks presented by Bragalli *et al.* [21] in 2008. In addition, a tenth WDS design is analyzed, namely a recent South African developmental case study named the *R21 Corridor* (R21), supplied by GLS Software [103].

Optimisation for WDS design is currently not yet part of standard engineering practice, although many optimisation tools are already incorporated into certain design programs [251]. Note that, without the comments outside of the flowchart, Figure 3.1 may be used for any manual engineering planning project which excludes an attempt at mathematical optimisation. Other difficulties with the inclusion of optimisation are that the model is a simplified version of reality, the inability of algorithms to fully capture the design process, and general distrust of automated algorithms which replace traditional engineering judgment. However, it is also easy to demonstrate (1) that humans are unable to understand and predict the implications of a design, due both to data uncertainty / variability and the complex non-linear interactions between components, and (2) that the use of simplistic greedy search heuristics fails to find globally optimal designs. Therefore, some level of optimisation is essential to enable effective decision-making. The optimisation model must be broad enough to address the wide range of important criteria for WDS design, yet simple enough to be used by a non-specialist. Any approach should be efficient, holistic, and systematic, in order to enable correct decision making.

3.2 An Overview of Optimisation Methods

Optimisation methods refer to mathematical techniques used to adjust the parameters (or *decision variables*) of a model system automatically, in order to achieve the best possible (optimal) outcomes, as defined by the system objective function(s), and frequently subject to constraints on the model. For example, this may relate to the search for a least-cost design which satisfies a required performance level, limited by an available budget. An optimisation problem is formulated mathematically as the maximisation or minimisation of one or more objective functions of the decision variables, subject to a set of equality and/or inequality constraints. The range of possible solutions is often called the *search space* or *parameter space*. Similarly, the range of

the objective function values to which the search space maps is often called the *objective space*. Solutions are identified as *feasible* when they satisfy the optimisation constraints, and *infeasible* otherwise [251, 259].

Optimisation methods range from analytical techniques, which employ gradients and higher order derivatives of objective functions, to heuristic search methods and advanced metaheuristics, which often mimic natural phenomena. *Exact methods* (or *deterministic methods*) are those which guarantee locating the global optimal solution to a class of problems, provided a sufficient amount of processing time is available. One such method is exhaustive enumeration, whereby every possible configuration of a model is analyzed, an option very rarely available in real world problems, owing to the enormous number of solutions. Other exact methods include *Linear-* and *Dynamic Programming*. Exact methods are often unable to handle discrete variables, and typically too slow to be used in solving large problem instances due to the 'curse of dimensionality'. Numerous *Nonlinear Programming* methods exist for solving nonlinear problems, however, they are normally approximate methods, since they may become trapped at local optima in the search space [259].

Heuristic is derived from the Greek "heuristikein" or "heurisko", which may be translated as "to find out" or "to discover" [189]. Heuristics are basically *rules of thumb*, criteria, or principles for deciding amongst several alternative courses of action, in order to achieve some objective. They represent a trade-off between simplicity and adequate decision-making power. An example of a heuristic might be a greedy rule, such as choosing the alternative which yields the best immediate gain in objective function value. A *heuristic search* might be used to conduct optimisation, although it generally cannot guarantee finding an optimum [176, 259].

Metaheuristics are algorithms that operate at a higher level than heuristics, as indicated by the prefix *meta*, which may be translated from Greek as "beyond, above, at a higher level", and is often used in the sense of "transformation" [189]. A metaheuristic may be defined as a high-level strategy for directing heuristics in the search for optimal solutions in domains where the task is hard [153]. Metaheuristics do not guarantee finding a global optimum, however, they are often able to locate good solutions which may closely approach the global optimum. They are usually considered *global search techniques*, in that they contain mechanisms to escape local optima (such as the ability to move to a worse solution in the short term, in the hope of finding a better solution in another region of the search space). Furthermore, because of their generality, metaheuristics may be used on a broad class of hard problems. Metaheuristics are frequently the only viable option in solving a difficult problem, owing to the large complexity or size of the problem, or because of other elements such as discontinuity, black-box objective functions, high levels of noise, deceptive local optima, and the mixture of discrete and continuous variables. Metaheuristics are usually able to model problems more realistically and are generally less limited in terms of problem type restrictions (such as requirements of linearity or differentiable functions). Metaheuristics are not always the best techniques to apply, given that many problems are solvable optimally, however in the case of WDS design, they prove extremely relevant [176, 251, 259].

Metaheuristics usually contain *adaptive elements*, in that they alter their strategy as the search progresses. One example of this is an initial period of high exploration, followed by a phase where the search is gradually intensified around localized regions in the objective space, usually coinciding with 'valleys' in the objective response surface (assuming a minimisation problem). Certain formalized optimisation methods are too simplistic to be considered metaheuristics, despite containing heuristic elements, such as the *Finbonacci search*, the *Hooke and Jeeves pattern search*, and the *simplex search*. All of these techniques are *non-adaptive*, and use a numerical difference technique to approximate partial derivatives and conduct improving steps.

A metaheuristic might use one of these methods to conduct a local search within the broader framework of a global optimisation strategy. Many metaheuristics mimic natural processes in their search procedure, such as Evolutionary Algorithms (EAs) which are based upon biological evolution, and employ a population of solutions. These types of metaheuristics are sometimes referred to as *heuristics derived from nature* (HDN) [251, 259].

Examples of metaheuristics include Genetic Algorithms (an example of an EA), Simulated Annealing, Tabu Search, and Ant Colony Optimisation. All of these may be considered adaptive. EAs are mentioned most frequently in the WDS literature as showing promise [251]. The advantages of these for application to WDS optimisation include their inherent discrete formulation (as such they easily handle discrete component sizes), their ability to conduct a robust global search, their lack of requirement for partial derivative calculation (as they use only objective function values), and their mechanisms which allow the search to escape local optima [73, 274].

Another optimisation paradigm considered in this dissertation is that of *hyperheuristics*, an emerging search technology that may be defined as a (meta)heuristic which controls the selection and operation of other (meta)heuristics in a dynamic fashion. Hyperheuristics were largely motivated by the goal of raising the level of generality at which optimisation systems can operate [25], and have been shown to produce performance boosts in some sample studies. In this dissertation the *AMALGAM* hyperheuristic of Vrugt and Robinson [244] is analyzed for use in WDS design optimisation. Several variants of the original AMALGAM formulation are also studied.

An optimisation-simulation framework which typically applies in WDSDO is shown in Figure 3.2. The optimisation module generates candidate solutions using some optimisation method, which must be tested for feasibility and hydraulic performance over a range of demand loadings by the simulation module. This module may either comprise an extended period analysis simulator (for a given time-series of demands), or a stochastic simulator (which samples probability distributions in order to generate demands) [141, 251]. Special scenarios such as pipe failures, fire-flows, and zero-flow conditions may also be generated by the simulation module. The hydraulic simulator calculates the pressures and flows throughout the network for a specific instance of demands. This may, for example, be performed by EPANET [203]. Performance information is passed back to the optimisation procedure to interpret objective function values and generate new candidate solutions.

3.3 Least-Cost Optimal Design Problem for WDS

In the classical problem of WDSDO designers focussed mainly on the sub-problem of *pipe network design*, probably because pipes constitute the bulk of the capital cost [157]. This problem is formulated as follows: For a given layout of pipes, and specified demand at the nodes (usually peak demand), and minimum pressure requirements, given that the head loss in a pipe is a function of flow in the pipe, pipe diameter, length and hydraulic properties, find the combination of pipe diameters which produces the minimum cost design subject to the following constraints [159, 218]:

1. Continuity of flow must be satisfied at all junctions (nodes) in the network (including nodal demand quantities).

2. The total head loss around a loop must equal zero, or the head loss along a path between two reservoirs / tanks must equal the surface elevation difference.

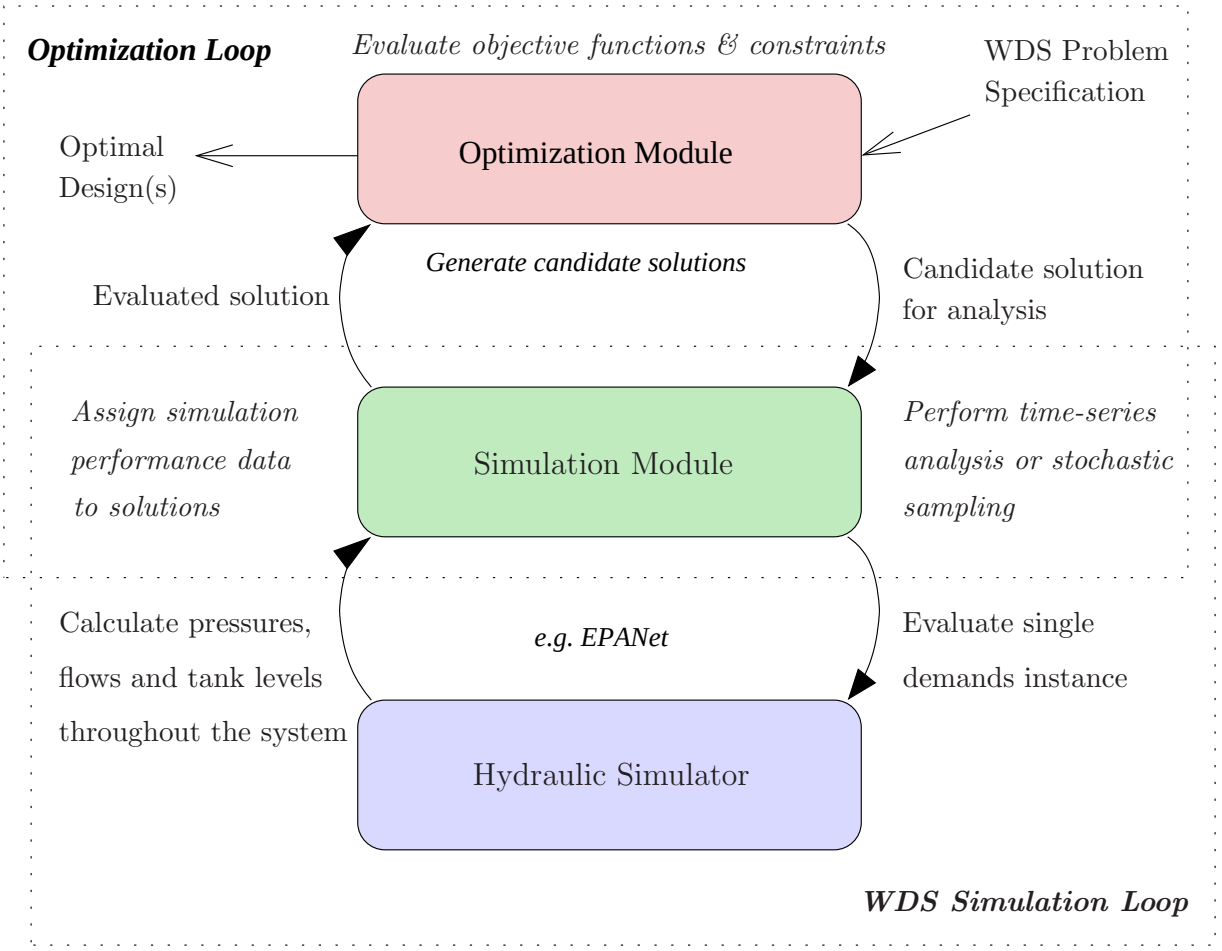


Figure 3.2*: An optimisation-simulation framework for WDS design optimisation (adapted from [141]).*

3. Pressure head at certain nodes must satisfy minimum and maximum limits.
4. Pipe diameters are constrained to a set of discrete sizes.

This basic formulation may be expanded to include the sizing or selection of additional WDS components such as valves, pumps and storage tanks [262], and various component settings, such as pumping schedules and valve settings. There may be existing elements in a system with known (fixed) characteristics [159]. The rehabilitation of an existing network or network subsection may be included by defining additional operations on pipes such as cleaning, replacing or parallelling [251, 274].

It is important to note that this formulation is specified for a single demand load only. In practice, systems experience a wide range of different flow conditions, possibly including a static 'no-flow' flow condition, at which point system pressures are highest in gravity systems due to zero friction losses. The design may also aim to achieve good system performance over the course of an extended period of simulation. It is common that a system is designed to perform well during peak conditions, but may be inefficient during times of low demand [251].

Optimisation constraints may also include reliability and redundancy constraints. An example of one such constraint is the compulsory inclusion of alternate flow paths (loops) to allow for

pipe failure or network maintenance. In practice, loops are a critical feature of distribution systems [159, 262], making them much more robust and reliable; therefore the design process should encourage their inclusion.

A similar concern is spare capacity and flexibility in the face of future uncertain demands. Good indicators of capacity include the flow which can be delivered to numerous nodes during a fire event, or the amount of pressure in excess of some minimum pressure at certain indicator nodes [251]. Excess energy in the system allows it to age gracefully, minimizing the impact of gradually reduced performance, and also accommodates system failure, since more energy will be expended on friction when additional fluid is forced along alternate pathways [227].

Water quality is an additional concern, and is indirectly proportional to system capacity, because of the increased residence time of water in larger pipes and tanks [251].

While there is generally an assumption of steady state flow conditions under a given loading condition, systems are usually tested under multiple loading conditions to ensure reliability of the design [159].

In order to design tanks and pumps, the daily demand patterns have to be analyzed in sequence to consider the tank operations and pumping cost [159]. Weekly or even seasonal patterns may be analyzed. However there is always a large degree of uncertainty. As a minimum consideration, systems should be designed to handle peak loading conditions.

This design optimisation problem is traditionally formulated 'for a given layout'. Owing to the large computational complexity of the WDS design problem, it is typically separated into two phases. The first phase is the determination of an *appropriate layout* for the water network, such that each customer in a region of interest is connected to a water supply, often enforcing the inclusion of loops to allow for uninterrupted supply in times of pipe failure. The second phase is the *sizing of the WDS components*, including pumps, valves, tanks, and pipes. The primary goal in the second phase is to ensure that required quantities of water are supplied at adequate pressures to all consumers in the network [13, 157]. Sizing pipes in order to find a least-cost solution which satisfies a single demand loading case and minimum pressure requirements, produces 'optimal' designs in the form of branched networks. These are problematic, because pipe failure affects an entire branch of the network. Layout design for looped networks is a complex problem which has received relatively scant attention in the literature. To avoid unreliable, branched designs, it is strongly recommended that systems be designed using multiple demand scenarios and explicit reliability objectives, possibly enforcing loops and minimum pipe diameters [251].

Researchers also typically assume a fixed layout, based on physical limitations such as street right of ways, private easements and topography [4]. The design of the pipe layout can have an enormous impact on the overall cost of the system, both because of differing frictional losses and different site excavation costs [251], and multiple feasible layouts may exist which provide improved performance. In reality there is a tight coupling between the layout and component sizing problems, so that it may actually be preferable to combine the phases and design layout and pipe sizes simultaneously. It may therefore be wise to facilitate some level of layout design in the optimisation process.

As mentioned in the previous chapter, each potential network to be tested in the solution space must be hydraulically balanced. This refers to the problem of calculating the hydraulic properties of the network under steady state flow. It has become common practice to use a separate hydraulic simulation package to perform this balancing, so that constraints 1 and 2 above are considered implicitly by the simulation software. The optimisation routine would

then iteratively call the simulation package to calculate the hydraulics for each member of the current solution set, and use the results to make decisions on how to derive a new solution set [211].

Traditional hydraulic simulation uses DDA, whereby the mathematical model enforces demand satisfaction throughout the WDS, and calculates the resultant flows and pressures. Although this technique should be correct within the range of ordinary operation, extreme loads on the system may produce physically impossible situations, such as demand being satisfied at a node with negative pressures. However, the enforcement of pressure constraints should allow the model to operate within its stable range. The alternative to DDA is the more physically accurate PDA, which is currently not in widespread use, although PDA simulation packages are available [226].

3.4 Formulation of the Least-Cost WDS Design Problem

In this section, the mathematical formulation of the least-cost optimal design problem is stated in a very general sense. A comprehensive optimisation of WDSs is formulated by including pipe sizing, rehabilitation, tank, pump and valve design for the entire system under steady state and extended period simulations [251]. This formulation is adapted to include multiple objectives in Chapter 5, and specialized for particular benchmark systems in Chapters 6–9.

Given a water network comprising n nodes and $\mathfrak{l}$ sizable components (pipes, valves, pumps and tanks), the general least-cost optimisation problem may be stated mathematically in terms of the various design variables $\boldsymbol{x}$, nodal demands $\boldsymbol{d}$, and nodal pressure heads $\boldsymbol{h}$. Here $\boldsymbol{x}$ is a vector of selected characteristic values (or physical dimensions) for the $\mathfrak{l}$ sizable system components, $\boldsymbol{d}$ is a vector of length n specifying demand flow rates at each node, and $\boldsymbol{h}$ is a vector of length n whose entries are the pressure head values for the n nodes in the system (note that head depends on $\boldsymbol{x}$ and $\boldsymbol{d}$). Since each component may have more than one sizable characteristic, the length of $\boldsymbol{x}$ depends on the model and the network size with $|\boldsymbol{x}| = \kappa \geq \mathfrak{l}$. Here $\boldsymbol{x}$ may include, for example, the diameter of the pipes, the capacity of the pumps, valve type and setting, and tank volume, diameter and base elevation [159]. The objective function and the constraint sets are discussed in the following subsections.

3.4.1 The Objective Function

The objective is to minimize a *cost function* $f(\boldsymbol{h}, \boldsymbol{d}, \boldsymbol{x})$. This cost function may include installation costs, material costs, and the present value of the running costs and/or maintenance costs for a potential system over its entire lifetime. This single objective formulation may be used to accommodate multiple objectives by aggregating their functions and using weighting coefficients. However, this is definitely sub-optimal in terms of both solution quality and time-efficiency, and therefore not recommended. The objective function may be linear or nonlinear, allowing for various types of components to be designed. By excluding existing components from the cost function, the model also allows for expansion of an existing network [159]. For the simple problem of minimizing the cost of an n_{p}-pipe system (without pressure goals), the objective function may be expressed as $f(\boldsymbol{x}) = \sum_{k=1}^{n_{\mathrm{p}}} C(D_k, L_k)$, where $C(D_k, L_k)$ is a cost function of the pipe diameter D_k, and the pipe length L_k of the k-th pipe [83, 159]. For optimisation methods that cannot accommodate constraints explicitly, it is common practice to add a *penalty term* to the cost function, in order to penalize constraint violations (such as deviation

from system pressure requirements) [159]. This technique requires a penalty factor to scale constraint violations to the same magnitude as cost.

3.4.2 Conservation of Flow Constraints

A set of at least n constraints of the form $\boldsymbol{g}(\boldsymbol{q}, \boldsymbol{d}, \boldsymbol{x}) = \boldsymbol{0}$ includes a conservation of flow equation for each of the nodes in the system, incorporating the nodal water demands and the flows for all pipes branching from a node. More precisely in terms of the flows at nodes $i = 1, \ldots, n$ this is

$$g_i(\boldsymbol{q}, \boldsymbol{d}, \boldsymbol{x}) = g_i(\boldsymbol{q}(\boldsymbol{d}, \boldsymbol{x}), \boldsymbol{d}) = \sum_{j \in \mathcal{N}_i} q_{ij} - d_i = 0, \qquad i = 1, \ldots, n,$$

where $\mathcal{N}_i$ is the set of nodes connected to node i, where q_{ij} is the flow in the pipe connecting nodes i and j, and d_i is the demand at node i. Additional constraints may be added for multiple loading conditions. These may be required to be satisfied according to some temporal pattern [251].

3.4.3 Energy Equation Constraints

A system of equations of the form $\boldsymbol{e}(\boldsymbol{h}, \boldsymbol{d}, \boldsymbol{x}) = \boldsymbol{0}$ is required for the primary loops in the system, specifying that energy is conserved around each loop. The same applies for pseudo-loops (paths between fixed-grade (known head) nodes, such as a reservoir or a tank). The loops and pseudo-loops must be identified by examining the network layout. If there are n_{FG} fixed grade nodes, then $n_{\mathrm{FG}} - 1$ pseudo-loop equations are required. For a total of n_{L} primary loops in the system, these equations may be expressed as

$$e_l(\boldsymbol{h}(\boldsymbol{d}, \boldsymbol{x})) = \sum_{i,j \in \mathcal{I}_l} h_{\mathrm{L}_{ij}} - \sum_{k \in \mathcal{P}_l} h_{\mathrm{p}_k} = 0, \qquad l = 1, \ldots, n_{\mathrm{L}},$$

where $\mathcal{I}_l$ is the set of nodes in loop l, $h_{\mathrm{L}_{ij}}$ is the head loss in the pipe connecting nodes i and j, $\mathcal{P}_l$ is the set of pumps in loop l, and h_{p_k} is the head produced by the k-th pump. A similar set of equations apply for the pseudo-loops, except that the right-hand side of the equation is the known head difference. Alternatively, instead of using loop equations conservation of energy may be expressed solely in terms of pipe headloss equations [169]. The conservation of flow constraints and energy equation constraints are nonlinear, and together represent conservation of mass and energy laws pertaining to pressure and flow distribution in the system. These constraints may be satisfied intrinsically by a hydraulic simulation package [159, 251].

3.4.4 Pressure Head Constraints

Upper and lower bounds of the form $\boldsymbol{h}_{\min} \leq \boldsymbol{h}(\boldsymbol{d}, \boldsymbol{x}) \leq \boldsymbol{h}_{\max}$ may be prescribed on pressure heads $\boldsymbol{h}$ at each node [159]. The upper bound may be related to maintaining the structural integrity of the pipes or some maximum pressure a customer is able to handle. It is usually advisable to treat these as soft constraints, or allow for moderate deficits at some of the nodes, since it may not be possible to satisfy them in all parts of the network, making it impossible to find feasible solutions.

3.4.5 Design Constraints

Design constraints of the form $\boldsymbol{j}_{\min}(\boldsymbol{x}) \leq \boldsymbol{j}(\boldsymbol{x}) \leq \boldsymbol{j}_{\max}(\boldsymbol{x})$ on the variables $\boldsymbol{j}(\boldsymbol{x})$ specify physical limitations or characteristic value sets from which components may be selected [159]. These constraints may represent restrictions on discrete variables, such as pipes which come in a variety of commercial diameters, or the rehabilitation of pipes by specifying cleaning, replacing, and duplicating operations for existing pipes. Such maintenance operations change the effective diameters and friction characteristics of the pipes.

3.4.6 General Constraints

A general constraints set of the form $\boldsymbol{w}_{\min}(\boldsymbol{h}, \boldsymbol{q}, \boldsymbol{d}, \boldsymbol{x}) \leq \boldsymbol{w}(\boldsymbol{h}, \boldsymbol{q}, \boldsymbol{d}, \boldsymbol{x}) \leq \boldsymbol{w}_{\max}(\boldsymbol{h}, \boldsymbol{q}, \boldsymbol{d}, \boldsymbol{x})$ may be used to incorporate limits on terms that are functions of both hydraulics and design variables, such as a limitation on the flow velocity in a pipe, and may include budget limitations, a minimum level of some reliability / redundancy measure, minimum turnover times of water in storage, and other operational constraints [251]. The generality of these constraints allows all types of systems (branched, looped, gravity systems, systems with pumps and tanks) to be analyzed at any level of complexity [159, 223, 262]. It is important to note that both the objective function and constraints are typically non-linear in modern formulations of the problem.

3.4.7 Entire Problem Formulation

The WDSDO problem is therefore to

$$\left.\begin{array}{ll} \text{minimize} & f(\boldsymbol{h}, \boldsymbol{d}, \boldsymbol{x}), \\ \text{subject to} & \boldsymbol{g}(\boldsymbol{q}, \boldsymbol{d}, \boldsymbol{x}) = \boldsymbol{0}, \\ & \boldsymbol{e}(\boldsymbol{h}, \boldsymbol{d}, \boldsymbol{x}) = \boldsymbol{0}, \\ & \boldsymbol{h}_{\min} \leq \boldsymbol{h}(\boldsymbol{d}, \boldsymbol{x}) \leq \boldsymbol{h}_{\max}, \\ & \boldsymbol{j}_{\min}(\boldsymbol{x}) \leq \boldsymbol{j}(\boldsymbol{x}) \leq \boldsymbol{j}_{\max}(\boldsymbol{x}), \\ & \boldsymbol{w}_{\min}(\boldsymbol{h}, \boldsymbol{q}, \boldsymbol{d}, \boldsymbol{x}) \leq \boldsymbol{w}(\boldsymbol{h}, \boldsymbol{q}, \boldsymbol{d}, \boldsymbol{x}) \leq \boldsymbol{w}_{\max}(\boldsymbol{h}, \boldsymbol{q}, \boldsymbol{d}, \boldsymbol{x}). \end{array}\right\} \tag{3.1}$$

3.5 Design Considerations for Water Distribution Networks

WDS design optimisation involves many complicating practical issues, making automated design difficult to realize. In this section some of these complicating issues are discussed.

There is considerable uncertainty in WDS planning. The most significant uncertainty comes in the form of estimating current and future water demands. Small systems may be greatly affected by relatively minor events, such as a new customer joining the network. In a large system, the complexity of growth is highly uncertain. An economic downturn or upturn may have a dramatic effect on demand. Demand estimation is discussed briefly in Chapter 4 [251].

There is a trade-off between having a conservative, oversized system and a least-cost, lean system. The over-allocated system entails a higher capital cost and longer residence times which may affect water quality adversely, but it has the advantage of providing additional water for fire flow and additional capacity for future growth. A system which only just meets the current demands may experience low pressures, inadequate fire flows, be unable to respond adequately to

system failures, and impose unrealistic limits to growth. Building robust, reliable systems should be as important as lowering the capital costs. These situations introduce competing objective functions which seek to minimize cost and maximize capacity [251]. Another complication is that there is a feedback effect between pipe sizing and demands. If a pipe has spare capacity, chances are that it will be utilized by new developments [251]. There is a prevailing opinion amongst engineers that it is desirable to over-design a WDS so as to achieve robustness and allow room for growth.

There is also considerable uncertainty in cost information. It is possible that cost data are only accurate to ±20%, with solutions differing in cost by only 1–2%. This is not so problematic if the relative costs are consistent, because the solutions will still be ordered correctly in terms of cost, however it may cause serious problems when considering the trade-off between capital and running costs. The cost of installing a pipe is not merely a function of its length and diameter; it also includes excavation and other costs. If the network layout itself has to be designed, alternative pipe routes may involve extremely different excavation, paving and right-of-way costs, which may dwarf the pipe costs [251]. This strongly suggests the incorporation of layout design in the automated design procedure.

Another important consideration which complicates the problem is staged development, or development over time. In 1999, Halhal *et al.* [117] developed an optimisation methodology which schedules improvements to a WDS optimally. This maximizes accumulated benefits over time and minimizes the sum of the present value of staged investments.

Pumps may be required to run at close to maximum efficiency for the majority of the test period, and it is possible to design their periodic on/off schedules. Tanks may be required to exhibit a daily or weekly water recharge cycle (tanks should fill and empty over their operational range). Both the initial installation and operational costs should be considered over the specified period. These costs must be expressed in present value. Savic *et al.* [211] have developed a method for an optimal selection of pumps for installation in new or upgraded pump stations [251]. Tank design also requires the simulation of emergency situations and instantaneous peak flows, for which emergency storage capacity should be available. A full simulation over the course of a day may be too time intensive for use in design optimisation; a small representative number of periods of the day is typically used. Approximation techniques are required to maintain consistency of tank levels when approximate periods are used. Representative periods may include peak day demand and average day demand, in 24 hourly periods. Designing pumps, tanks and valves adds the additional complication of mixing discrete (pipe sizes) and continuous variables [169].

In addition to spare storage capacity, designing for fire-flow scenarios may increase the size of pipes. Models may even consider the trade-off between available fire-flow and economic losses due to building damage caused by fire [90].

Another interesting problem is the trade-off between excess pressure and potential leaks in the system. Pipes under higher pressure will deteriorate more rapidly, and when they develop leaks, water losses will be larger. Several researchers have suggested incorporating leakage models [229]. A brief discussion of leakage models appears in the next chapter.

It is also possible to consider pipe failures in the process of WDS design, incorporating their replacement costs over time, and determining how the network will be affected hydraulically. Some researchers have used a single pipe failure model in which individual failure of all the pipes in the system are simulated, one at a time, and used the effect on customer service as a result of these failures to determine a reliability measure [49]. However, this does not scale practically to larger systems. A more practical model may be a probabilistic one in which the failure of pipes

over the system lifetime is modelled, using failure rates as determined from historical records or physical experimentation [112].

If an existing system is expanded or rehabilitated, the optimisation algorithm requires a calibrated model of the system in question. Models may be calibrated using real test data, obtained either from site fire-flow and pressure tests, metered demands, online monitoring systems (so-called *SCADA* systems) or a combination of these. Calibrating a model is an optimisation exercise in its own right and many algorithms have been developed for it (see pages 251–291 in [251]). Data collection and hydraulic model calibration are not dealt with in this dissertation.

Numerous researchers have proposed models enforcing loops, but care must be taken to ensure that loops are actually reliable, and not merely topological features. For example, a 50 millimeter diameter pipe connected in a loop to large 1 meter diameter pipe will not provide a sufficient alternative flow-path should the large pipe fail [250]. Several reliability measures have been designed specifically to encourage loops of similarly sized pipes or balanced flow characteristics. Two such reliability measures are presented in Chapter 4.

Finally, aesthetic concerns may also have an impact on designs, especially when considering raised water tanks [251]. These considerations are difficult to incorporate into models, except as constraints on tank elevation and size. However, at some point consumers are probably going to have to sacrifice scenery for a functional WDS.

3.6 A Concise History of WDS Design Problem Solutions

The traditional method for designing WDSs involves a trial and error approach guided by expert experience. A common rule of thumb has been to ensure that the slope of the hydraulic grade or the flow velocity along a pipe lay within certain bounds. The small number of configurations which could thus be tested was an obvious limitation [83, 251].

The earliest attempts at water network optimisation were by Babbit and Donald in 1931, Camp in 1939, and the first computerized attempt by Schaake and Lai in 1969 [251].

A major event in the comparison of water network design optimisation techniques occurred in 1985 at the "Battle of the Network Models." This was a series of sessions held at the ASCE Water Resources Planning and Management Conference in Buffalo, New York, where researchers were required to optimize the design of a realistic model system called "Anytown". The participants used optimisation models to size the piping system, whilst manually choosing the location and size of tanks. The participants used different optimisation models, which were based on linear programming, partial enumeration, or nonlinear programming techniques. No attempt was made to optimize the location and dimensions of tanks or pumps automatically, which relied entirely on expert judgment. The majority of the participants were able to find a solution that would work at peak loading, but would not have adequate capacity to fill the tanks at non-peak times [87]. The algorithm that performed the best during these trials was the partial enumeration algorithm by Gessler [99].

Of the earlier optimisation models reported in the literature, some of the most useful were those by Alperovits and Shamir [9] in 1977, Quindry *et al.* [197] in 1983, Morgan and Goulter [111], and Gessler and Walski [99], both in 1985. A common shortcoming of these models was their inability to design and analyze a complete WDS. Their limitations included the maximum network size they could handle, the number of loading conditions that could be analyzed, and the types of components that could be designed [159, 251].

In 1977 Alperovits and Shamir [9] developed an approach known as the *linear programming gradient method*, whereby the original nonlinear problem is approximated as a sequence of linear subproblems. The primary variables are the flows in the network. For each flow distribution the other decision variables (such as pipe diameters or pump capacities) are optimized iteratively by means of linear programming. This approach has been adapted and improved by many researchers such as Quindry *et al.* [197] in 1981, Featherstone and El-Jumaily [89] in 1983, Fujiwara *et al.* [94] in 1987, and Loganathan *et al.* [166] in 1995. However, these linear approximation methods do not guarantee optimality to an intrinsically non-linear problem, produce unrealistic output solutions (so-called *split-pipe solutions* where a pipeline is divided into several sub-lengths of different diameters) or those having a lack of redundancy (no loops), and also often suffer from efficiency problems [83].

A number of researchers have applied nonlinear optimisation techniques to WDS design problems. This includes Shamir [214] in 1974, Ormsbee and Contractor [187] in 1981, Chiplunkar *et al.* [33] in 1986, El-Bahrawy and Smith [79] in 1987, Su *et al.* [223] in 1987, Lansey and Mays [159] in 1989, Duan *et al.* [74] in 1990, Cullinane *et al.* [45] in 1992. The limitations of nonlinear optimisation techniques are that they frequently become trapped at local optima and that they typically employ continuous variables, requiring conversion to discrete values which may reduce the quality of the solution [83, 251]. Mathematical programming approaches generally function by iteratively fixing pipe flow rates or pressure heads, optimizing the pipe sizes for the given values, updating the flows or heads, and re-optimizing, until convergence is achieved. Most of these approaches are limited to considering only a single demand loading condition. Although Morgan and Goulter [111] considered multiple loadings, it also produced *split-pipe* solutions [157].

It is possible to simulate all network configurations in a *complete enumeration* of the search space, but this becomes highly impractical as systems grow larger. For example, a medium-sized system of 100 pipes, each with 10 diameter options would require 10^{100} hydraulic simulations. Even if a million such simulations could be conducted in a second, this would require more than 3^{86} years to complete, far longer than the age of the known universe. In 1985 Gessler and Walski [99] proposed a selective enumeration method for eliminating certain inferior solutions from being evaluated by a hydraulic simulator (see §3.7.1). However, in 1992 Murphy and Simpson [178] showed that the approach failed to locate an optimal solution even for a medium-sized network [83].

It was suggested by Goulter [109] in 1988 that the minimum cost design for a given layout and single loading case should be a branched network (that is, a network with no loops). While this may be true, loops are an essential feature of actual distribution systems as they provide alternative flow paths in the case of pipe failure or for maintenance purposes. One can achieve a degree of redundancy in pipe network optimisation by ensuring that the layout has appropriate loops and by specifying minimum diameters for all pipes [251].

In 1990 Monbaliu *et al.* [175] proposed a greedy rule-based gradient search technique. Initially, all pipes are set to their minimum diameters and the pressures at the network nodes are computed by simulation. Until all pressure constraints are satisfied, an iterative procedure is employed whereby the pipe with the maximum head loss per unit length is increased to the next available size and the network pressures recomputed. This algorithm's effectiveness cannot be ensured, owing to the complex nonlinear interactions between the system components [83]. Other greedy WDS design heuristics have been proposed, such as those by Todini [227] in 2000, and Afshar *et al.* [4] in 2005.

Two-phase searches were developed by some researchers in order to improve the chance of

finding global optima [83]. This included models by Fujiwara and Khang [94] in 1990, Eiger *et al.* [78] in 1994, and Loganathan *et al.* [166] in 1995.

Several researchers have investigated layout geometry optimisation, neglecting the strong coupling which exists between component sizing and layout determination for pipe networks. This includes work by Goulter and Morgan [111] in 1985, Walters and Lohbeck [253] in 1993, Davidson and Goulter [53] in 1995, Walters and Smith [255] in 1995, Davidson [52] in 1999, and Geem *et al.* [96] in 2000. One exception to the two phase design methodology appears in the work by Afshar *et al.* [4], where a so-called *maximal layout* of all practically possible pipe pathways is considered, and the algorithm is allowed to eliminate pipes during the course of optimisation.

During the 1990s researchers began to investigate the use of stochastic optimisation techniques for solving WDS design problems. Numerous *metaheuristics* were implemented for this purpose. Genetic algorithms, simulated annealing and tabu searches are common metaheuristics. Heuristic searches typically consider only the objective function value of current solutions during an iterative search procedure. These solutions are evolved over generations, using stochastic variation operators (*e.g. mutators*, information exchanges (*crossover*)). Metaheuristics often encounter difficulty when dealing with constraints, and must instead use specialized techniques to accommodate them, such as weighted penalty functions that adapt the objective function. A drawback of metaheuristics is the large number of simulations that must be performed, since response surface gradient information is typically not used [83, 218, 251].

A host of metaheuristics have been applied to least-cost WDS design optimisation. Genetic algorithms (GAs) for solving WDS design have been used by Hadji and Murphy in 1990 [115], Murphy and Simpson [178] in 1992, Walters and Cembrowicz [252] in 1993, Simpson *et al.* [218] in 1994, Dandy *et al.* [51] in 1996, Savic and Walters [209] in 1997, Halhal *et al.* [116] in 1997, Gupta *et al.* [114] in 1999, Lippai *et al.* [163] in 1999, Walters *et al.* [254] in 1999, and Wu *et al.* in 2001 [262] and 2002 [265]. *Simulated annealing* was applied by Loganathan *et al.* [166] in 1995, and by Cunha and Sousa [46] in 1999. *Tabu searches* were employed by Fanni *et al.* [85] in 2000, and by Cunha and Ribeiro [47] in 2004. Simpson *et al.* [219] were the first to apply *Ant Colony Optimisation* (ACO) to WDS optimisation in 2001. A study by Zecchin *et al.* [275] in 2007 compared five different ACO algorithms for WDS design. *Particle swarm optimisation* was applied by Vairavamoorthy and Shen [236] in 2004, and more recently by Izquierdo *et al.* [132] in in 2008. In 2002, Geem *et al.* [97] applied the musically inspired *harmony search* to the optimisation of WDSs, and again in 2009 [98] including pumps, demonstrating better results than both GAs and simulated annealing. In 2003 Eusuff and Lansey [83] developed the *Shuffled Frog Leaping Algorithm*, a *memetic* metaheuristic which proves more efficient than GAs for WDS design. Mohan and Babu [174] used *Honey-Bee Mating Optimisation* in 2010, improving optimisation efficiency by an order of magnitude over previous metaheuristics.

Researchers struggled for many years with the problem of formulating their models to accommodate additional components such as tanks and pumps, and to handle the simultaneous inclusion of multiple objectives. It has since become widely acknowledged that there is more to WDS design than just deciding on pipe sizes, and that least-cost designs are not adequate [250]. Most multi-objective WDS design formulations have considered the objectives of cost and reliability [141]. The reliability of a WDS is a nebulous notion, owing to the vast number of differing definitions of and approaches towards measuring reliability over the years. It is generally accepted that there is a strong correlation between reliability, redundancy and capacity, and that increased reliability comes at a price, requiring larger or more WDS components [251].

Attempts at multi-objective optimisation for water network design were made by amongst others Walski *et al.* [247] in 1990, Halhal *et al.* [254] in 1999, Xu and Goulter [268] in 1999, Todini [227]

in 2000, Dandy and Engelhardt [48] in 2001, Farmani *et al* [86, 88] in 2003 and 2005, Tolson *et al.* [230] in 2004, by Prasad and Park [194] in 2004, Kapelan *et al.* [141] in 2005, Keedwell and Khu [143] in 2006, di Pierro *et al.* [68] in 2009, and Vasan and Simonovic [243] in 2010.

Although a Pareto-optimal set may be approximated by repeatedly solving a single-objective optimisation problem with different goals, this method is known to be computationally inefficient. *Multi-objective evolutionary algorithms* (MOEAs) may be used to determine an entire Pareto-optimal front in a single optimisation run. Multi-objective optimisation for WDS design is discussed further in Chapter 5.

3.7 A Survey of WDS Design Optimisation Methods

This section constitutes a survey of the optimisation techniques which have been used to solve the least-cost WDS design optimisation problem. These methods are thus all single-objective optimisation techniques, although some may be adapted for multiple objectives, as shall be demonstrated in Chapter 5. The methods discussed include Partial Enumeration, Linear and Nonlinear Programming, Simulated Annealing, Tabu Search, Genetic Algorithms, Ant Colony Optimisation, Shuffled Complex Evolution, Particle Swarm Optimisation, and the Shuffled Frog Leaping Algorithm.

3.7.1 Enumeration and Grouping

In 1985, Gessler [99] developed the *Partial Enumeration Method* (PEM) to optimize water networks based on the enumeration of a limited number of alternatives. Inferior solutions may be eliminated from evaluation using two considerations. The first consideration is that after a combination of pipe sizes has been found which gives a hydraulically feasible solution, there is no need to test any other combination which has a significantly higher cost. The second consideration is that after an infeasible solution has been found, any other combination with sizes equal to or smaller than these is also infeasible. Another technique which dramatically reduces the solution space is placing adjacent pipes together in groups which are sized identically. This reflects reality in that utilities are unlikely to have pipes connected with dramatically different diameters (except of course for service lines branching off main pipes). However, the specification of sensible groups in itself represents a tricky optimisation problem [251].

The PEM drastically reduces the number of combinations which have to be balanced hydraulically. However, it still constitutes a rough pruning of the search space and cannot guarantee finding a global optimum [246]. Additionally, although it works well for small networks, the PEM does not scale sufficiently to handle real-world networks [178].

3.7.2 Linear Programming

Linear programming (LP) approaches have been used to reduce the complexity of the nonlinear system by solving a sequence of linear sub-problems. The simplified problem of sizing only pipe diameters may be expressed as the minimisation of a cost function, $f(\boldsymbol{x}) = \sum_{i=1}^{n_\mathrm{p}} \sum_{j=1}^{\omega} C_i(x_{i,j})$, where $x_{i,j}$ denotes the length of the portion of pipe i which has diameter j, where n_p denotes the number of pipes in the system, where ω denotes the number of sizes, and where $C_i(x_{i,j})$ denotes the cost function of purchasing a length $x_{i,j}$ of pipe i of diameter j.

In addition to the basic hydraulic constraints, another constraint ensures that the sum of all segments of pipe between any two nodes is equal to the length between those two nodes. That is,

$$\sum_{j=1}^{\omega} x_{i,j} = L_i,$$

where L_i denotes the total length of pipe i. Note that if a particular pipe size is not included, then the length of its segment is zero.

An optimum solution obtained by this method consists of one or two pipe segments of different discrete sizes between each pair of nodes. These 'split-pipe' solutions are highly impractical, particularly when short pipe lengths of different diameters are used [250]. Furthermore, the matrix inversion required to solve the linear problem is inconvenient and computationally expensive for large systems. Therefore, LP approaches to pipeline design are not recommended, and a description of such approaches is only included here for historical interest.

3.7.3 Non-linear Programming

Non-linear programming (NLP) methods treat pipe sizes as continuous variables, and therefore require rounding of sizes to discrete diameters, which is typically sub-optimal. NLP requires the calculation of the partial derivatives of the objective function value with respect to all the decision variables, which is only possible if the decision variables are continuous, and may nonetheless be difficult. Nonlinear programming has been used for pipe sizing problems by Jacoby [135] in 1968, Lam [155] in 1973, Ormsbee and Contractor [187] in 1981, and Lansey and Mays [159] in 1989, but has been largely unsuccessful as a robust optimisation technique.

One of the most popular approaches to nonlinear programming is the *Generalized Reduced Gradients* (GRG) method, able to solve systems with nonlinear objectives and constraints. GRG was popularized by Lasdon and Waren [160] in 1982. A serious limitation of NLP formulations for WDS design was the size of problem they could handle. This was addressed by the 1989 model of Lansey and Mays [159], which represents a 'mature' NLP model for WDS design using GRG. It is able to design components in addition to pipes and consider multiple demand loading conditions. The largest portion of constraints in the WDS design problem is the hydraulic constraints of continuity of flow and conservation of energy. Their model significantly reduces the number of constraints accommodated by the optimisation model by *outsourcing* the hydraulic simulation role to an external simulator (they employed KYPIPE by Wood [261]), which enforces hydraulic constraints implicitly. This reduces the problem dimensionality greatly, effectively allowing much larger systems to be designed under GRG. To the best of this author's knowledge, this was the first time that outsourcing to an external simulator was used as a strategy in automated WDS design, a methodology which went on to become mainstream.

Lansey and Mays [159] reformulated the model in (3.1) to express nodal head $\boldsymbol{h}$ as a function of design variables $\boldsymbol{x}$, and modified the objective function to include an *augmented Lagrangian penalty function* [193] with a term for each pressure constraint. This yields a reduced model of the form

$$\left.\begin{array}{ll} \text{Minimize} & f(\boldsymbol{h}(\boldsymbol{x}), \boldsymbol{x}, \boldsymbol{\mu}, \boldsymbol{\sigma}) = f(\boldsymbol{x}) + \frac{1}{2}\sum_i \sigma_i \left(c_i - \frac{\mu_i}{\sigma_i}\right)^2 + \frac{1}{2}\sum_i \frac{\mu_i^2}{\sigma_i}, \\ \text{Subject to} & \boldsymbol{j}_{\min}(\boldsymbol{x}) \leq \boldsymbol{j}(\boldsymbol{x}) \leq \boldsymbol{j}_{\max}(\boldsymbol{x}), \end{array}\right\} \tag{3.2}$$

where σ_i and μ_i are *penalty weights* and *Lagrange multipliers* respectively for the i-th constraint, where c_i denotes the modified constraint equation, and where $j_{\min}$, j and $j_{\max}$ have the same

meaning as in (3.1). The model uses a two level procedure of nested loops, in which the outer loop adjusts the penalty weights and Lagrange multipliers (using a heuristic based on the magnitude of infeasibilities), and the inner loop uses the GRG2 code by Lasdon and Waren [160] to determine the change of design variables for fixed $\boldsymbol{\sigma}$ and $\boldsymbol{\mu}$ values. GRG2 is based on the GRG method, and employs a two-step procedure to calculate reduced gradients and adjust decision variables accordingly. KYPIPE was used within GRG2 to evaluate the objective function. Unfortunately, despite a multi-start search, the Lansey and Mays model still tends to become trapped at local optima, a problem frequently associated with NLP techniques. Lansey and Mays handled solutions with continuous pipe sizes by calculating an equivalent split pipe solution. A better technique is that of Afshar *et al.* [4] which uses a heuristic method to convert continuous sizes to discrete diameters.

Both linear and non-linear programming methods require significant simplifications to solve real world systems. Another major disadvantage of these approaches is that they produce only a single solution per optimisation run. This paradigm has been superseded by a multi-objective approach producing multiple non-dominated solutions [159, 251]. Owing to these drawbacks, mathematical programming techniques are not considered further in this dissertation.

3.7.4 Simulated Annealing

The idea behind the method of *Simulated Annealing* (SA) is derived from the analogy of heating and cooling of materials in order to increase their strength, as is frequently done in pottery and metalwork. The method is based on similarities between the way in which a metal cools and freezes into a minimum energy crystalline structure and the search for a minimum in a more general system [46, 146].

SA begins at a high 'temperature', which means that it initially has great flexibility in moving randomly through the solution space. If the search space can be viewed as having valleys at the bottom of which are local optima (minimisation problem), then ideally every valley should initially be accessible. As time continues, the temperature is decreased, which decreases the flexibility of the search, and may cause it to become trapped in a particular valley. If a new (random) move results in a solution which has a lower altitude, that solution is adopted as the current solution. Not all moves to points higher than the current best are immediately rejected. An acceptance criterion is used to decide which solutions to adopt during the search. This criterion is based on the height difference between the last saved lowest valley and the current solution. Probabilistic decisions are made about whether to stay in a new lower valley or to move out of it. The acceptance criterion depends on the current 'temperature' and uses the *Boltzmann probability distribution* [146].

The *annealing schedule* provides the rules for lowering the temperature over time. Decreasing the temperature too slowly makes the search inefficient, and cooling too rapidly results in the search becoming trapped in a sub-optimal region of the solution space. The cooling schedule typically employs either a constant amount temperature reduction scheme, or a constant factor temperature reduction scheme (such as 10% of the current temperature), for each temperature phase. The first scheme explores an equal number of solutions in each temperature zone, and the second scheme spends more time in the lower temperature zones. Another important aspect is the determination of the number of random steps required at each temperature [46].

The basic SA algorithm, as formulated in [46], appears in pseudocode form as Algorithm 1.

In 1999 Cunha and Sousa [46] proposed a scheme whereby the neighbourhood of a solution $\boldsymbol{x}$ is defined as any solution $\boldsymbol{x}'$ differing in configuration by a single pipe whose size is above or

Algorithm 1 Simulated Annealing Algorithm

Input: A combinatorial optimisation problem with a domain set for each decision variable. An initial configuration $\boldsymbol{x}_1$, an initial temperature T_1, a stopping temperature T_f, and a cost function $C(\boldsymbol{x})$ providing the cost of configuration $\boldsymbol{x}$.

Output: An approximation of the global optimum.

1: Set $j \leftarrow 0$.
2: Set $j \leftarrow j + 1$.
3: Choose a random neighbour $\boldsymbol{x}'_j \in N(\boldsymbol{x}_j)$, where $N(\boldsymbol{x}_j)$ is the neighbourhood of $\boldsymbol{x}_j$.
4: Generate a random $u \in [0, 1]$.
5: **if** $u \leq \exp([C(\boldsymbol{x}_j) - C(\boldsymbol{x}'_j)]/T_j)$ **then**
6: Set $\boldsymbol{x}_{j+1} \leftarrow \boldsymbol{x}'_j$
7: **else**
8: Set $\boldsymbol{x}_{j+1} \leftarrow \boldsymbol{x}_j$
9: **end if**
10: Choose $T_{j+1} \leq T_j$
11: **if** $T_{j+1} \geq T_\mathrm{f}$ **then**
12: Repeat from step 2.
13: **end if**

below its current size. Whether or not a new solution is accepted as the new starting point is based on the *Metropolitan criterion*, $\exp((f(\boldsymbol{x}) - f(\boldsymbol{x}'))/T)$. Note that $f(\boldsymbol{x}) - f(\boldsymbol{x}')$ is negative when new solutions are more expensive than the originals, so that the Metropolitan criterion yields a probability value in (0,1). This allows solutions to be considered even if they are worse than the current solution. Furthermore, improved solutions are always selected ($e^z > 1$ for $z > 0$). As the temperature lowers, worse solutions will have an increasingly lower probability of acceptance. The initial temperature should therefore be defined to provide a reasonably high probability of moving from a better solution to a worse one. Cunha and Sousa used an initial probability of 20–90% as a guideline. They also performed a 10–90% reduction of temperature at each temperature stage, calculating 10, 40 and 70 iterations per stage [46]. In 1985, Aarts and Van Laarhoven [1] demonstrated that the number of iterations the algorithm should make at each temperature is an exponential function of problem size. This permits the system to move close to its stationary distribution before making a temperature reduction. As such, a global optimal solution will occur with a probability equal to 1 as the temperature is reduced to zero [46]. This does not bode well for large water distribution networks which have a problem size which grows exponentially as a function of number of constituent components.

Simulated annealing may be applied to highly non-linear problems with multiple local optima and many constraints. It is considered to be a robust and general technique which frequently approaches a global optimum. The algorithm (given some assumptions on the cooling of the temperature) is proven to converge to an optimal solution of a problem. Unfortunately, this may take infinitely long to achieve [176]. SA was first used for WDS design by Loganathan *et al.* [166] in 1995. In 1999, Cunha and Sousa [46] achieved high quality solutions to two benchmark problems from the literature. However, SA is not considered further in this dissertation, as the focus is on population-based metaheuristics, which lend themselves more readily to multi-objective optimisation.

3.7.5 Tabu Search

The method of *tabu search*, proposed by Glover [102] in 1986, makes systematic use of various memory structures, storing an itinerary of visited solutions. This information is used to restrict the choice of moves by prohibiting the reversal of recently employed moves, thereby avoiding cyclic local searches. It functions as a neighbourhood search, with a single current solution which is updated during successive iterations. Essentially, it is a steepest decent search with a *short term memory*, although variations may include a *long term memory* which stimulates diversification. A pseudocode listing for a generic tabu search appears in Algorithm 2 [73].

Algorithm 2 Simple Tabu Search Algorithm

Input: A combinatorial optimisation problem specification including a domain set for each decision variable. An initial configuration $\boldsymbol{x}_1$, a memory size $\hat{m}$. An objective function $f(\cdot)$ to determine solution fitness. An adjustment function to reach neighbours.

Output: An approximation of a global optimum solution.

1: Set the current solution $\boldsymbol{x}$ to the initial solution $\boldsymbol{x}_1$. Set the tabu list $\mathcal{T}$ to be the empty set, $\mathcal{T} \rightarrow \emptyset$.
2: Generate a set of neighbours of $\boldsymbol{x}$ (the set of solutions accessible by an elementary *movement* or modification). If the complete set of neighbours $\mathcal{N}(\boldsymbol{x})$ is too large, apply a reduction or filtering technique to reduce its size. Let the remaining set be $V(\boldsymbol{x}) \subset \mathcal{N}(\boldsymbol{x})$.
3: Remove all candidates $\boldsymbol{x}' \in V(\boldsymbol{x})$ for which the move $\boldsymbol{x} \rightarrow \boldsymbol{x}'$ appears in the tabu list.
4: Evaluate the objective function $f(\boldsymbol{y})$ for each remaining $\boldsymbol{y} \in V(\boldsymbol{x})$. Let $\boldsymbol{y}^*$ be the solution which has the best objective function value amongst these elements.
5: Replace the current $\boldsymbol{x}$ with $\boldsymbol{y}^*$, even if it has a worse objective function value. Insert the reverse move, $(\boldsymbol{y}^* \rightarrow \boldsymbol{x})$, into the tabu list $\mathcal{T}$. Remove the oldest element of the tabu list if its size exceeds $\hat{m}$.
6: Repeat steps 2 to 4 until a termination condition is reached.

The choice of a neighbourhood is important for effective search and computational reasons. A neighbourhood that is too large may be reduced by, for example, randomly choosing a subset of $\mathcal{N}(\boldsymbol{x})$, also known as a *candidate list* [101]. Considered in terms of a set of attribute modifications $\mathcal{M}$, which cause a movement to a neighbour, this reduction of the neighbourhood may be achieved by partitioning $\mathcal{M}$ into a number of subsets and examining only one subset per iteration. Another method is to compute the entire current neighbourhood of the current solution for all $\mathcal{M}$, and to order it in terms of increasing objective function values. This list may then be used for consecutive solutions by assuming that a good move for a nearby neighbour is also a good move for the current solution. As the current solution moves away from this original neighbour, the ordered list of moves will become less relevant and will have to be updated and resorted [73].

The size of the tabu list $\hat{m}$, also known as the *tabu tenure*, is normally only a fraction of the size of $\mathcal{M}$, typically a few tens of moves. However, it is advised that a sensitivity analysis be performed on the tabu tenure, as the effect thereof may vary substantially from problem to problem. The use of the tabu list allows the search to progress in different trajectories from the same solution. The larger the tabu tenure, the more likely a search will be to move out of a local optimum valley, but also the less thoroughly it will be able to explore that valley. Tabu search proves to be more effective when the tabu tenure is not static, but rather varies stochastically between certain limits, or changes dynamically on the basis of characteristics observed during the search [73].

Long term memory may be employed to reduce the probability of revisiting solutions [101]. Long term memory may be required as a mechanism to escape local optima, especially in cases of extended valleys in objective space. One technique in use is an additional tabu list which prohibits moves whose frequency of occurrence has exceeded a predefined threshold, or the penalization of solution fitness proportional to the frequency of their appearance in the search. This penalization method may also be of benefit for cases where the objective function assumes a limited number of values. An obvious problem is the selection of an appropriate penalty factor, which should neither be too small nor too large. Another mechanism for achieving long term memory is to oblige moves which have not yet been executed after a large number of iterations. It is possible to oscillate periodically between using long term and short term memory, or to use the two simultaneously. If such oscillation occurs, this may be understood in terms of periods of *diversification* (long term memory) and *intensification* (short term memory) [73, 101].

Tabu search was utilized for WDSs design optimisation by Fanni *et al.* [85] in 2000, and by Cunha and Ribeiro [47] in 2004, who used a neighbourhood wherein an individual differs from its neighbour by exactly one pipe whose diameter is one size larger or smaller. They employed moves to increase and decrease pipe sizes in periods of diversification and intensification, making use of frequency memory, including the number of times a pipe's diameter was changed, the number of times it was assigned a particular value, and the number of iterations for which a pipe maintained its current size. Although they were able to achieve high quality solutions in times competitive with alternative methods, such as genetic algorithms, no general conclusions could be drawn about which is the most appropriate design metaheuristic [47]. Tabu search is not considered further as a solution methodology for WDS design in this dissertation, as the focus is on population-based metaheuristics, which lend themselves more readily to multi-objective optimisation.

3.7.6 Genetic Algorithms

Genetic algorithms (GAs) draw parallels from Darwinian evolution, the principles which underly the development and diversification of life. They use a simplified evolutionary model to 'evolve' good solutions, and determine solutions to highly complex problems which may be intractable to classical optimisation techniques [251]. The original theory underlying GAs was formulated by Holland in 1975 [123], and further developed by others including Goldberg [105, 106, 107] and Deb [56, 57].

GAs incorporate ideas such as a population of diverse solutions to a problem, the survival and reproduction of the fittest solutions (with fitness determined by the objective function value), the inheritance of genetic material by child solutions from their parents by means of genetic *crossover operations*, and occasional *mutations* which cause lateral movement through the solution space, thus avoiding local optima. The transition operators are governed by probabilistic rather than deterministic rules [105]. Although GAs use stochastic mechanisms to perform their simulation, they are distinctly non-random in that there is pressure to 'evolve' by improving their fitness. Mutation constitutes the small but necessary random-search component of genetic algorithms [251]. Whereas conventional optimisation techniques commonly search from a single point, GAs search from an entire population of points, capable of climbing many peaks in parallel, also known as a *multimodal search*. If special knowledge of the solution space is known beforehand, *seed solutions* may be planted to assist the search [105]. In order to be used by GAs, solutions must be described by a finite coded representation (typically a binary string) or *chromosome*, made up of a linear succession of genes, such that both gene value and position within the chromosome have meaning. This is analogous to the chromosomes found in human DNA. Each

chromosome represents a possible location in a multidimensional search space. A chromosome may represent a set of decision variable values, such as pipe sizes, which require optimisation. GAs are highly flexible with respect to the shape of the feasible region and genetic encoding of a system solution, so that they may be used in a wide variety of problems [6, 105, 106, 209].

A pseudocode listing for the standard genetic algorithm formulation is provided in Algorithm 3 [251].

Algorithm 3 Standard Genetic Algorithm

Input: A combinatorial optimisation problem specification including a domain set for each decision variable. A population size N, a probability of crossover p_c, and a probability of mutation, p_m. A genetic code formulation with a function mapping code substrings to a decision variable values. An objective function $f(\cdot)$ to determine individual fitness.

Output: A converged population of solutions containing an approximation of a globally optimal solution to the combinatorial optimisation problem.

1: Randomly generate an initial population of N solutions.
2: Calculate the fitness of each individual solution by means of the objective function.
3: Generate a new population using the crossover and mutation operators, applied with probability p_c and p_m respectively. Individuals with higher fitness must have a higher probability of reproducing.
4: Calculate the fitness of the new solutions.
5: Repeat steps 3 to 5 until a termination condition is reached.

It is suggested that GAs employ a moderate population size, a high crossover probability, and a low mutation probability, although GAs may be rather insensitive to the actual probability values applied. Typical values for a population size is between 50 and 1 000, depending on the size of the search space and computational complexity of calculating fitness [6, 105].

Individuals are selected to mate based on their fitness. A popular selection method is *fitness-proportionate selection*, whereby the number of times an individual is expected to reproduce is proportional to the ratio of its fitness to the total fitness of the population. This method may be implemented using *roulette-wheel selection* proposed by Goldberg [106] in 1989. This assigns every individual a slice of a simulated roulette wheel, with the size of the slice depending on their fitness, so that those with a larger slice have a greater chance of selection when the wheel is 'spun' [251]. *Selection pressure* is the probability that the best solution is chosen for reproduction. To avoid premature convergence this should usually be in the region of twice the probability of selecting a solution of average fitness. *Binary tournament selection* is another selection mechanism whereby two solutions are selected at random from the population and the fitter of the two are selected as a parent. This provides more consistent selection pressure than roulette-wheel selection throughout the optimisation [56].

GAs usually incorporate soft constraints in the form of fitness function *penalty terms* used to penalize insufficient levels of service (such as low pressure) or exceeding normally hard constraint limits. This may result in infeasible solutions becoming members of the population, but ideally the penalty term should ensure that they have a lower fitness relative to the feasible members. Stated another way, penalty functions should be graded as a function of distance from feasibility. This is sometimes achieved in the form of a squared error added to the base fitness [105]. A *penalty multiplier* is often used to normalize the penalty values to the same scale as the basic fitness [105, 251]. Another technique frequently used is a *repair operator*, which attempts to 'fix' infeasible solutions. This typically requires heuristic information from the problem domain. Such operators are recommended when available [6].

The reproduction operation occurs either as a direct copy of a solution into a successive generation [105], or as a *chromosome crossover* operation, analogous to how paired chromosomes in nature exchange pieces of genetic material. The crossover operator takes two individuals and cuts their chromosome strings at some randomly chosen point, typically selected using a uniform probability distribution [105]. The front section of each chromosome remains in place and the tail section is swapped between the two. This produces two new individual chromosomes. Crossover need not always occur and typically takes place with a probability in the range 0.5 to 1 [251].

Mutation plays a secondary role in the reproduction phase. It is not the dominant force in evolution, as is popularly believed. Mutation randomly alters the individual genes (chromosome string characters or bits) with a small probability, introducing desired diversity. If the probability of mutation is too high, the algorithm degenerates into a random search, which is undesirable. The change in the frequencies of specific gene values (alleles) in a population due to randomness (*i.e.* mutation) is known as *genetic drift*, which, if excessive, is degenerative. The frequency of mutation to obtain good results in empirical studies is of the order of 1 mutation per thousand bit transfers. This rate is similar to that found in nature. The typical values of probability of mutation are between 0.001 and 0.01. Another suggestion is that mutation probability be inversely proportional to population size N, $0.1/N < p_{\mathrm{m}} < 5/N$. Mutation should theoretically enable any solution to be reached from any other solution (albeit with a minuscule probability) in order to guarantee convergence to the global optimum given infinite processing time [105].

The GA may terminate after a specified number of generations, or when the differences in total population fitness of successive generations stabilize within a specified range for enough consecutive generations [105]. An example of a GA applied to a simple quadratic maximisation problem is presented in Appendix C.1.

Where specialized techniques for solving a problem exist, these may well outperform GAs in terms of speed and accuracy. GAs may be computationally intensive depending on the application, and populations tend to improve more slowly as the search approaches a global optimum [251]. GAs calculate the objective function value of every generated solution. Since they produce a population of solutions, they are ideally suited for multi-objective analysis, which requires a set of alternative solutions [265]. Moreover, GAs seem capable of meeting WDS design needs without distorting or oversimplifying the optimisation problem. GAs are free from a particular program structure, require a minimum of auxiliary information about the problem (only a fitness function), and have a perspective which may be considered global [105]. They are considered to be efficient and extremely robust, capable of yielding good solutions even in cases of highly nonlinear problems with multiple optima, discontinuities, and non-differentiable functions. They are able to perform a thorough search of the solution space, and have the ability to approach global optimality, as has been demonstrated in numerous investigations [105, 251]. For these reasons the use of GAs is considered in this dissertation within the context of WDS design.

In fact, GAs have become extremely popular for use in WDS design and rehabilitation. They were successfully applied by Goldberg and Kuo [105] in 1987, Walters and Lohbeck [253] in 1993, Simpson and Murphy [178] in 1992, Simpson *et al.* [218] in 1994, Dandy *et al.* [51] in 1996, Halhal *et al.* [116] in 1997, Savic and Walters [209] in 1997, Wu and Simpson [262] in 2001, Wu *et al.* in 2001 [264] and 2002 [265].

3.7.7 Ant Colony Optimisation

Ant colony optimisation (ACO) is a discrete combinatorial optimisation metaheuristic inspired by the observation of the foraging behaviour of real ant colonies. It was first suggested by Dorigo and his colleagues in the early 1990s, and the generic metaheuristic formulation was proposed in 1999 [168]. ACO algorithms (ACOAs) have been used to solve various benchmark instances of combinatorial optimisation problems. Numerous variants of the original *Ant System* algorithm by Dorigo *et al.* [70] have been devised, including a class of ACOAs which guarantee convergence to a global optimum.

A colony of ants locates food by sending out foragers who initially explore their surroundings in an essentially random manner. Once a food source is located, the ant returns to the nest, while depositing a chemical substance called *pheromone* on the ground [70, 168]. Other ants can detect these pheromone trails and choose to follow, with higher probability, paths which are marked by greater pheromone concentrations. Ants can reinforce trails by leaving their own pheromones, and may choose alternative paths with a smaller probability [168]. This reinforcement is a positive feedback mechanism which allows more ants to locate food. A negative feedback mechanism is provided by pheromone decay [251]. The exploration of new paths helps to find improved solutions. It should also be noted that shorter paths between destinations will increase in pheromone intensity due to shorter traversal times, allowing more ants to travel along them in a given time period. These simple mechanisms constitute a form of indirect communication which enables the ant colony to solve the difficult problem of finding a shortest path to a food source. The seemingly intelligent behaviour that emerges from a group of social insects working together is referred to as *swarm intelligence* [168, 251]. A simple example of how the ants find a shortest path to a food source is presented in Appendix C.2.

The first applications of ACO were to the *traveling salesman problem* by Dorigo and Di Caro [70] in 1999. The heuristic is particularly suited to routing and networking problems which involve finding shortest or most efficient paths [251]. It has since been applied to many difficult combinatorial optimisation problems [71]. Ant colony optimisation was applied to WDS design by Simpson *et al.* [219] in 2001, Maier *et al.* [168] in 2003, and Zecchin *et al.* [274, 275] in 2005 and 2007.

ACO is similar to GAs in that both rely on generating a population of solutions using nature inspired stochastic models. The major difference between the two is that the memory (evolutionary information) of the ACOA is associated with the environment itself in the form of pheromone concentrations, whereas the GA stores its evolutionary information intrinsically in the current population of solutions [168].

In general, ant colony algorithms make use of a parameterized probabilistic model known as the *pheromone model*. This employs a vector of model parameters, $\boldsymbol{\mathcal{T}}$, which typically represents the components of a model solution (such as pipe sizes), each associated with a pheromone trail. These pheromone trail parameters $\mathcal{T}_i \in \boldsymbol{\mathcal{T}}$ have variable pheromone values $\widehat{\tau}_i$, which are directly related to the probability of choosing a particular component when constructing a solution [71].

An ACOA employs a colony of artificial ants cooperating to find a solution to a discrete optimisation problem. These ants adopt a stochastic decision-making policy using the pheromone model which stores local information about the environment. These model ants typically incorporate features not found in their natural counterparts, including memory, sight and a discrete time model [168, 274]. The optimisation search proceeds in iterations. Each iteration consists of N cycles whereby each of N individual agents (ants) constructs a single solution by sampling, using the pheromone vector $\boldsymbol{\mathcal{T}}$. The pheromone vector is updated only once an iteration has

been completed [274].

To implement an ACOA, the combinatorial optimisation problem in question must be mapped to a graph $G(\boldsymbol{x}, \mathcal{L}, \mathcal{C})$, where $\boldsymbol{x} = \{x_1, x_2, \ldots, x_\kappa\}$ is a vector of κ decision points, $\mathcal{L}_i = \left\{l_{i(j)}\right\}$ is a set of options (j is the option index) available at the decision points $i = 1, \ldots, \kappa$, and $\mathcal{C} = \left\{c_{i(j)}\right\}$ is the set of costs associated with each option at each decision point. A finite set of constraints, $\mathcal{V}(\boldsymbol{x}, \mathcal{L})$, may be assigned over the elements of $\boldsymbol{x}$ and $\mathcal{L}$. A feasible path over G is any solution $\boldsymbol{x}$, which is a member of the solution space $\mathcal{S}$, including an optimal solution $\boldsymbol{x}^*$. The cost of a solution is measured by means of a cost function, $C(\boldsymbol{x})$ so that $C(\boldsymbol{x}^*) < C(\boldsymbol{x})$ for all $\boldsymbol{x} \in \mathcal{S} \setminus \boldsymbol{x}^*$ [71, 168].

It is important to note that ACO cannot deal with constraints explicitly so that, if they are present, algorithms must be adapted to include penalty functions which penalize solutions outside of the feasible region. This may actually be a benefit in some cases, since it allows more flexible movement through the solution space. There are many different forms that penalty functions can assume, which may account for the number of constraints violated, the degree of violation, and extent of the search [274].

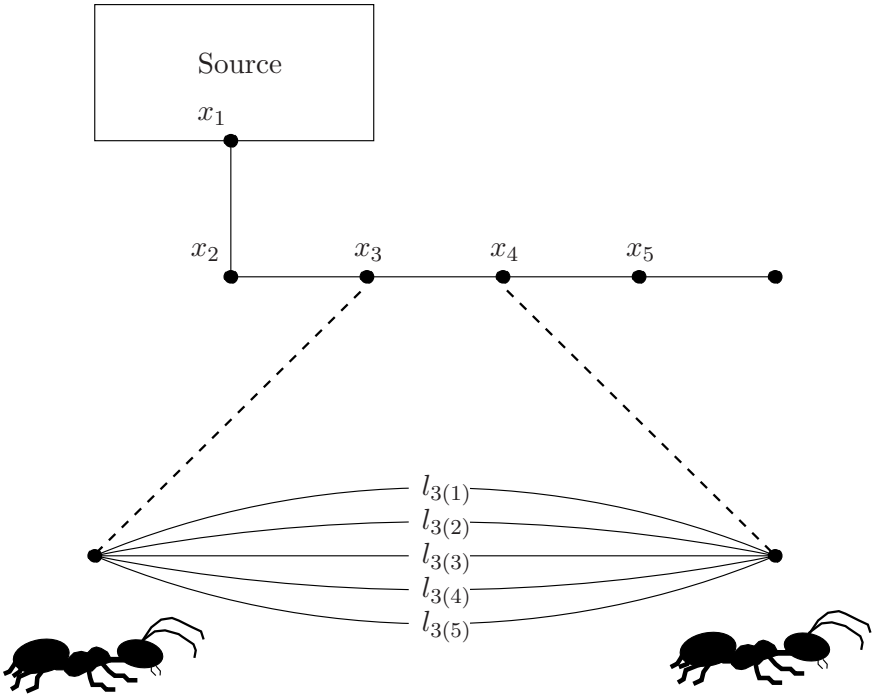


Figure 3.3: *Ant colony optimisation applied to a simple water network.*

Figure 3.3 is a schematic showing how ACO may be visualized with regards to WDSDO. Each pipe has a decision point associated with it. Decision point x_3 has been expanded to show five pipe options $l_{3(1)} - l_{3(5)}$. These potential pipes are associated with their respective costs and pheromone intensity values. The cost function would therefore be $C(\boldsymbol{x}) = \sum_{i=1}^{n} C_{(i,j)\in\boldsymbol{x}} L_i$, where $C_{(i,j)\in\boldsymbol{x}}$ is the cost per unit length for chosen option j at pipe i, and L_i is the length of pipe i.

In 2003, Maier *et al.* [168] developed a problem formulation for applying the Ant System algorithm to WDSDO. This was improved upon by Zecchin *et al.* [274] in 2005 who conducted an extensive parametric study to determine guidelines for assigning values to the various parameters in the algorithm specifically for WDS design optimisation. A pseudocode listing of the adapted formulation may be found in Algorithm 4.

In step 1 of Algorithm 4 the pheromone vector elements are initialized to a base value $\widehat{\tau}_0$. This affects the relative importance of pheromone additions, especially during the early stages of algorithm execution. It is advised that $\widehat{\tau}_0$ should be proportional to an average or 'best' update amount. In steps 2 and 3, the ants take turns to construct sample solutions by traversing the network and stochastically selecting options at each decision point based on the pheromone vector concentrations and a local heuristic factor which favours cheaper options. The relative importance of these are controlled by exponent parameters α and β, whose values must be determined empirically. In step 4, the total cost, C_{T}, of each solution is computed based on physical cost and a penalty cost, which uses as input the pressure deficits at the nodes obtained using a hydraulic simulator. The penalty function is typically zero if the solution is feasible, or equal to the maximum pressure deviation across all the nodes multiplied by a user-defined penalty factor, p_{f}, possibly representing cost per meter of head violation. That is, $P(\boldsymbol{x}) = p_{\mathrm{f}} \times \max_{i=1,\ldots,n} \{h_{\min,i} - h_i(\boldsymbol{x}), 0\}$, where n is the number of nodes in the network, $h_{\min,i}$ is the pressure requirement at node i, and $h_i(\boldsymbol{x})$ is the simulated pressure at node i for solution $\boldsymbol{x}$. Step 5 involves updating the pheromone vector, whereby a pheromone decay process reduces concentrations evenly using a *pheromone persistence factor* ρ_{e}, and the most recently generated solutions contribute to the deposition of additional pheromone using a function with *pheromone reward factor* Ψ.

This mechanism is designed to ensure that lower cost solutions obtain more pheromone, and typically makes use of only the best solution in an iteration. The algorithm continues for a predefined number of iterations or until convergence occurs.

One of the major problems with ACO is that its performance depends heavily on user selected parameters. Based on experimentation with three WDS case studies, Zecchin *et al.* [274] developed deterministic and semi-deterministic expressions for the ACO parameters as summarized in Table 3.1. The value α represents the relative importance of the pheromone concentration of an option, or the 'learned importance' thereof. The value β represents the relative importance of the local cost, or the 'intrinsic desirability' of an option. As $\alpha \to 0$, the importance of the pheromone scheme diminishes and so does the algorithm's performance. For $\alpha > 1$, as α increases, the pressure on convergence increases, resulting in premature convergence. Solution quality is even more sensitive to the value of β. For high values of β (> 1.5), the algorithm was found to be unable to explore the solution space thoroughly enough, because too much emphasis is placed on local costs. The best results were obtained by choosing $\alpha = 1$ and $\beta = 0.5$, suggesting that learned importance is the most critical for the success of the algorithm. The parameter τ_0 was found to be a relatively robust parameter, but an expression was still developed in accordance with the idea that its value should be related to the total potential pheromone additions (as per (3.4)), and obey a simple proportional relationship with respect to Ψ (as demonstrated in Appendix C.2). This yields the expression $\widehat{\tau}_0 = \Psi\sqrt{\kappa \times \overline{n_{\mathrm{opt}}}}/C(\boldsymbol{x}^*)$, where κ denotes the number of decision variables, where $\overline{n_{\mathrm{opt}}}$ denotes the average number of options per decision variable, and where $C(\boldsymbol{x}^*)$ denotes the cost of a near optimum solution (typically some known low cost solution). It was found that higher values of ρ_{e} (as $\rho_{\mathrm{e}} \to 1$) resulted in superior solutions, but slower convergence. However, at exactly 1 there is no pheromone decay and the algorithm does not sufficiently explore the solution space, possibly resulting in an inability to find feasible solutions. A value of $\rho_{\mathrm{e}} = 0.98$ was found to produce the best performance overall. The value of N must be scaled according to the size of the problem. An excellent approximation of the best performing value of N in the three case studies was found to be $N = \kappa\sqrt{\overline{n_{\mathrm{opt}}}}$. The parameter Ψ was found to be a relatively insensitive parameter which did not heavily influence the performance of the algorithm. Setting $\Psi = C(\boldsymbol{x}^{\max})$, where $C(\boldsymbol{x}^{\max})$ is the cost of the most expensive solution, was chosen for the convenience of

Algorithm 4 Ant Colony Algorithm Applied to WDS Optimisation

Input: A WDS network layout with n nodal inputs and associated demands, and minimum pressure requirements, $h_{\min,1}, \ldots, h_{\min,n}$. A set of configuration options for each sizable element. A base pheromone value $\widehat{\tau}_0$, a penalty factor p_f, number of ants N, pheromone exponent α and local cost exponent β controlling their relative importance, a pheromone persistence factor $\rho_\mathrm{e} \in (0, 1)$, a pheromone reward factor Ψ, and the maximum number of generations, $G_{\max}$.

Output: A converged pheromone vector and design solution which is an approximation of a global least-cost design satisfying the pressure requirements.

1: Initialize the pheromone vector elements with the value $\widehat{\tau}_0$. Set iteration counter $t \leftarrow 0$.

2: The N ants are allowed to enter the network consecutively and each construct a single trial solution by traversing the network.

3: Each ant constructs a solution incrementally by sampling an option at each of n decision points using probabilities derived from the pheromone vector. The probability that option j is selected at decision point i at the k^{th} cycle of the t^{th} iteration is given by

$$p_{i(j)}(k,t) = \frac{\left[\widehat{\tau}_{i(j)}(t)\right]^{\alpha}\left[\zeta_{i(j)}(t)\right]^{\beta}}{\sum_{l_{i(j)}\in\mathcal{L}_i}\left[\widehat{\tau}_{i(j)}(t)\right]^{\alpha}\left[\zeta_{i(j)}(t)\right]^{\beta}}, \tag{3.3}$$

where $\widehat{\tau}_{i(j)}$ is the concentration of pheromone associated with option $l_{i(j)}$ at iteration t, $\zeta_{i(j)} = l/c_{i(j)}$ is a heuristic factor favoring local options that have a smaller cost, and α and β are exponent parameters which control the relative importance of pheromone and the local heuristic factor, respectively.

4: The solution is then evaluated by a hydraulic simulator and its cost computed. Total cost, C_T, is the sum of the total component cost, $C(\boldsymbol{x})$, and a penalty function, $P(\boldsymbol{x})$, derived using the maximum pressure deviation from $h_{i,\min}$ requirements, $C_\mathrm{T} = C(\boldsymbol{x}) + P(\boldsymbol{x})$. The process of an ant generating a solution and calculating its cost is called a 'cycle'.

5: Update the pheromone vector after the completion of one iteration (t), of N cycles. The update equation has the general form $\widehat{\tau}_{i(j)}(t+1) = \rho_\mathrm{e}\widehat{\tau}_{i(j)}(t) + \Delta\widehat{\tau}_{i(j)}(t)$, where ρ_e is the pheromone persistence factor < 1 which models the pheromone decay process (and assists in avoiding premature convergence), and $\Delta\widehat{\tau}_{i(j)}$ is the change in pheromone concentration associated with option $l_{i(j)}$ as a function of the solutions found in iteration t. A common method to compute $\Delta\widehat{\tau}_{i(j)}(t)$ is known as the *iteration best* method, whereby

$$\Delta\widehat{\tau}_{i(j)}(t) = \begin{cases} \frac{\Psi}{C_\mathrm{T}(\boldsymbol{x}^t_{\text{best}})} & \text{if edge } (i,j) \in \boldsymbol{x}^t_{\text{best}} \\ 0 & \text{otherwise,} \end{cases} \tag{3.4}$$

where Ψ is the *pheromone reward factor*, ideally chosen so that an infeasible solution is guaranteed to be more expensive than a feasible one, and $\boldsymbol{x}^t_{\text{best}}$ is the best solution in terms of lowest cost found in iteration t.

6: Set $t \leftarrow t + 1$. Stop if the t reaches the predefined maximum, $G_{\max}$, or the solution has converged. Otherwise, repeat from step 2.

automatically scaling the reward factor to within the magnitude of the problem network costs. The value of the penalty factor, p_f, was assigned to ensure that all infeasible solutions would be more expensive than the most expensive solution, $C(\boldsymbol{x}^{\max})$, guaranteeing that they would not be selected as locally optimal solutions. To facilitate this, p_f may be defined such that the smallest total network cost is $C_\mathrm{T}(\boldsymbol{x}^{\min}) = C(\boldsymbol{x}^{\min}) + P(\boldsymbol{x}^{\min}) = C(\boldsymbol{x}^{\min}) + p_\mathrm{f}d$, where d is an

assumed maximum pressure deficit, and is equal to $C(\boldsymbol{x}^{\max})$. That is, $C(\boldsymbol{x}^{\min})+p_{\mathrm{f}}d = C(\boldsymbol{x}^{\max})$. Therefore $p_{\mathrm{f}} = \left[C(\boldsymbol{x}^{\max}) - C(\boldsymbol{x}^{\min})\right]/d$. Now, if d is chosen using a very conservative estimate for the particular case study (Zecchin *et al.* used $d = 0.01$m in all their cases), then infeasible solutions should always be more expensive than feasible ones [274, 275].

Parameter	**Description**	**Value**
α	Exponent for pheromone values	1
β	Exponent for desirability values	0.5
$\widehat{\tau}_0$	Initial pheromone value	$\Psi\sqrt{\kappa.\overline{n_{\mathrm{opt}}}}/C(\boldsymbol{x}^*)$
ρ_{e}	Pheromone persistence factor	0.98
N	Number of ants	$\kappa\sqrt{\overline{n_{\mathrm{opt}}}}$
Ψ	Pheromone reward factor	$C(\boldsymbol{x}^{\max})$
p_{f}	Penalty factor	$\left[C(\boldsymbol{x}^{\max}) - C(\boldsymbol{x}^{\min})\right]/d$

Table 3.1: *Parameter guidelines for Ant Colony Optimisation [274].*

Algorithms developed from the ACO metaheuristic include the original *Ant System* (AS) and the $\mathrm{AS}_{\mathrm{elite}}$ algorithms, devised by Dorigo *et al.* [70] in 1996, the *Ant Colony System* (ACS) by Dorigo and Gambardella [72] in 1997, the $\mathrm{AS}_{\mathrm{rank}}$ algorithm by Bullnheimer *et al.* [23] in 1999, and the *Min-Max Ant System* (MMAS) by Stützle and Hoos [222] in 2000. In 2007 Zecchin *et al.* [275] conducted a comparative study of five different ant colony algorithms applied to four case study WDSs. These sample networks included the two-reservoir problem (TRP) introduced by Simpson *et al.* [218] in 1994, the New York tunnels problem (NYTP) first proposed by Schaake & Lai [212] in 1969, the Hanoi problem (HP) originally considered by Fujiwara & Khang [95] in 1990, and the doubled New York tunnels problem (2-NYTP) devised by Zecchin *et al.* [274] in 2005. Their results showed that the $\mathrm{AS}_{\mathrm{rank}}$ and MMAS algorithms produced consistently superior performance for all the case studies, with MMAS being the slower but stronger algorithm in terms of solution quality [275].

One significant disadvantage of ACO is the large number of parameters involved, which requires extensive parameter tuning, and makes the method impractical for rapid application to real-world problems. Finally, it is interesting to note the similarity between ACO and *Estimation of Distribution Algorithms* (EDAs), both of which build a probability model of variable distributions. Unlike ACO, EDAs are population-based algorithms which rely on pure statistics. EDAs shall be considered further in Chapter 5.

3.7.8 Shuffled Complex Evolution

The *Shuffled Complex Evolution* (SCE) metaheuristic was developed by Duan *et al.* [75] in 1992 as a means to efficiently calibrate rainfall-runoff models. SCE uses a combination of probabilistic and deterministic operations. It employs the systematic evolution of a complex of points, based on a competitive evolution concept (proposed by Holland [123] in 1992). Finally, it makes use of *complex shuffling*, a technique which allows information interchange between different solution families (complexes) [76, 162]. Although it was developed for continuous optimisation problems, it may be adapted for discrete problems by rounding a decision variable value to the nearest discrete value [80].

The algorithm is initiated with a random population of N points from the feasible space. The points are sorted in order of increasing fitness, calculated by means of an objective function.

The population is then partitioned into several *complexes* ($\widehat{z}$ complexes each consisting of $\widehat{w}$ solutions). Each complex is allowed to evolve independently to search the solution space in different directions. Each individual point in a complex has the potential of participating in the reproduction process to obtain new points. For each complex, points are selected with a probability based on individual point fitness, to form a *subcomplex* consisting of o individuals. The modified Nelder and Mead Simplex Method (NMSM) [181] is then applied to each subcomplex. Any improved solution point replaces the point with the worst performance in the simplex. Thereafter solutions in the evolved complexes are pooled together, sorted, shuffled and reassigned to new complexes in order to enable information sharing. This process is repeated until some stopping criterion is met [162].

Liong and Atiquzzaman [162] applied Shuffled Complex Evolution to least-cost optimisation of pipe systems, using a penalty term C_1 devised by Abebe and Solomatine [2] in 1998, to penalize nodal pressure below specified minimum values. If the pressure is below the minimum limit, $h_{\min}$, and greater than zero, the penalty function is

$$C_1 = p_{\mathrm{f}} \times C_{\max} \times \max\{h_{\min} - h_i\}, \qquad \text{for } i = 1, \ldots, n,$$

where p_{f} is the penalty function multiplier used to ensure a smooth transition from feasibility to infeasibility, where $C_{\max}$ is the maximum possible network cost using the most expensive components only, and where $\max\{h_{\min} - h_i\}$ is the maximum of the pressure deviations for the n nodes in the network. If the pressure is below zero, the penalty function changes to

$$C_1 = 2\left(p_{\mathrm{f}} \times C_{\max} - C_{\min}\right),$$

where $C_{\min}$ is the current minimum network cost.

A pseudocode listing of the SCE algorithm is presented in Algorithm 5 (taken from Liong and Atiquzzaman [162] with some minor adaptations from [184]). The algorithm includes a *competitive complex evolution* sub-procedure, which may be visualized as a local search within the complex. A pseudocode listing for this sub-procedure is presented in Algorithm 6.

In 1994 Duan *et al.* [77] compiled a report on the proper selection of the SCE optimisation parameters and recommended the following identities: $\widehat{w} = 2\varphi + 1$, $o = \varphi + 1$, $\alpha = 1$, $\beta = \widehat{w}$ and $\widehat{z} = \widehat{z}_{\min}$, where φ is the number of parameters being estimated. This leaves only the selection of the number of complexes, $\widehat{z}$, which depends on the dimensionality of the problem.

Liong and Atiquzzaman [162] applied the SCE algorithm in 2004 to the design optimisation of two test networks and professed categorically better performance in terms of computational speed compared to previous methods. However, they did not describe their discretization scheme and the author was unable to replicate their results. For this reason SCE is not employed in this dissertation. SCE was used by Nunoo and Mrawira [184] in 2004 for infrastructure works programming to maximize the cost-effectiveness of maintenance (benefits divided by cost) over a period of time. They listed the benefits of the algorithm as:

1. The ability to achieve rapid global convergence in the presence of multiple local optima,
2. The robustness of the search against local traps in "pits" and "bumps" in the objective surface,
3. Robustness against differing parameter sensitivity or parameter interdependence,
4. The ability to model a problem with high parameter dimensionality [184].

Algorithm 5 Shuffled Complex Evolution Algorithm Applied to WDS Optimisation

Input: A combinatorial optimisation problem specification including a set of options for each decision variable. An initial configuration $\boldsymbol{x}_1$, a population size $N = \widehat{z} \times \widehat{w}$, where $\widehat{z}$ is the number of complexes and $\widehat{w}$ ($\widehat{w} \geq 2$) is the number of individual solutions in each complex, a cost function and penalty factor, and a minimum number of complexes $\widehat{z}_{\min}$.

Output: A converged population of solutions which contains an approximation of a global optimum to the combinatorial optimisation problem.

1: Randomly generate an initial solution population of size $N = \widehat{z} \times \widehat{w}$. The population is therefore $\mathbf{P} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_N\}$, where $\boldsymbol{x}_i$ is the i^{th} solution.
2: Compute the basic cost, C, of each point in the population.
3: Perform a hydraulic analysis of each solution. Compute nodal pressure head deficits and note the maximum pressure deficit.
4: If necessary, compute the penalty cost, C_1, using the maximum pressure deficit.
5: For each solution, calculate the total network cost as the sum of the base and penalty cost.
6: The total cost, $f_i = C + C_1$, is used as a measure of fitness — the lower the better [162]. Sort the N solution pairs $(\boldsymbol{x}_i, f_i)$ in order of decreasing fitness [184] (renumber the subscripts to reflect this new order).
7: Partition the sorted points into p complexes, $\mathbf{A}^1, \mathbf{A}^2, \ldots, \mathbf{A}^{\widehat{z}}$, of size $\widehat{w}$ each. This is achieved by letting $\mathbf{A}^k = \left\{(\boldsymbol{x}_j^k, f_j^k) \mid \boldsymbol{x}_j^k = \boldsymbol{x}_{k+\widehat{z}(j-1)}, f_j^k = f_{k+\widehat{z}(j-1)}, \quad j = 1, \ldots, \widehat{w}\right\}$ [184].
8: Evolve each complex according to the *competitive complex evolution* process (described in Algorithm 5).
9: Recombine the complexes into a single population. At this stage complexes may be discarded by excluding the one with the lowest total fitness and setting $\widehat{z}$ equal to $\widehat{z} - 1$, with $\widehat{z} \geq \widehat{z}_{\min}$ [184].
10: Stop if the relative change in the fitness values within the last j ($10 \leq j \leq 15$) shuffling loops is less than a pre-specified tolerance, or if the maximum user-specified number of function evaluations is reached;
11: Otherwise repeat from step 6 with the new population.

3.7.9 Particle Swarm Optimisation

Particle swarm optimisation (PSO) is a metaheuristic inspired by the flocking behaviour of birds and insect swarms. Kennedy and Eberhart [144] proposed the original PSO algorithm in 1995, and it has steadily gained popularity, owing to its features of robustness and rapid convergence. Although PSO was originally developed for continuous optimisation, it may be adapted for discrete optimisation [132].

In PSO, individual solutions in a population are treated as particles flying through the decision space, each associated with a current *velocity* $\boldsymbol{v}$, a memory of its previous *personal best position* $\boldsymbol{p}_{\text{best}}$, knowledge of the *global best position* $\boldsymbol{g}_{\text{best}}$ and, in some cases, a local best position $\boldsymbol{l}_{\text{best}}$, within some neighbourhood — defined either in terms of euclidian distance in decision or objective space, or by some *neighbourhood topology*. Particles are initialized with a random velocity at a random starting position. For particle i at position $\boldsymbol{x}_i^t$ during iteration t, velocity and position are updated as

$$\boldsymbol{v}_i^{t+1} = w\boldsymbol{v}_i^t + c_1 u_1(\boldsymbol{p}_{i,\text{best}} - \boldsymbol{x}_i^t) + c_2 u_2(\boldsymbol{g}_{\text{best}} - \boldsymbol{x}_t^i), \quad \text{and} \tag{3.5}$$

$$\boldsymbol{x}_i^{t+1} = \boldsymbol{x}_t^i + \boldsymbol{v}_i^{t+1} + u_3 \boldsymbol{x}_i^t \tag{3.6}$$

respectively, where w denotes the *inertial weight*, controlling the effect of a particle's previous

Algorithm 6 Competitive Complex Evolution Sub-algorithm

Input: A number of complexes each containing $\widehat{w}$ individual solutions, number of offspring per evolution G_α, and number of evolutions per complex shuffling, G_β. The domain set of the feasible region $\mathcal{F}$.

Output: The set of modified complexes which have evolved according to a mixture of deterministic and stochastic rules.

1: For each complex, select o points ($2 \leq o \leq w$) from the complex based on a *triangular probability distribution* to construct a sub-complex. This has the form $P(\boldsymbol{y}_k = \boldsymbol{x}_i^k) = (p_i) = (2(w+1-i)/(w(w+1)), \quad \text{for } i = 1, \ldots, w$, where $\boldsymbol{y}_k$ represents a point selected for the k^{th} sub-complex. Note that this gives the maximum probability of selection to the point with the greatest fitness $p_1 = 2/(w+1)$, and the minimum probability of selection to the point with the worst fitness $p_w = 2/(w(w+1))$.
2: Sort the sub-complex in decreasing order of fitness.
3: Compute the centroid of the sub-complex, $\boldsymbol{g} = \sum_{i=1}^{o-1} \boldsymbol{y}_i$, excluding the worst point, $\boldsymbol{y}_p$.
4: Generate a new point by reflecting $\boldsymbol{y}_p$ through the centroid, $\boldsymbol{u} = 2\boldsymbol{g} - \boldsymbol{y}_p$.
5: **if** $\boldsymbol{u} \in \mathcal{F}$ **then**
6: Compute its fitness, f_u.
7: **else**
8: Calculate the smallest hypercube containing A^k and randomly generate a point, $\boldsymbol{y}_\text{r}$, within this space and compute its fitness, f_{y_r}. Set $\boldsymbol{u} = \boldsymbol{y}_\text{r}$ and $f_u = f_{y_\text{r}}$.
9: **end if**
10: **if** $f_u < f_{y_p}$ **then**
11: Set $\boldsymbol{y}_p = \boldsymbol{u}$.
12: **else**
13: Compute a contraction point which is halfway between the centroid and the worst point, $\mathbf{c} = \frac{1}{2}(\boldsymbol{g} + \boldsymbol{y}_p)$ and calculate its fitness f_c.
14: **if** $f_c < f_{y_p}$ **then**
15: Replace $\boldsymbol{y}_p$ with $\boldsymbol{c}$.
16: **else**
17: Randomly generate a point $\boldsymbol{y}_\text{r}$ in $\mathcal{F}$, compute f_{y_r} and replace $\boldsymbol{y}_p$ with $\boldsymbol{y}_\text{r}$. This may be seen as a mutation step.
18: **end if**
19: **end if**
20: Repeat steps 3 to 19 G_α times. This may be seen as the number of offspring which are generated. The new offspring replace the original members of the sub-complex.
21: Repeat steps 1 to 20 G_β times. This may be seen as the number of evolutions which occur for each complex per complex shuffling.

velocity, where c_1 and c_2 are the *learning factors* for *cognitive* and *social* learning respectively, where $u_1, u_2 \in [0, 1]$ are uniform random variables, and where $u_3 \in [-\frac{1}{2}, \frac{1}{2}]$ is a uniform random variable controlling turbulence [244]. A pseudocode listing for the basic PSO algorithm appears in Algorithm 7.

Izquierdo *et al.* [132] applied PSO to WDSDO in 2008. They developed an adaptation of the original algorithm whereby solution collisions (a problem that occurs frequently in PSO) are checked using several of the fittest particles, and any colliding solutions are randomly regenerated with a new position and velocity. This adaptation greatly improves population diversity and global convergence characteristics. Furthermore, they employed an adaptive inertial weight of the form $w = 0.5 + \frac{1}{(2\ln(t)+1)}$, where t is the generation number. This has the effect of ac-

Algorithm 7 Particle Swarm Optimisation Algorithm

Input: A combinatorial optimisation problem with a domain set for each decision variable, an inertial weight w, a cognitive learning factor c_1, and a social learning factor c_2, minimum and maximum velocity values $v_{\min}$ and $v_{\max}$, and a fitness function $f(x)$ providing the fitness of configuration $\boldsymbol{x}$, a population size N, a maximum number of generations $G_{\max}$

Output: An approximation of a globally optimal solution to the combinatorial optimisation problem.

1: Set $t \leftarrow 0$.
2: Initialize a random population P of N solutions $\boldsymbol{x}_i^0$, with random initial velocities $\boldsymbol{v}_i^0 \in [v_{\min}, v_{\max}], \quad i = 1, \ldots, N$.
3: Set the best position of each solution $\boldsymbol{p}_{i,\text{best}}^0 = \boldsymbol{x}_i^0$ and set $\boldsymbol{g}_{\text{best}} = \boldsymbol{x}_1^0$
4: **for** each $\boldsymbol{x}_i^t \in P$ **do**
5: Calculate the fitness $f(\boldsymbol{x}_i^t)$ and set $\boldsymbol{g}_{\text{best}} = \boldsymbol{x}_i^t$ if $f(\boldsymbol{x}_i^t) > f(\boldsymbol{g}_{\text{best}})$.
6: Set $\boldsymbol{p}_{i,\text{best}} = \boldsymbol{x}_i^t$ if $f(\boldsymbol{x}_i^t) > f(\boldsymbol{p}_{i,\text{best}})$.
7: **end for**
8: **for** each $\boldsymbol{x}_i^t \in P$ **do**
9: Generate random numbers $u_1, u_2 \in [0, 1]$ and $u_3 \in [-\frac{1}{2}, \frac{1}{2}]$.
10: Set $\boldsymbol{v}_i^{t+1} = w\boldsymbol{v}_i^t + c_1 u_1(\boldsymbol{p}_{i,\text{best}} - \boldsymbol{x}_i^t) + c_2 u_2(\boldsymbol{g}_{\text{best}} - \boldsymbol{x}_i^t)$
11: Set $\boldsymbol{x}_i^{t+1} = \boldsymbol{x}_i^t + \boldsymbol{v}_i^{t+1} + u_3\boldsymbol{x}_i^t$
12: **end for**
13: Set $t \leftarrow t + 1$.
14: **if** $t \geq G_{\max}$ **then**
15: Repeat from step 4.
16: **end if**

celerating the search in the initial stages, and gradually reducing the importance of the current velocity, ideally such that the particle converges to an optimum. They also used values of $c_1 = 3$ and $c_2 = 2$. Finally, they adapted the algorithm to accomodate discrete variables by discretizing the velocities in order to create discrete step trajectories for these variables. Izquierdo *et al.* [132] tested their algorithm on the NYTUN and HANOI WDS benchmarks, and achieved large computational savings (an order of magnitude better than previous methods) whilst closely approximating known global optimum solutions.

PSO is employed in later chapters of this dissertation. However, it requires adaptation in order to be used for multi-objective optimisation. A multi-objective version of PSO shall be presented in Chapter 5.

3.7.10 Shuffled Frog Leaping Algorithm

The *Shuffled Frog Leaping Algorithm* (SLFA) is a *memetic* metaheuristic proposed by Eusuff and Lansey in 2003 [83]. Such metaheuristics are based on an extension of genetic evolutionary concepts which include the analogy of social *memes*. Memes was first conceptualized by the biologist Richard Dawkins in his book *The Selfish Gene* [54]. They may be defined as social ideas or behaviours which are passed on laterally through a population (that is, in theory, from any individual to any other individual), as opposed to only from parent to child, and may be thought of as a unit of cultural evolution. The SLFA very closely resembles the Shuffled Complex Evolution algorithm in structure, except for the *memetic evolutionary phase* which replaces the modified Simplex algorithm, and the fact that it is specialized for discrete optimisation. This

memetic evolutionary phase is similar in concept to Particle Swarm Optimisation [83].

SFLA develops from the analogy of a population of frogs leaping about in a swamp in order to locate food, and communicating with each other about their success. The frogs leap onto rocks positioned at discrete locations. The frog in the worst position takes a leap in the direction of the frog in the best position, and this continues for a specified number of frogs. The concepts of a complex and sub-complex are replaced with the similar concepts of a *memeplex* and *sub-memeplex* which may be thought of as different communities of frogs. Memeplexes are generated in an identical fashion to the complexes of the SCE algorithm. The various memeplexes also undergo shuffling at the end of each evolutionary phase, in an analogy of exchanging information across cultures [83].

Algorithm 8 Competitive Memeplex Evolution Sub-algorithm

Input: A number of memeplexes each containing $\widehat{w}$ individual solutions. Number of required evolutions G_α, the domain set of the feasible region $\mathcal{F}$.

Output: A set of modified memeplexes which have evolved according to a mixture of deterministic and stochastic rules.

1: For each memeplex, select o points from the memeplex based on the *triangular probability distribution* used in SCE.
2: The sub-memeplex is sorted in decreasing order of fitness, stored in the vector $\boldsymbol{y}$, with $\boldsymbol{y}(1)$ and $\boldsymbol{y}(o)$ being the positions of the best and worst frog respectively.
3: The goal of a frog is to improve its individual meme by learning from the best frog in the sub-memeplex, or from the best frog globally. Record the positions of the best and worst frog as $\boldsymbol{x}_B = \boldsymbol{y}(1)$ and $\boldsymbol{x}_W = \boldsymbol{y}(o)$ respectively.
4: Update the position of the worst frog. The step size is calculated as $\boldsymbol{s} = \min[\text{int}(u \times (\boldsymbol{x}_B - \boldsymbol{x}_W)), d_{\max}]$ for a positive step and $\boldsymbol{s} = \max[\text{int}(u \times (\boldsymbol{x}_B - \boldsymbol{x}_W)), -d_{\max}]$ for a negative step, where u is a random variable $\in [0, 1]$ and $d_{\max}$ is the maximum step size allowed (could be the maximum incremental change in pipe diameter value in the context of WDS optimisation). The new position for the worst frog becomes $\boldsymbol{y}(o) = \boldsymbol{x}_W + \boldsymbol{s}$.
5: **if** $\boldsymbol{y}(o) \in \mathcal{F}$ **then**
6: Compute its fitness, $f_{\boldsymbol{y}o}$.
7: **else**
8: Go to step 13.
9: **end if**
10: **if** the new value $f_{\boldsymbol{y}o}$ is better than the old $f_{\boldsymbol{y}o}$ value **then**
11: Replace the old $\boldsymbol{y}(o)$ value with the new $\boldsymbol{y}(o)$ value and go to step 16.
12: **end if**
13: **if** the frog's new position is either infeasible or not better than the old position **then**
14: Stop the spread of defective meme by randomly generating a new frog at some feasible location, $\boldsymbol{u}$, to replace the old frog. Set $\boldsymbol{y}(o) = \boldsymbol{u}$ and compute f_u.
15: **end if**
16: Upgrade the memeplex. After the memetic change for the worst frog in the sub-memeplex, replace $\boldsymbol{y}$ in their original locations in the memeplex and sort it in order of decreasing performance value.
17: Repeat steps 1 to 16 G_α times.

Before each shuffling phase, the position of the globally best frog, $\boldsymbol{x}_B$, is recorded. The *competitive complex evolution* sub-procedure of the SCE is supplanted by the *memetic evolution* sub-procedure, which proceeds as described in Algorithm 7.

In 2003 Eusuff and Lansey [83] tested SFLA on three sample WDSs from the literature and found previously best solutions for the two of them and a very nearly optimal solution for the third. However, SFLA found the optimal solutions in fewer iterations than GAs and simulated annealing [83]. Liong and Atiquzzaman [162] claim to have achieved superior results using the ordinary SCE, however this could not be replicated. The SFLA was developed for single-objective optimisation only, and shall not be used in this dissertation.

3.8 Chapter Summary

A broad introduction to the problem of WDSDO has been provided in this chapter in fulfilment of Objective 3(a) in §1.3 and in partial fulfilment of Objective 2. The WDS design process was presented, and a generic mathematical formulation of the least-cost WDSDO problem was given. This model caters for the design of all standard WDS components, including pipes, tanks, pumps and valves, and allows for constraints in addition to those normally placed on nodal pressure.

Several practical WDS design issues were examined, with the outlook towards achieving a design algorithm catering for real water systems. Some of the topics discussed included the uncertainty of data, the requirement of loops for redundancy, objectives in addition to cost which express the benefits of a design, using extended period analysis to design tanks and pumps, and designing for fire-flows.

A concise history of research into the WDS problem was included, showing how the problem evolved over the years from simple pipe network design to complex multi-objective design of all WDS components, including layout. The single-objective optimisation methods used over the years were described in some detail, including an in-depth look at several of the most important techniques and metaheuristics used to solve the problem in the past. These methods are Partial Enumeration, Linear and Nonlinear Programming, Simulated Annealing, Tabu Search, Genetic Algorithms, Ant Colony Optimisation, Shuffled Complex Evolution, Particle Swarm Optimisation, and the Shuffled Frog Leaping Algorithm.

Chapter 4

Essential Topics in WDS Design

In this chapter, the stage is set for a more realistic WDSDO model. The main topics of this chapter include WDS model parameter uncertainty, water demand estimation for normal operations and extreme events such as fires, the incorporation of operational costs in WDSDO, WDS reliability quantification and network layout design. Secondary topics described include tank design, water quality, transient analysis, leakage models, and valve design.

4.1 The Certainty of Uncertainty

WDSs are subject to a great deal of uncertainty, particularly in terms of water demand placed upon the system, and its temporal variation. Furthermore, uncertainty may exist in terms of actual pipe characteristics, which alter as the system ages, in terms of reservoir water levels, which generally depend on operational practices, and in terms of both changes in physical layout and functional requirements over time [141]. Owing to the great complexity inherent in hydraulic systems, models are always rough approximations of reality. In order to build a model that is robust to input uncertainty, the effects of this uncertainty must the quantified. The tendency in hydraulic engineering is towards deterministic design employing large safety factors and heavily over-designing WDSs [103]. At the other extreme, the least-cost optimisation paradigm produces designs at the limit of feasibility, for which unforeseen events or population growth will result in failure of the WDS. Uncertainty is a common theme in several of the topics in this chapter, and although it shall primarily be considered in terms of demand uncertainty in this dissertation, uncertainty shall also be discussed in the ensuing sections within the context of pipe roughnesses, operational costs, and reservoir levels.

4.2 Demand Estimation

Demand estimation refers to the process of assigning spatially distributed temporal water demand quantities (or *loads*) to new or existing WDSs, and potentially predicting future demands, thus determining the required system capacity for WDS design or rehabilitation. WDSs are subject to *water balance* constraints, which means simply that all input water to the system must equal all water exiting the system (accounting for changes in storage). The International Water Association (IWA) [131] has recommended an international 'best practice' *standard water balance* for use in WDS assessment. A schematic of this water balance appears in Figure 4.1,

demonstrating the broad range of outputs which must add up to the total system input volume. These outputs are what one is interested in estimating.

An estimate of demands must consider (1) ordinary consumer demands (including non-revenue consumers), (2) water losses, such as leakage, unauthorized usage, and metering inaccuracies, and (3) fire flow demands (which usually place the most extreme demands on the system). A simplified water balance constraint for a given period may be expressed as

$$V_{\text{inflow}} = V_{\text{demand}} + V_{\text{losses}} + V_{\text{fireflow}} + \Delta V_{\text{storage}},$$

where V_{inflow} denotes the total input or production volume, where V_{demand} denotes the sum of demands, where V_{losses} denotes the total losses, where V_{fireflow} denotes the total water used for fire-fighting, and where $\Delta V_{\text{storage}}$ denotes the change in storage volumes.

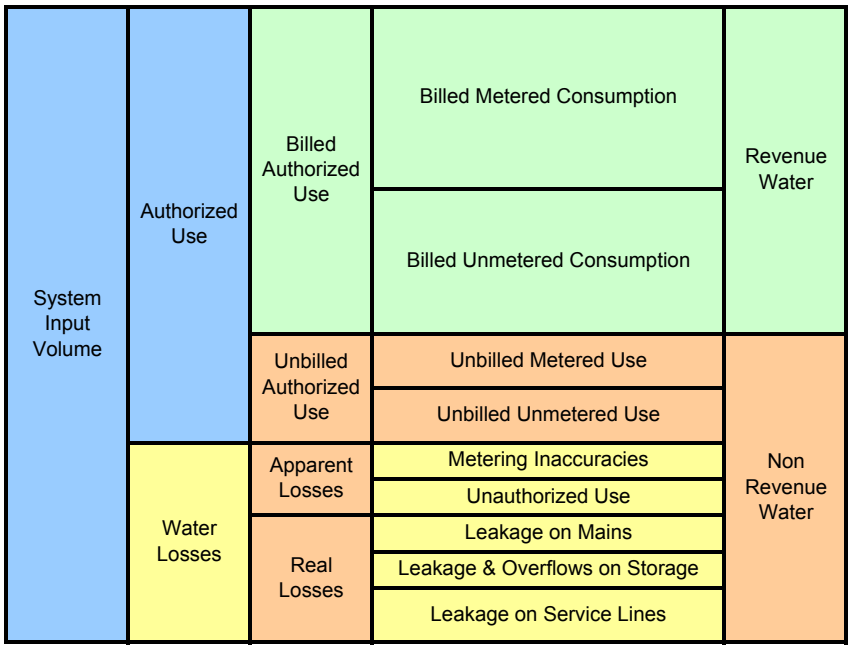


Figure 4.1: *IWA international standard water balance [131].*

4.2.1 Baseline Demands

Baseline demands characterize the expected (mean) value of water demands, and are usually assigned to individual nodes (typically these demands include both consumer demands and water losses). In the case of an existing WDS, demands are allocated to nodes on the basis of historical production and consumption (billing) records. Customer billing records are not usually grouped by node, so that some aggregation method is required (nearest node, pipe analysis, *etc.*). Assigning land parcels to nodes is frequently performed as a GIS[1]-based procedure, with polygons representing land parcels superimposed on a network graph of the WDS, and the allocation of polygons to nodes.

In the case of a new WDS, or the extension of an existing WDS, demands are typically assigned using expected unit demands, based on the *land use type*, facility type, or the number of

[1] A *Geographic Information System* is a software package which allows the storage, manipulation, and advanced querying of spatial and geographic data. This typically includes digital maps and layers of associated meta-data.

customers of a particular type, being served by a node. For example, supposing one knows the average consumption ℓ per person in a particular suburb, and the number of people n being served by a node, then the total average demand at that node is taken as $n\ell$. This technique is also known as *unit loading*. Expected water consumption for different unit types is derived from historical water utility records, frequently summarized in engineering and governmental guidelines. Information regarding the consumption patterns of neighbouring WDSs may also assist in improving the accuracy of demand estimates, and regional guidelines are often compiled for this purpose. Demands at a node may also be composite, in that several different unit types are served by a single node, in which case unit loading must be applied in a proportional manner [251].

In South Africa the *average annual daily demand* (AADD) is used as a baseline, which is then multiplied by *peak-to-average* ratios in order to estimate peak design demands. The most commonly used South African guideline for municipal water-demand estimation appears in the CSIR Red Book [44]. This guideline was first published in 1983, and has not changed significantly since. For domestic-water demand estimation in developed areas it advises AADD based on stand area (single residential stands), and for developing areas it provides per capita estimates for communal water points, stand pipes and yard taps. A 2008 study by Van Zyl *et al.* [242] revealed that the CSIR Red Book guideline requires revision, as it only accounts for 53% of the observed municipal demands in South Africa. They found that the most influential factors affecting water demand are stand area, stand value and geographical location (inland or coastal), but that stand value was too uncertain a descriptor to be used in a fixed guideline. They developed a new guideline curve for AADD based on stand area within various confidence limits [242]. Typical values of AADD for different land-use types appear in Table 4.1, as supplied by *GLS Software*[2] [103], which have been adapted from the *Johannesburg Water* and *Tshwane Metropolitan Municipality* guidelines [103]. Note that these values are expressed both in kiloliters (kl) per plot per day and kl per hectare (ha) per day, and that all estimates include water losses. In order to assign demands to nodes, one may estimate the AADD required from the area of land a node is servicing and its land use type. Baseline hourly demand is then simply AADD/24. The majority of water utilities maintain water usage records for

Land Use	**AADD Demand** (kl/plot/day)	**AADD Demand** (kl/ha/day)
Low cost housing	0.75	13.0
Small sized residential	1.00	15.0
Medium sized residential	1.75	12.0
Large sized residential	2.50	12.0
Cluster housing	0.75	18.0
Cluster housing > 30 units/ha	0.65	25.0
Agricultural holdings	3.00	3.0
Business/Commercial	5.00	20.0
Light Industrial	5.00	20.0
Heavy Industrial	5.00	25.0
Mixed Zoning	3.00	18.0

Table 4.1*: AADD of water by land use types for Gauteng Province, South Africa, supplied by GLS Consulting (Pty) Ltd [103].*

existing systems (usually at least for water production and customer usage) for billing and management purposes. These data are usually not available at the required time scale, so that additional, typically more intensive measurements may be necessary at representative points in

[2] A South African hydraulic engineering consultancy, based in Technopark, Stellenbosch.

the system. Large consumers, such as industries and hospitals, often warrant their own special measurements, particularly if they account for a few percentage points or more of total demand. New technologies allow for real-time and remote data logging, which facilitates significantly improved estimates of baseline demand variation [169]. However, whichever data are gathered, it is rarely sufficient to fully specify the demand loadings for an existing WDS. Losses will generally occur and must still be associated with individual nodes, despite a lack of knowledge regarding their exact location. All water losses must be disaggregated amongst users, frequently done on a uniform basis, or proportional to known nodal demands [251].

4.2.2 Demand Variation

A WDS must be designed to accommodate temporal demand variation. The typical procedure is to begin with baseline demands and apply a series of periodic *demand multipliers* (also known as a *demand pattern*) to mimic cyclic demand variation, or individual demand multipliers to simulate steady state demands, such as peak flow conditions. Fire flows and other special demand scenarios are usually also simulated separately, and demands are sometimes projected into the future for long term planning [251].

Different demand multiplier patterns are sometimes used for nodes serving different zones or land-use types, and sometimes a large consumer may have their own demand pattern. A typical diurnal (daily) pattern for a residential area exhibits relatively low demand at night, and has two demand peaks during daylight hours; a morning peak before working hours, and an evening peak after people return home. Most demand patterns repeat in a cyclic fashion; however, additional variation may be modelled, depending on the WDS.

Extended period analysis is necessary for tank design and pump scheduling, and peak flow analysis is required to design the WDS so that it may accommodate the maximum expected demand loading. The types of demand analyses which are frequently conducted include [251]:

1. *Average-day demand scenario*: The baseline is the average rate (*e.g.* m^3/h) of the average daily demand (*e.g.* annual demand / 365), occurring for a given planning horizon. A series of hourly demand multipliers may be used to simulate diurnal demand variation. This may be used to design the overall WDS for efficient operation on an average day.

2. *Maximum-day demand scenario*: The baseline is the average rate of the maximum expected daily demand, occurring for a given planning horizon. A series of hourly demand multipliers may be used to simulate diurnal demand variation. This may be used to design the overall WDS to cater for a maximum day (especially with regards to storage and pumping). An example of a maximum day factor is twice the AADD as baseline hourly demand [103].

3. *Peak-hour demand scenario*: The demand rate during the maximum hour of usage, occurring for a given planning horizon. This may be used to size pipes in order to handle maximum flow capacities.

Note that these scenarios may account for historical demands only, current demands, or projected future demands (*e.g.* possibly considering a 1, 5, 10, or 20 year projection). Other potential scenarios may include weekly demand patterns (*e.g.* $7 \times 24 = 168$ demand multipliers), maximum historical day demand, and seasonal demand variation (affected by climate and social activities such as tourism and gardening).

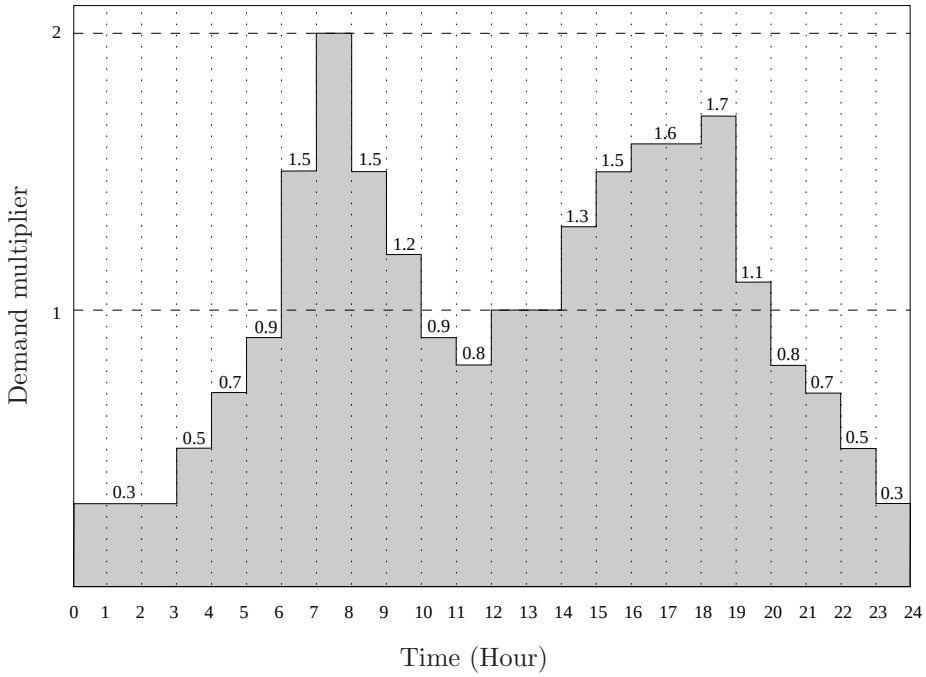


Figure 4.2*: WDS demand multipliers for a typical residential zone (24 hour demand cycle), supplied by GLS Software [103].*

As an example of demand multipliers, one might have a series of 24 demand factors to represent the hourly variation over the course of an average day. Typical hourly demand multipliers in a diurnal cycle for a generic residential zone are provided in Figure 4.2, and similarly typical multipliers for a generic industrial zone appear in Figure 4.3. These multipliers were supplied by GLS Software [103].

Water losses such as leakage may have a relatively constant flow rate. In a system with relatively high losses, it may be necessary to treat this separately from other demands, since it does not vary in the same manner as do other demand types [251]. In this case losses would be subtracted from baseline demands, the demand multipliers would be applied to the remainder of demands, and losses would then be added to the result. However, including water losses is erring on the safe side, since it produces higher effective demand quantities.

Demand patterns vary widely from customer to customer. In a new system, the best one can do is to employ typical patterns for various customer types. In an existing system, accuracy of demand patterns increases with the number of flow metering points that support periodic data-logging [251].

4.2.3 Fire Flow Demands

In the event of a fire, additional flow demands may constitute a large fraction of total demand [251]. Numerous standards exist for determining required quantities of fire flow, typically depending on the size, structure, proximity and construction materials of buildings, materials stored in a building, and on other aspects such as the operational or financial importance of the structures under fire protection. Fire flows are critical in determining the sizes of pipes, particularly for smaller pipes, typically leading to larger minimum sizes.

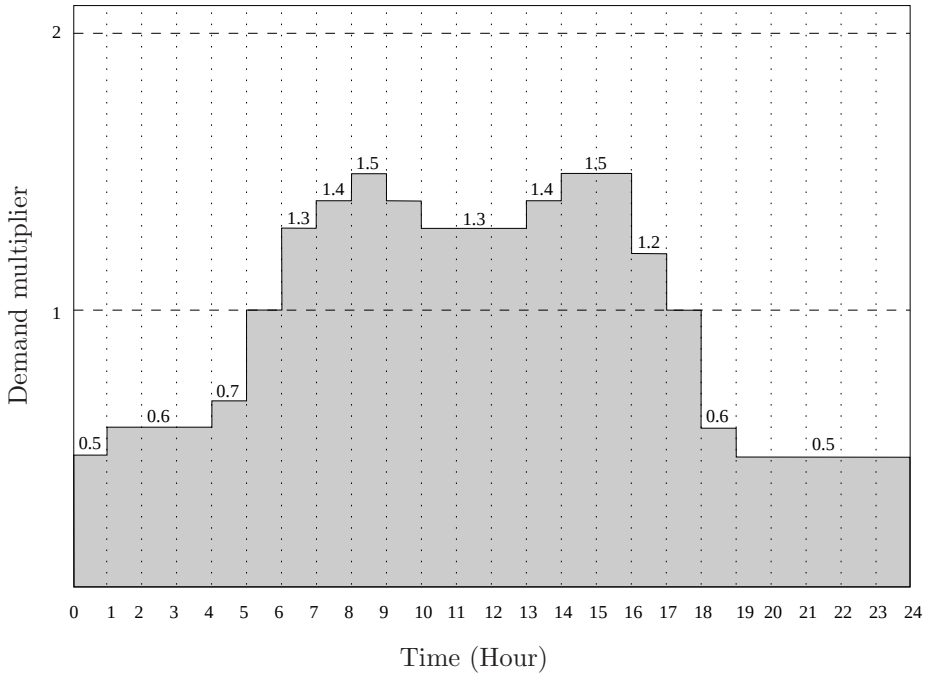


Figure 4.3*: WDS demand multipliers for a typical industrial zone (24 hour demand cycle), supplied by GLS Software [103].*

Fire protection is not required in all WDSs, but the tendency is to include them, particularly in urban areas, where potential fire damage may be extensive. Including fire protection also lowers building insurance rates. The American Insurance Services Office (ISO)[3] evaluates water supply systems on the basis of *Needed Fire Flow* method [129], with different formulas for categories such as residential, commercial or industrial buildings. One such a formula for needed fire flow F_N is

$$F_N = 18K_{\mathrm{C}}A^{0.5}O(X_{\mathrm{f}} + C_{\mathrm{f}}),$$

where K_{C} denotes the *construction class coefficient* of the building, where A is the effective area (ft^2), where O is the *occupancy factor* expressing the nature of the building contents (*e.g.* chemical / combustible), where X_{f} is the *exposure factor* determined by the distance to nearest buildings and their type, and where C_{f} is the *communication factor* which is determined by the locations and types of doors and windows [251]. The method is documented in the ISO *Fire Suppression Rating Schedule* [129] and the *American Water Works Association* (AWWA) *M-31 manual* [14].

In South Africa, more basic guidelines are used to specify expected fire flows by means of location / risk categories. This risk is primarily focused on potential financial damages, risk of spread, loss of life, and damages to critical infrastructure. The guidelines appearing in Table 4.2 constitute a combination of the Johannesburg Water, Tshwane MM guidelines, and *SANS*[4] *100090* fire flow standard, as supplied by GLS Software [103]. These conditions are modelled under a maximum day scenario ($2 \times AADD$ plus the appropriate fire flow according to the area's fire risk assessment), and only one fire at a time is considered per reservoir zone. WDSs for low cost housing and agricultural land do not usually cater for fire flows in South Africa, which is why no guidelines are provided for them.

[3] Not to be confused with the *International Standards Office*.

[4] *South African National Standards* are compiled by the official national standards bureau SABS.

Although it is assumed that the likelihood of multiple fires occurring during a peak day is very small, analysis should be site-specific and may depend on the size of the WDS.

It is unlikely that one would be able to specify a realistic probability distribution for the location and frequency of fire events in a new system. The following represents a novel procedure towards the stochastic automated incorporation of fire flow analysis in the design process. In this procedure a number of candidate nodes may be identified for fire events, for which fire flow demands are added to peak hour demands. The following nodes may be selected for analysis:

1. a node with largest demand rate,
2. a node with lowest pressure,
3. a node at the output end of a smallest pipe,
4. a node at the output end of a pipe with the smallest flow,
5. a node at the output end of a pipe with the largest flow,
6. a node furthest from a source,
7. a node nearest the centroid of the system, and
8. a random node in each different zone type (*e.g.* as per Table 4.2).

In this procedure the same node may be identified more than once, in which case only a single fire scenario is analyzed at that node. Note that, depending on the number of hydrants involved, several adjacent nodes may also experience fire flow simultaneously (assuming a hydrant at each node). Such nodes are to be selected as those nearest the identified node. Owing to the dynamic alteration of designs during WDSDO, and the stochastic elements used in this fire location scheme, different nodes would be selected during the course of optimisation. Even though only a few simulations are conducted per design during a single optimisation iteration (the minimum being one randomly selected scenario per iteration), the WDS designs will be subjected to a wide range of fire scenarios over the course of multiple iterations. The philosophy behind this identification procedure is that WDSs should be robust under all possible fire conditions. An extension to this scheme is the simulation of fire conditions for a number of consecutive periods, as part of a maximum day simulation (in order to determine whether tanks have sufficient fire flow storage). It is further recommended that any special fire conditions be specified manually, particularly through the identification of critical infrastructure which requires fire protection. The testing of this automated fire scheme is left as future work.

Fire location / Risk	Total fire flow (ℓ/s)	Flow at one hydrant (ℓ/s)	Min head at fire node (m)	Min head at other nodes (m)
High risk: CBD and other	200	25	15	10
Med risk 1: Industrial, Business	100	25	15	10
Med risk 2: Cluster Housing, High Rise Flat	30	30	15	10
Low risk: Single Residential Housing	15	15	8	8

Table 4.2*: Fire flow demands for Gauteng Province, South Africa, supplied by GLS Software [103].*

4.2.4 Emergency Storage

Emergency storage is required in the event that flow along the supply mains is interrupted. In this case ordinary demands must be satisfied entirely from emergency storage (assuming that such storage is available). This is discussed in more detail later in this chapter (§4.5.2).

4.2.5 Handling Demand Uncertainty

Only *deterministic demand strategies* have been touched upon thus far in this chapter. Although these are frequently used in practice, they do not account for the high levels of uncertainty present in real WDSs. Babayan *et al.* [16, 17] demonstrated that ignoring uncertainty in parameters (particularly demand) may result in the serious under-design of a network.

Schemes towards incorporating uncertain demands have been based on assigning demand probability distributions to individual nodes (*e.g.* both Xu and Gouler [267] and Babayan *et al.* [16] assumed a *Normal* (*Gaussian*) distribution), and then either sampled from these distributions to simulate demand variation (such as the *reduced sampling methodology* of Kapelan *et al.* [141]), or used analytical techniques (such as the *integration method* of Babayan *et al.* [16] which incorporates safety factors based on some multiple of the nodal head standard deviations (SDs)). These methodologies are also discussed in more detail later in this chapter (§4.3.1). Deterministic mean values of demand were employed in both papers; Babayan *et al.* [16] use a SD equivalent to 10% of the mean.

A serious deficiency of the techniques mentioned previously is the assumption of independent probability density functions. In actual WDSs, nodal demand is highly correlated in time [251]. This calls for some technique to obtain correlated demand samples, either by means of some artificial demand sampling pattern, or some statistical technique to induce correlation by ordering random samples. The latter method is employed in this dissertation towards reliability analysis. Demands were assumed to be normally distributed and temporally correlated at a coefficient of 0.5 between any two nodes (as per Kapelan *et al.* [141]). The method used to yield correlated samples is that of Iman and Conover [128].

4.2.6 Correlated Demands

In 1982 Iman and Conover [128] developed a distribution-free approach to inducing rank correlation among model input variables. This method was used by Kapelan *et al.* [141] in 2005 to obtain temporally correlated, normally distributed nodal demands. The technique works by constructing an $s \times n$ matrix $\mathbf{D}$ of normally sampled demands, where s is the number of demand samples for each node. An equivalently sized Van der Waerden matrix $\mathbf{V}$ is then generated in which each element is sampled from $N(0, 1)$. An $n \times n$ correlation matrix $\mathbf{C}$ is created with the desired correlation coefficient (*e.g.* one's on the main diagonal and 0.5 elsewhere), to which a Cholesky decomposition is applied ($\mathbf{C} = \mathbf{UL}$) in order to obtain the lower triangular Cholesky matrix $\mathbf{L}$. Then a ranking matrix $\mathbf{R}^*$ with the desired correlation coefficient matrix is obtained as $\mathbf{R}^* = \mathbf{VL}$. The samples in $\mathbf{D}$ are finally rearranged column-wise to have the same rank ordering as $\mathbf{R}^*$, producing $\mathbf{D}^*$, a set of demand samples with the desired inter-nodal correlation. Each row of $\mathbf{D}^*$ now represents a single correlated demand loading condition.

4.2.7 Projecting Future Demands

Predicting future water demands is extremely difficult, as the number of factors affecting the outcome is enormous. Demand changes are affected by population growth, economic growth or downturn, climatic changes, social changes (such as water conservation efforts or restrictive legislation), technology improvements (such as the installation of low-water plumbing), changing water tariffs, and so forth. Possibly the best way to accommodate this problem is the consideration of a few alternative future scenarios. One might, for example, consider linear growth, constant percentage growth, growth levelling off to a plateau, and so forth [251].

One technique for analyzing demand growth is employing a GIS, with a polygon layer storing data regarding expected future conditions. After these polygons are assigned to the appropriate nodes, the unit loading method may be applied in order to estimate demands. Such information may be available from government studies or census data regarding predicted population growth and future land-use [251].

Some researchers have developed sophisticated demand projection models, which use disaggregated functions of other economic and social growth variables to predict future water demands. One such model is the 1998 IWR-Main model of Optiz *et al.* [186].

4.3 Reliability: The Other Objective

Reliability is a measure of system performance, generally expressed as the ability of a system to meet the demands placed on it. Reliability may be quantified as the proportion of time that a system functions as intended (also termed *availability*). However, reliability in the context of WDSs is a somewhat nebulous concept, owing to the vast number of different interpretations over the years, and the lack of a universally accepted definition [112]. In a common sense interpretation, one wishes a WDS to be reliable such that *consumers receive desired quantities of water within the required pressure range as often as possible, subject to the full range of possible demand conditions, combined with possible component failure scenarios.* These demand and failure conditions are associated with different probabilities of occurrence and this definition of reliability is often referred to as *hydraulic reliability*, as it is focussed on user hydraulic requirements [4, 141]. Reliability is strongly related to the amount of spare capacity available to meet extreme demands (such as fire-flows) and to pathway redundancy (or loops) which provide alternate flow pathways in the event of pipe failure [268]. The ability of a network to respond gracefully to component failure by means of alternate flow pathways is sometimes termed the *flexibility* of a WDS.

Component failure scenarios usually entail a component being unavailable due to breakage or maintenance, which means that the WDS must continue operating at reduced capacity and may not be able to meet performance requirements. Component failures may include taking components off-line for maintenance, rupture of pipes, joint and valve failures, pump breakdowns, or failure may be caused by externalities such as power failures. A more insidious type of component failure is due to the gradual aging of the system, whereby deposition or corrosion may cause pipe characteristics to deteriorate. Such failure is typically not handled explicitly as a component failure, but is rather accommodated by conducting sensitivity analysis on pipe characteristics. The final type of failure is when the demand loading exceeds the system capacity. This may occur due to extreme events beyond the design capacity, or due to the natural growth of the population or economy. A WDS cannot be expected to perform without a degradation of service during these situations. However, WDSs should be designed to meet appropriate

minimum performance standards during emergencies or system failures [112, 268].

It is obvious that the sizing of WDS components has a considerable effect on the ability of a system to meet its goals. In addition to larger capacity systems being able to satisfy more extreme demand conditions, there is also a strong correlation between pipe size and breakage rates, with smaller pipes breaking more frequently. Component breakage probabilities may be inferred from historical records, and modelled using appropriate probability distributions [112]. Pipe breakages have also been modelled using data-driven techniques such as artificial neural networks and neuro-fuzzy systems, as well as multivariate linear regression models (Tabesh *et al.* 2009). The location of shut-off valves also plays an important role in system reliability, since they directly affect which parts of the system may be isolated, and consequently define the available flow paths [251].

A *probabilistic approach to reliability* best matches the definition of hydraulic reliability (*i.e.* the probability that a design satisfies hydraulic requirements at any given moment). In general, probabilistic models may be solved analytically or by means of stochastic simulation. Analytical solutions are obtained either by conditioning over all possible events (provided that this is computationally feasible and that conditional probabilities are known), or by using some analytical approximation technique which replaces the stochastic formulation with a deterministic one. Stochastic simulation involves repetitive random sampling from probability distributions in order to determine their effects on output variables. The latter method is more generally applicable; however, it may be several orders of magnitude slower [17].

Alternative techniques for reliability quantification include so-called *reliability surrogate measures* (designed to have a strong positive correlation with *probabilistic reliability*) [194, 226], *graph theoretic techniques* which focus on redundancy and connectivity analysis (such as counting the number of independent paths to a source from every demand node) [4], *severity indices* such as the ratio of the total available annual supply to the total annual demand, *frequency and duration indices* which measure how frequently and for how long failure events of a given severity occur [112], and *economic indices* which measure the financial impact of failure events, such as revenue lost and repair costs [112]. In this dissertation it was decided to focus on probabilistic reliability and reliability surrogate measures, owing mainly to their generality and popularity in the hydraulic engineering community.

If a numerical quantification of reliability may be calculated for candidate designs, then it makes good sense to include this in the WDSDO model as an additional objective to be maximized. The following subsections contain discussions on probabilistic reliability, reliability surrogate measures, and graph theoretic reliability indicators.

4.3.1 Probabilistic WDS Reliability

Let $\acute{S}(\boldsymbol{x};\boldsymbol{z})$ denote a *feasibility function* for a design $\boldsymbol{x}$, under a given state combination $\boldsymbol{z} \in \mathcal{D}_{\boldsymbol{x}} \otimes \mathcal{F}_{\boldsymbol{x}}$, where $\mathcal{D}_{\boldsymbol{x}}$ denotes the set of possible demand loading conditions and $\mathcal{F}_{\boldsymbol{x}}$ denotes the set of possible failure events (including the event of no failures). Let $\acute{S}(\boldsymbol{x};\boldsymbol{z}) = 1$ in the event that the nodal demands are satisfied in the network (*i.e.* outflows q_i are equal to nodal demands d_i), and the nodal heads are within the permissible range ($h_{\min} \leq h_i \leq h_{\max}$), for all nodes $i = 1, \ldots, n$; and $S(\boldsymbol{x};\boldsymbol{z}) = 0$ otherwise. In DDA, demand flows are satisfied automatically; therefore reliability only depends on the satisfaction of head constraints. One may express the probabilistic reliability R of design $\boldsymbol{x}$ as

$$R(\boldsymbol{x}) = P(\acute{S}(\boldsymbol{x};\boldsymbol{z}) = 1).$$

Reliability R may be described as *the probability of finding the system in the feasible state at any given moment*. This is not the only possible formulation of probabilistic reliability. For example, one may consider probabilistic functions of individual node reliabilities, or the probabilities of attaining different grades of feasibility (*e.g.* the probability that $c\%$ of nodes attain their pressure goals, and $(100-c)\%$ are within 5% of their goals).

Each sample combination $\boldsymbol{z} \in \mathcal{D}_{\boldsymbol{x}} \otimes \mathcal{F}_{\boldsymbol{x}}$ is associated with a non-negative probability of occurrence $p(\boldsymbol{z})$ during a similar time period, which is subject to the condition that $\sum_{\boldsymbol{z}} p(\boldsymbol{z}) = 1$. In order to determine whether $\acute{S}(\boldsymbol{x};\boldsymbol{z}) = 1$, one must conduct a hydraulic simulation for condition $\boldsymbol{z}$ and evaluate the hydraulic constraints. This must be evaluated for every $\boldsymbol{z} \in \mathcal{D}_{\boldsymbol{x}} \otimes \mathcal{F}_{\boldsymbol{x}}$ in order to determine the value of R. The analytical evaluation of R may be achieved as

$$R(\boldsymbol{x}) = \sum_{\boldsymbol{z} \in \mathcal{D}_{\boldsymbol{x}} \otimes \mathcal{F}_{\boldsymbol{x}}} p(\boldsymbol{z}) \acute{S}(\boldsymbol{x};\boldsymbol{z}).$$

Note that this may require an extremely large number of hydraulic simulations, growing exponentially with the size of the WDS. It is very unlikely that these probabilities $p(\boldsymbol{z})$ are known, but the problem may be subdivided into the constituent events. For example, $\mathcal{F}_{\boldsymbol{x}}$ may be conditioned over the events of no components failing $\mathcal{F}_{\boldsymbol{x},0}$, one component failing $\mathcal{F}_{\boldsymbol{x},1}$, and so forth, up to $\kappa = |\boldsymbol{x}|$ component failures $\mathcal{F}_{\boldsymbol{x},\kappa}$.

Let $\mathcal{I}_{\boldsymbol{x},k}$ denote a set of k components, which is a subset of the κ components constituting design $\boldsymbol{x}$. There are $\binom{\kappa}{k}$ sets of this type. Let $\mathcal{I}^i_{\boldsymbol{x},k}$ denote the i-th such set, containing k specific components. Let $\hat{F}\left(\mathcal{I}^i_{\boldsymbol{x},k}\right) = \mathcal{F}^i_{\boldsymbol{x},k}$ denote the event of this set of components failing. It is often assumed that events of individual component failure are independent of the failure condition of the other components [267]. This would yield the probability of failure as

$$p(\hat{F}\left(\mathcal{I}^i_{\boldsymbol{x},k}\right)) = \prod_{x_j \in \mathcal{I}^i_{\boldsymbol{x},k}} p(\hat{F}\left(\{x_j\}\right)),$$

where $p(\mathcal{F}\left(\{x_j\}\right))$ is the probability of failure of a specific component x_j.

In a similar manner, given the independence of demand loadings and failure events, one may condition over the various demand scenarios $\boldsymbol{d}_j \in \mathcal{D}_{\boldsymbol{x}}$ with their associated probabilities of occurrence $p(\boldsymbol{d}_j)$. Reliability may then be calculated as

$$R(\boldsymbol{x}) = \sum_{j=1}^{|\mathcal{D}_{\boldsymbol{x}}|} p(\boldsymbol{d}_j) \sum_{k=0}^{\kappa} \sum_{i=1}^{\binom{\kappa}{k}} \left(\prod_{x_j \in \mathcal{I}^i_{\boldsymbol{x},k}} p(\hat{F}\left(\{x_j\}\right)) \right) S\left(\boldsymbol{x}; \boldsymbol{z} = \left\langle \boldsymbol{d}_j, \mathcal{F}^i_{\boldsymbol{x},k} \right\rangle\right). \tag{4.1}$$

There are several problems with this model (including the assumption of independence), but the most serious one remains the number of hydraulic simulations required to compute the value of R. Even for a medium-sized WDS consisting of 100 components, limited to 24 demand conditions, and assuming that the probability of more than three simultaneous component failures is negligible, more than 4 million hydraulic simulations would be required. This makes the full analytical calculation of probabilistic reliability for large WDSs virtually impossible. The majority of researchers simplify the problem by considering only single component failures [267]. This significantly reduces the number of simulations required to evaluate the reliability of a single design to $|\mathcal{D}| \times (\kappa + 1)$.

Another problem is the uncertainty associated with the demand loadings and component failure probabilities. It may be very difficult to determine exact probabilities of the different events,

particularly for a new system with no historical record [112]. In a more realistic model, parameter interdependence would have to be accommodated, requiring integration over a continuous joint probability distribution for demands. Such a joint probability function is typically not known, and would require extensive computation to determine [267]. Dependency might also exist between other parameters. For example, a single pipe failure may cause a chain of failures in a section of the WDS, owing to the additional strain placed on the working components. Unfortunately, there is very little research available on this topic. However, it is generally found that component availability in a WDS scenario is high, with multiple simultaneous failures occurring infrequently. Furthermore, there exists empirical research on pipe break rates which may be used to implement the assumption of component failure independence [112]. The method of Cullinane *et al.* [45] uses historical data to estimate the probability of component failure for component i as $p(\hat{F}(\{i\})) = \lambda_i/(\mu_i + \lambda_i)$, where λ_i is the mean value of the time for which a component is in a failed state, and μ_i is the mean value of the time between component failures.

Numerous researchers have demonstrated that ignoring uncertainty in demand and model parameters can lead to overestimation of WDS reliability [112]. In 2005 Babayan *et al.* [16] defined *robustness* as the ability of a network to provide adequate supply to customers despite fluctuations in the design parameters. Uncertainty in pipe characteristics (diameter / roughness) and reservoir levels may also be considered in a reliability model. Xu and Goulter [267] found that reliability was most sensitive to demands and minimum pressure head limits, less sensitive to initial reservoir levels, and relatively insensitive to pipe roughnesses.

The first model to consider uncertainty in nodal heads was that of Lansey *et al.* [158] in 1989. However, head was not expressed as a function of uncertain input parameters in this model. Xu and Goulter [267, 268] were the first to develop fully probabilistic models for WDS design, which use approximation techniques for the estimation of individual nodal reliabilities. These values are used either to constrain the optimisation model to minimum reliability levels at a number of critical nodes, or are aggregated by some function to express overall system reliability. The probability of node failure F_i for node i is taken as

$$F_i = P(h_i < h_{\min}) = 1 - R_i,$$

where R_i is the reliability of the node. They developed two different reliability approximation models in 1998 and 1999. The first model uses the mean value first-order second-moment method (MVFOSM) of Yen *et al.* [270] in 1986 to compute the first two moments of nodal pressure heads. The second model uses the more accurate first order reliability method (FORM) of Hasover and Lind [120] in 1974 to develop a reliability index for constrained nodes. Unfortunately, both models suffer from drawbacks which make them impractical to use on a large scale, including the repetitive computationally expensive calculation of derivatives and matrix inversions (although FORM is more accurate, it is even more computationally demanding), assumptions of parameter independence, and potential numerical errors [16]. However, it is very informative to consider the approaches adopted in these models [267].

The 1998 WDS probabilistic reliability model of Xu and Goulter [267] is a linear approximation model which caters for uncertainty in demands $\boldsymbol{d}$, pipe roughness coefficients $\boldsymbol{c}$, and reservoir levels $\boldsymbol{h}_{\mathrm{R}}^0$. The following simplifying assumptions are made: (1) All uncertain parameters (demands, *etc.*) are normally distributed, and (2) sufficient accuracy may be achieved by using a linear approximation of the nonlinear hydraulic model, which employs expected values of the uncertain parameters, provided that the original nonlinear model is solvable at these expected values. This linearized system is constructed by a first-order Taylor series expansion on the

expected values $\overline{\boldsymbol{h}}$ (nodal heads), $\overline{\boldsymbol{d}}$, $\overline{\boldsymbol{c}}$, and $\overline{\boldsymbol{h}_{\mathrm{R}}^0}$, that is

$$\begin{aligned} G(\boldsymbol{h},\boldsymbol{d},\boldsymbol{c},\boldsymbol{h}_{\mathrm{R}}^0) &\approx G(\overline{\boldsymbol{h}},\overline{\boldsymbol{d}},\overline{\boldsymbol{c}},\overline{\boldsymbol{h}_{\mathrm{R}}^0}) + \frac{\partial G}{\partial \boldsymbol{h}}(\boldsymbol{h}-\overline{\boldsymbol{h}}) + \frac{\partial G}{\partial \boldsymbol{d}}(\boldsymbol{d}-\overline{\boldsymbol{d}}) + \frac{\partial G}{\partial \boldsymbol{c}}(\boldsymbol{c}-\overline{\boldsymbol{c}}) \\ &\quad + \frac{\partial G}{\partial \boldsymbol{h}_{\mathrm{R}}^0}(\boldsymbol{h}_{\mathrm{R}}^0 - \overline{\boldsymbol{h}_{\mathrm{R}}^0}) \\ &= G(\overline{\boldsymbol{h}},\overline{\boldsymbol{d}},\overline{\boldsymbol{c}},\overline{\boldsymbol{h}_{\mathrm{R}}^0}) + J\Delta\boldsymbol{h} + J_d\Delta\boldsymbol{d} + J_c\Delta\boldsymbol{c} + J_{h_{\mathrm{R}}^0}\Delta\boldsymbol{h}_{\mathrm{R}}^0, \end{aligned}$$

where J denotes the Jacobian matrix with respect to heads, and J_d, J_c, and $J_{h_{\mathrm{R}}^0}$ denote the Jacobian or sensitivity matrices for demands, pipe roughnesses and initial reservoir levels. This yields the following linear model for nodal pressure heads,

$$\boldsymbol{h} = \overline{\boldsymbol{h}} - J^{-1}J_c\Delta\boldsymbol{c} - J^{-1}J_{h_R^0}\Delta\boldsymbol{h}_{\mathrm{R}}^0 - I\Delta\boldsymbol{d}, \tag{4.2}$$

where I is the identity matrix. The inverse Jacobian may be calculated by hydraulic simulation using the expected values (Xu and Goulter calculated inverses by means of Zollenkopf bi-factorization [267]). Since the uncertain parameters are pairwise statistically independent, the nodal head variances σ_i^2 may be calculated using MVFOSM as

$$\sigma_i^2 = \sum_{j=1}^{n+n_{\mathrm{p}}+n_{\mathrm{R}}} \hat{b}_{ij}\sigma_{x_j}^2, \qquad i = 1,\ldots,n,$$

where n is the number of nodes, where n_{p} is the number of pipes, where n_{R} is the number of reservoirs, where $\hat{b}_{ij}$ is the appropriate element of matrix $\left[-J^{-1} : -J^{-1}J_c : -J^{-1}J_{h_{\mathrm{R}}^0}\right]$, and $\sigma_{x_j}^2$ is the variance of the random variable. Since the parameters are all normally distributed, the linear approximation of heads is also normally distributed. Thus, given $\Phi(\cdot)$ as the standard normal cumulative distribution function, the probability of hydraulic failure at a specific node i may be calculated as

$$F_i = P(h_i < h_{\min}) = 1 - \Phi(Z_i) = 1 - R_i$$

where Z_i is defined by

$$Z_i = \frac{\overline{h}_i - h_{\min}}{\sigma_i}.$$

In order to calculate the system reliability, the linear model has to be re-evaluated for each different network configuration under component failure. As before, one may then condition over the various events of failure conditions. Xu and Goulter [267] only consider single component failure. In order to simplify the notation, let $\widetilde{P}_{\mathrm{f}}^i$ denote the probability of failure of component i, with $\widetilde{P}_{\mathrm{f}}^0$ the probability of no failures. Nodal reliability R_i is therefore estimated as

$$R_i = \frac{\sum_{j=1}^{|\boldsymbol{d}|} p(d_j) \sum_{i=0}^{\kappa} \widetilde{P}_{\mathrm{f}}^i P(h_i < h_{\min} \mid \widetilde{P}_f^i)}{\sum_{j=1}^{|\boldsymbol{d}|} \sum_{i=0}^{\kappa} \widetilde{P}_{\mathrm{f}}^i},$$

where the denominator normalizes the events to single component failures only. Unfortunately, owing to dependencies between the nodal heads, this technique may not be used to estimate system reliability as per (4.1). Instead a weighted aggregation method is used, so that system reliability is

$$R = \sum_{i=1}^{n} w_i R_i,$$

where the weight w_i may be assigned according to the importance of feasibility at node i. One method of doing so is setting w_i as the ratio of the demand at i to the total demand. Provided $\sum_i w_i = 1$, system reliability is taken as the weighted mean of nodal reliability. Xu and Goulter [267] verified their method by means of *Monte-Carlo simulation* (MCS) and found it to be accurate within the region of the expected values. However, the results deteriorated as variation in uncertain parameters increased, particularly in the case of large demand variation. This is due to the linearity assumption, whereas real systems have a nonlinear relationship between input variables and head.

Babayan *et al.* [16] developed an integration-based method whereby the stochastic probability constraint

$$P(\boldsymbol{h} > \boldsymbol{h}_{\min}) \geq P_{\min}$$

is converted to a set of deterministic constraints of the form

$$\xi_{\mathrm{h}_i} \geq h_{i,\min} + \vartheta\sigma_{\mathrm{h}_i}, \qquad i = 1, \ldots, n,$$

where ξ_{h_i} and σ_{h_i} are the mean and SD of the head at node i, and ϑ is a safety factor for the nodal heads (*e.g.* a value of $\vartheta = 3$ would include most of the range of h_i). Babayan *et al.* assumed that heads are a function of uncertain demands and pipe roughnesses only, and used independent Gaussian PDFs for the nodal demands, as well as independent uniform distributions for the pipe roughness coefficients [17]. Furthermore, they assumed the validity of the *superposition principle*, which is

$$\begin{aligned} & h_i\left(\xi_{\chi_1} + \delta_1, \ldots, \xi_{\chi_v} + \delta_v\right) - h_i\left(\xi_{\chi_1}, \ldots, \xi_{\chi_v}\right) \\ \approx & \sum_{j=1}^{v} \left[h_i\left(\xi_{\chi_1}, \ldots, \xi_{\chi_j} + \delta_j, \ldots, \xi_{\chi_v}\right) - h_i\left(\xi_{\chi_1}, \ldots, \xi_{\chi_v}\right)\right] \\ = & \sum_{j=1}^{v} \left[h_i\left(\chi_j\right) - h_i\left(\boldsymbol{\xi}\right)\right], \end{aligned}$$

where $\boldsymbol{\chi} = \{\chi_1, \ldots, \chi_v\}$ is the set of uncertain parameters, where $\boldsymbol{\xi} = \left\{\xi_{\chi_j}\right\}$ is the vector of parameter means, and where δ_j is some perturbation of the j-th mean. This yields an approximation for the mean and SD of nodal heads as

$$\Gamma_{i,j} = \int_{-\infty}^{\infty} \left[h_i(\chi_j) - h_i\left(\boldsymbol{\xi}\right)\right] p_j(\chi_j)\, \mathrm{d}\chi_j \tag{4.3}$$

$$\xi_{h_i} \approx h_i(\boldsymbol{\xi}) + \sum_{j=1}^{v} \Gamma_{i,j} \tag{4.4}$$

$$\sigma_{h_i}^2 = \sum_{j=1}^{v} \int_{-\infty}^{\infty} \left[h_i(\chi_j) - h_i\left(\boldsymbol{\xi}\right) - \Gamma_{i,j}\right]^2 p_j(\chi_j)\, \mathrm{d}\chi_j, \tag{4.5}$$

where $p_j(\chi_j)$ is the PDF of parameter χ_j. These expressions may be evaluated by means of numerical integration [24] (see Appendix B.4). Babayan *et al.* [16] used an optimisation technique whereby the head means and SDs were computed for an initial configuration, using (4.3)–(4.5), in order to identify n_c critical nodes as those which violate their head constraints. Significant variables χ_j are also identified from their percentage contribution to (4.5), for example by taking all those variables which contribute more than 2%. Optimisation proceeds with a GA, using a penalty function of the form $P = p_{\mathrm{f}} \sum_{i=1}^{n_c} \max(0, h_{i,\min} + \vartheta\sigma_{h_i} - \xi_{h_i})$ in order to improve head deficit at the critical nodes. Computational efficiency is improved by only using the significant

variables in (4.3)–(4.5). Every few generations the sets of critical nodes and significant variables are updated.

Instead of analytical calculation, a common strategy is to estimate reliability by means of Monte-Carlo simulation (MCS). This requires a large number trials (typically more than 1 000), where each trial uses a stochastically sampled combination of input parameters (demands, roughnesses, *etc.*), and each uncertain parameter value is sampled from its own PDF. Reliability would then be estimated as the ratio of those trials which are feasible to the total number of trials. As the number of trials tends to infinity, the MCS estimate approaches the actual reliability exactly. Xu and Goulter [267] conducted an empirical study for WDSs of different sizes, and concluded that the distribution of hydraulic results changed very little after 3 000 trials. Nevertheless, they employed 5 000 trials in later studies. It is also uncertain whether this would be sufficient for very large systems with thousands of variables. Regardless, this still entails too many hydraulic simulations to be computationally efficient during WDSDO, where MCS must be conducted for each candidate design. Numerous researchers have attempted to simplify the problem, such as by using sampling reduction techniques [141], by making assumptions such as the probability of multiple simultaneous component failures being negligible [112], or by using meta-models such as artificial neural networks to speed up hydraulic simulations significantly [22].

The 2005 model of Kapelan *et al.* [141] uses a multi-objective GA combined with the notion of *Latin Hypercube Sampling* (LHS) introduced by McKay *et al.* [173] to significantly reduce the number of samples required per generation. The goal of LHS is to achieve a more uniform sampling which requires far fewer samples in order to be representative of a distribution than traditional pseudo-random sequence sampling used in MCS. It remains desirable, however, to maintain differences between consecutive samplings. Suppose that N samples must be generated from a decision space $\boldsymbol{D}$ which is continuous and uniformly distributed. LHS proceeds by generating a sequence of N draws in probability space [0,1] using the formula

$$\Omega(j) = \frac{j-1}{N} + u_j, \quad j = 1, \ldots, N,$$

where u_j is a uniform random variable between 0 and $\frac{1}{N}$. This sequence is then mapped to decision space using the inverse cumulative probability distribution function in order to obtain a sample, $d(j) = F^{-1}(\Omega(j))$ [121]. LHS may easily be adapted for discrete variables. In the WDS problem, for a n_p pipe system and a solution population of size N, it is required to draw $N \times n_\mathrm{p}$ samples for initialization. Using LHS and a uniform distribution of pipe sizes, one may generate a sequence $\boldsymbol{s}_i$ of N draws for each pipe $i = 1, \ldots, n_\mathrm{p}$. Each of these n_p sequences may then be shuffled randomly, after which a solution $\boldsymbol{x}_j$ may be constructed by setting the value of its i-th pipe variable to the j-th entry of $\boldsymbol{s}_i$.

Kapelan *et al.* [141] combined LHS with an averaging technique over the lifetime of a solution in order to reduce the required number of samples per generation even further. This works by assigning an initial age $a_{i,t_0} = 0$ to a solution i when it is first created in generation t_0, incrementing the age in each consecutive generation for which it survives, and taking reliability R_i as the average of the computed reliabilities over the generations (*i.e.* $R_i = \sum_{t=t_0}^{t_n} R_{i,t}/a_{i,t_n}$, where t_n is the current generation and $R_{i,t}$ is the computed reliability in generation t). Using this methodology they were able to obtain good results with very few samples (5–20) per generation.

In 2006 Babayan *et al.* [17] compared the analytical integration model [16] to the LHS sampling method [141], both employing GAs, towards the single-objective, reliability-constrained optimisation of the NYTUN WDS benchmark under uncertainty of demands and pipe roughnesses. Both methods are capable of producing near-optimal solutions with a similar computational

efficiency, outperforming MCS by two orders of magnitude. They favoured the use of the LHS method because of its generality, slightly improved robustness, and ease of use [17].

Reliability is usually measured in terms of feasibility, defined as the situation of attaining minimum pressure at all nodes. This is typical of analysis employing a demand-driven hydraulic model. If instead one uses a pressure-driven model, then a more fine-grained analysis may be employed, allowing one to quantify how much of the required demand is satisfied. To this end two *demand satisfaction* (DS) measures were employed in this dissertation rather than simply providing the probability of feasibility, providing additional knowledge regarding the degree of failure of a WDS. The first measure is *average demand satisfaction under uncertainty*, denoted by ADSU $= \text{AVG}_{\mathcal{D}}(\sum_{i=1}^{n} q_i / \sum_{i=1}^{n} d_i)$, where d_i is the demand and q_i is the actual outflow at node i under uncertain demands in a MCS setting. The second DS measure is *average demand satisfaction under pipe failure* (ADSF) calculated for each design, defined as before, but averaged across all single-pipe failure conditions $\mathcal{F}$. The Object Oriented Toolkit for EPANET (OOTEN) developed by Van Zyl [240] in 2007 was used in this dissertation. Rather than conducting PDA by means of simulating demand nodes as emitters (which failed to converge to stable solutions for the larger WDS benchmarks), a more robust methodology was employed whereby DDA was repeatedly conducted, adjusting the nodal demands until minimum pressures were satisfied.

Another avenue of future computational acceleration entails parallel processing, enabling the use of multiple networked computers or highly parallel processors, such as graphics cards. This has the potential to reduce hydraulic simulations to linear time complexity. Although these advances may have the effect of reducing the number of combinations and computational requirements by orders of magnitude, the growth of scenario combinations is still exponential in the number of components, and uncertainties grow with the size of the system. Whether this may be overcome by the named techniques and technological advances alone remains to be seen. In order to address this issue, several researchers have suggested *reliability surrogate measures* which are computed for a critical subset of demand scenarios [194, 227]. These shall be discussed in the following subsection.

4.3.2 WDS Reliability Surrogate Measures

To avoid the computational extravaganza of MCS a surrogate reliability measure may be employed instead, using a critical subset of demand loadings (typically a peak loading condition and perhaps some failure / fire scenarios). *Minimum excess nodal head* or the *sum total of excess nodal heads* may be used as very simple reliability surrogate measures. Three more advanced measures are presented in this section, namely the *Resilience Index*, *Network Resilience*, and *Flow Entropy*.

Resilience Index

Introduced by Todini [227] in 2000, the *Resilience Index* is an indicator of excess system power (energy per unit time). It is based on the notion that internal energy losses will increase when demand increases or pipe failure occurs. Therefore it is desirable to provide more power at each node in a looped network than is required, so that a sufficient surplus exists to be dissipated internally in case of failures. This surplus is then used to characterize the resilience.

If $P_{\text{r,tot}} = \gamma \sum_{k=1}^{n_{\text{R}}} q_{\text{R},k} h_{\text{R},k}$ is the total available power supplied to the system, where $q_{\text{R},k}$ is the flow and $h_{\text{R},k}$ is the pressure head supplied by the k^{th} reservoir, and γ is the specific weight of

water, then

$$P_{\mathrm{r,tot}} = P_{\mathrm{r,int}} + P_{\mathrm{r,ext}},$$

where $P_{\mathrm{r,int}}$ is the power dissipated in the pipes, where $P_{\mathrm{r,ext}} = \gamma \sum_{i=1}^{n} q_i h_i$ is the power delivered to users in terms of flow q_i and head h_i at node i, and where n and n_{R} denote the number of nodes and the number of reservoirs in the network, respectively.

Todini introduced a Resilience Index $I_{\mathrm{r}} = 1 - P^*_{\mathrm{r,int}}/P^*_{\mathrm{r,max}}$, where $P^*_{\mathrm{r,int}} = P_{\mathrm{r,tot}} - \gamma \sum_{i=1}^{n} q_i^* h_i$ is the power dissipated in the network to satisfy the total demand (at any given pressure — typically in excess of that required) and $P^*_{\mathrm{r,max}} = P_{\mathrm{r,tot}} - \gamma \sum_{i=1}^{n} q_i^* h_i^*$ is the maximum power available to be dissipated internally if the constraints in terms of both demand and head are satisfied at the nodes. After appropriate substitutions, the Resilience Index may be written as

$$I_{\mathrm{r}} = \frac{\sum_{i=1}^{n} q_i^*(h_i - h_i^*)}{\sum_{k=1}^{n_{\mathrm{R}}} q_{\mathrm{R},k} h_{\mathrm{R},k} - \sum_{i=1}^{n} q_i^* h_i^*}. \tag{4.6}$$

Network Resilience

Prasad and Park [194] developed the *Network Resilience* metric in 2004 in response to the Resilience Index of Todini described above. The advantage of Network Resilience is that it explicitly rewards *reliable loops* of similarly sized pipes by penalizing sudden changes in pipe diameter. The Todini Resilience Index does not explicitly reflect the effects of redundancy. A branched network with sufficient surplus head may therefore still be desirable, so that maximisation of surplus head or power alone may not be sufficient for a reliable network. Prasad and Park demonstrated that increases in the value of the Resilience Index did not necessarily improve network reliability, and that using Network Resilience as an objective instead of the Resilience Index, produced more robust designs, in that the designs were better able to handle component failure [194].

The notion of Network Resilience (I_{n}) incorporates the effects of both surplus power and reliable loops. The surplus power at any node i is given by $P_{\mathrm{r},i} = \gamma q_i(h_i - h_i^*)$. A loop may be considered reliable if the pipes incident with a node are not widely varying in diameter. If $d_1 > d_2 > d_3$ are the diameters of three pipes incident with node i, then the uniformity of that node is given by $U_i = (d_1 + d_2 + d_3)/(3d_1)$ and in generalized form as

$$U_i = \frac{\sum_{j=1}^{n_{\mathrm{p}}^i} d_j}{n_{\mathrm{p}}^i \times \max\left\{d_1, \ldots, d_{n_{\mathrm{p}}^i}\right\}},$$

where n_{p}^i is the number of pipes incident with node i. Note that $U_i = 1$ if pipes incident with a node all have the same diameter, while $U_i < 1$ if pipes incident with a node have different diameters. For nodes incident with only one pipe, the value of U_i is taken to be one. The combined effect of both surplus power and connecting pipe uniformity of node i, called *weighted surplus power*, is expressed as $X_i = U_i P_{\mathrm{r},i}$. For the network as a whole, it is given by

$$X = \sum_{i=1}^{n} X_i = \sum_{i=1}^{n} U_i P_{\mathrm{r},i} = \sum_{i=1}^{n} U_i q_i (h_i - h_i^*).$$

This may be normalized by dividing with maximum surplus power to obtain Network Resilience as

$$I_{\mathrm{n}} = \frac{X}{X_{\max}} = \frac{\sum_{i=1}^{n} U_i q_i^*(h_i - h_i^*)}{\sum_{k=1}^{n_{\mathrm{R}}} q_{\mathrm{R},k} h_{\mathrm{R},k} - \sum_{i=1}^{n} q_i^* h_i^*}, \tag{4.7}$$

where $X_{\max}$ is the maximum surplus power. Network resilience may also be viewed as equivalent to the Resilience Index with surplus power at node i given a weight of U_i based on the uniformity in diameter of pipes incident with it. Care should be taken before applying Network Resilience blindly. In some cases, such as when service lines are connected to mains pipes, differences of diameter are essential. However, this problem may be overcome by the fact that service lines are typically not designed at the same time as the primary bulk pipe system, and their demand is lumped together at the nodes [194].

Flow Entropy

Information entropy is a statistical measure developed by Shannon [215] in 1948 which quantifies the degree of uncertainty present in a probabilistic system. Entropy is commonly expressed as

$$\mathcal{E} = \Upsilon_n\left(p_1, \ldots, p_n\right) = -\sum_{i=1}^{n} p_i \ln p_i,$$

where p_i is the probability associated with the i-th outcome, and $-\ln p_i$ is the so-called *self-information* of a random variable. Entropy theory is often used to determine the least-biased probability distribution for a system given limited data [67]. Entropy has the property that it is maximal if all probabilities are equal, *i.e.* $\Upsilon_n\left(p_1, \ldots, p_n\right) \leq \Upsilon_n\left(\frac{1}{n}, \ldots, \frac{1}{n}\right)$, which concurs with the notion that uncertainty is highest when all events are equiprobable.

In 1990 Awumah *et al.* [15] were the first to consider entropy measures of WDS reliability based on Shannon's function. In 1993 Tanyimboh and Templeman [225] defined *Flow Entropy* for use in WDS analysis. The Flow Entropy of a WDS is given as

$$\begin{aligned} \mathcal{E}_{\mathrm{f}} &= \mathcal{E}_{\mathcal{R}} + \sum_{i=1}^{n} \frac{Q_i}{Q} \mathcal{E}_i \qquad (4.8) \\ &= -\sum_{k \in \mathcal{R}} q_{\mathrm{R},k}/Q \ln(q_{\mathrm{R},k}/Q) - \frac{1}{Q} \sum_{i=1}^{n} Q_i \left[d_i/Q_i \ln(d_i/Q_i) + \sum_{j \in \mathcal{N}_i} q_{j,i}/Q_i \ln(q_{j,i}/Q_i), \right] \end{aligned}$$

where $\mathcal{E}_{\mathcal{R}}$ denotes the entropy of the sources (all reservoirs, tanks or external source nodes $k \in \mathcal{R}$), where n is the number of nodes, where Q_i denotes the total flow reaching node i and $\mathcal{E}_i$ denotes the entropy of node i, where Q is the sum of nodal demands, where $q_{\mathrm{R},k}$ is the inflow from source k, where d_i is the demand at node i, where $\mathcal{N}_i$ denotes the set of all the nodes immediately upstream from and connected to node i, and $q_{j,i}$ is the flow in pipe from node j to node i.

Flow Entropy is a measure of the uniformity of pipe flows [195]. Therefore, higher values of $\mathcal{E}_{\mathrm{f}}$ characterize a more balanced system, able to respond more gracefully to component failure. Flow Entropy has the desirable characteristic that it intrinsically rewards pathway redundancy (or loops), since distribution of flow across additional pathways will increase the entropy value. This is in contrast to the previous two reliability measures which only increase pathway redundancy incidentally, particularly if pipe failures are simulated. Despite this pressure on pathway redundancy, the system will not degenerate into a dense grid of small pipes, because there is a balancing force of smaller pipes limiting flow, increasing velocity and causing greater head losses, which is penalized by the WDS model constraints. The ability of entropy maximisation to enhance network reliability has been demonstrated by several researchers [195, 226].

Mixed Reliability Surrogate

It was surmised that a more practical representation of real-world reliability requirements could potentially be attained by a combination of the reliability surrogates, thereby incorporating the strengths of the different techniques. This is simply taken as normalized Flow Entropy $\mathcal{E}_{\mathrm{f}}$, multiplied by the Resilience Index I_{r}. This yields

$$I_{\mathrm{c}} = \mathcal{E}_{\mathrm{f}}/\mathcal{E}_{\mathrm{f,max}} \times I_{\mathrm{r}}.$$

This measure has the advantage of addressing uniformity of flow and pathway redundancy by means of Flow Entropy, and excess nodal power by means of the Resilience Index. Since both terms are in the range [0,1], they are assigned equal importance.

In this dissertation it was decided to compare these four surrogate measures, namely the reliability index, Network Resilience, Flow Entropy, and the Mixed Surrogate measure.

4.3.3 Graph Theoretic Reliability Measures

A WDS is usually represented as a network graph of nodes and edges (links). Graph theoretic techniques for WDS reliability estimation deal with the application of algorithms to analyze connectivity between graph nodes in order to quantify pathway redundancy (typically between demand nodes and sources). Reliability indices based on connectivity analysis alone are usually classed *mechanical reliability*, since they do not consider hydraulic operations. It should immediately be noted that the connectivity of a demand node to a source is a necessary but not sufficient condition for demand satisfaction [112], so that while these reliability indices do provide some insight into the flexibility of a WDS — they constitute a rough approximation only of hydraulic success.

The first use of graph theory for the reliability assessment of WDSs was by Goulter [109] and Jacobs and Goulter [133] in 1988. The most popular graph theory technique in WDS design involves the calculation of *minimum cut sets*. A cut-set is a set of edges which, when removed, disconnects one or more nodes in a graph. A minimum cut set is the smallest set of edges for which this will occur. Minimum cut-sets may also be defined in terms of demand-source pairs, that is any set of edges disconnecting a particular node from a particular source, or in terms of an individual node and all sources. If the probabilities of pipe (edge) failures are known, and the individual node to source minimum cut-sets have been computed, then one is able to calculate the probability that a node, or any node in a set of nodes (including the entire network) is disconnected from all sources [269].

One of the difficulties with the minimum cut-set method is that the problem of finding a minimum cut-set is NP-hard, which essentially means that the complexity of the problem is exponential in the size of the network when using the best known algorithms to solve the problem. Researchers have attempted to simplify the problem by edge reduction techniques (*e.g.* lumping a series of edges and nodes together as a single entity), but these techniques still constitute a significant computational burden [112]. Graph theoretic reliability measures are therefore not employed in this dissertation.

4.4 Network Layout Design

Traditionally, the pipe network layout is designed by the engineer and taken as fixed during the process of component sizing. In reality, network layout design is tightly coupled to the

WDSDO problem, both in terms of costs incurred and WDS reliability. It would be ideal to conduct optimisation on both problems simultaneously. However, this may easily cause the complexity to explode to infeasible levels (especially if one considers all possible interconnections between nodes). The problem may still be tractable if approached from a practical engineering perspective. In reality there are many physical limitations to the actual routes a pipeline may follow, owing to buildings, roads, other infrastructure and natural features. These elements allow the designer to immediately eliminate the majority of potential links from consideration, resulting in a *maximal practical layout* comprising several redundant links[5]. Another approach is for the engineer to design the basic layout in the traditional manner, and then insert as many useful redundant links as the layout permits (referred to from now onwards as a *redundant layout*).

The design methodology of Afshar *et al.* [4] calls for an initial *maximal layout* for a WDS network, in which all potentially useful links are provided. During the course of optimisation, links may be eliminated (by sizing them to zero) whilst still satisfying given engineering connectivity constraints, thus achieving simultaneous layout and component optimisation. It is recommended that an initial layout satisfy at least a level 2 reliability, *i.e.* each demand node is connected via two or more independent paths to source nodes (tanks or reservoirs), thereby guaranteeing that no additional part of the network will be isolated in the event of a single-pipe failure [4]. The approach of Afshar *et al.* is adopted in this dissertation in the context of a redundant layout, permitting a basic level of network layout design.

4.5 Additional Topics

There are several additional topics in WDSDO which warrant investigation, but could not be included in this dissertation due to scope constraints. The following topics constitute features which should appear in a comprehensive WDSDO model.

4.5.1 Total Costs: Incorporating Maintenance and Energy Costs

Embracing a long-term view with regard to WDS design is desirable, especially in a developmental setting where infrastructure is frequently inadequate and planning tends to be short-term. Towards this end, objective function costs may include replacement and maintenance costs over the lifespan of the system. There exists a relationship between the capacity or size of the installed WDS and its maintenance costs, owing to the fact that larger components have a smaller frequency of breakage. However, this relationship is non-trivial, since larger pipes and other components are more expensive to fix and replace. Therefore, the probabilities of breakage must be taken into account together with the cost to repair in order to obtain a useful estimate of maintenance costs over some time-span. In some cases, additional initial capital costs may be justified in view of minimizing costs over the WDS lifespan. The idea then is to adapt the cost objective function by adding the *present value*[6] (P_{V}) of repair and maintenance costs over an expected lifespan (frequently 20 or 25 years) to capital costs.

[5] This is another topic which warrants further research — using a GIS to derive such a layout from a given specification of nodes and a data layer of land-use polygons.

[6] Determining the present value of a future cost entails adjusting the cost to zero-time by means of discounting inflation and interest. This is equivalent to the amount of money required now to pay the cost in the future. The formula for present value P_{V} (at time zero), given a future value $F_{\mathrm{V}|t}$, is

$$P_{\mathrm{V}} = F_{\mathrm{V}|t}/(1+i_r)^t,$$

Furthermore, there is a trade-off between the capacity of installed pumps and pipe sizes, since larger pipes lose less energy to friction and may potentially be supplied by a smaller pump. However, in order to evaluate the true costs involved, one must consider the present value of the energy costs of pumping in addition to the initial pumping equipment costs. Energy costs may be calculated from an *energy price*, pump power consumption, and the operating point on a pump's volume-efficiency curve.

4.5.2 Tank Design

Tanks are important components of WDSs, particularly for large systems. They provide alternative water sources during periods of high demand or emergency conditions (such as pipe outages and fires), thereby improving WDS robustness. Furthermore, by storing a backup supply of water in alternative locations, they engender a larger effective delivery capacity for the same set of pipe sizes (since the water may flow from different sources and aggregate at the outflow nodes), which enables a reduction in the size of the supply mains (and associated cost savings). They fulfil the purpose of *pressure equalization*, particularly when they are located at the extremes of a system (where accumulated head losses along the path from the primary reservoir may be significant) [249].

The design of tanks poses a formidable challenge, owing to the difficulty of simulating the tank water cycle (particularly in a multi-tank environment), and the host of potential pitfalls associated with tanks. These include (1) *water quality issues*, such as water stagnating in a tank (which causes deterioration of disinfectant and poses a threat to human health), (2) *tank overflow problems* (causing water loss and drainage issues), (3) *tank location and sizing concerns* (tanks are large, unsightly and typically need to be raised), (4) *tank management difficulties* (intervention in terms of flow control into or out of a tank may be required, or tanks may have to be taken off-line for maintenance), (5) and even *security issues*, such as the threat of contaminants being introduced (perhaps maliciously) into the system [249]. Tank design was traditionally conducted as a manual process. However, this is a difficult process, particularly when the designer must enable several tanks to work together effectively.

A more technical problem pertaining to design optimisation is that tanks naturally introduce continuous variables, since tanks may be sized quite arbitrarily. One way to accommodate this is the adaptation of a mixed optimisation model, which caters for both discrete and continuous variables [195]. Another technique is the discretization of the search space, reducing the possible values of tank parameters [238].

Tank design requires an extended period of simulation over the course of a daily or weekly cycle, in order to determine tank operational behaviour and make adjustments where necessary. The primary goals in tank design are (1) that pressure constraints are satisfied throughout the system, (2) that tanks empty and fill over their operational range in order to avoid stagnation problems, (3) that tank water levels are returned to their original high level at the end of

where i_r is the *discount rate*, equivalent to the *interest rate* one could earn on an investment, and t is the time in years. If the current cost is C, then the pumping cost in t years is $C(1+f_r)^t$, where f_r is the *inflation rate*. Therefore the future value of a cost is $F_{\mathrm{V}|t}^{[C]} = C(1+f_r)^t$, which may be discounted to present value as

$$P_{\mathrm{V}}^{[C]} = C(1+f_r)^t/(1+i_r)^t = C\left(\frac{1+f_r}{1+i_r}\right)^t.$$

There is obviously uncertainty regarding the future rates of inflation and interest. In the absence of predictive economic models, one may simply use the current rates. This may also present an opportunity for sensitivity analysis on a range of inflation and interest values.

the demand cycle (a tank would typically be full in the morning, feed water into the system during the day, and recharge overnight)[7], and (4) that tanks provide emergency storage to cater for increased demand during fire flows, or supply the system in case flow from the primary reservoir is interrupted (a common guideline is a total emergency storage of AADD + [fire flow requirements]) [103, 238, 249]. In the case of advanced tank analyses, the actual water mixing and chemical properties may be simulated for water quality estimation. However, this requires intensive fluid modelling, which is beyond the scope of this dissertation.

Tanks are typically raised, owing to the requirements of supplying water at a minimum head, and due to the space restrictions on the ground [238]. A tank may be fully specified by a number of parameters. Assuming a cylindrical tank, these parameters include the location of a tank in the WDS, the diameter and length of the connecting pipe (commonly known as a *riser*), the tank base elevation, the height, and the diameter. Researchers often identify temporal hydraulic properties of a tank as additional parameters (sometimes used explicitly as decision variables during optimisation), such as the *initial water level* (at the start of the simulation period), the *maximum normal operating water level*, the *overflow water level*, and the *minimum normal operating water level*[8]. The most important of these is the minimum normal operational water level, since tanks are frequently designed to cease supplying water upon reaching this level, unless an emergency occurs [238]. Technically, one is also required to know the initial water level of a tank for simulation purposes, although this may be assumed to be some fraction of the height (*e.g.* 90%) [195]. In general, it is a good idea to employ fewer decision parameters in order to assist optimisation convergence. Vamvakeridou-Lyroudia *et al.* [238] take this to the extreme, by assuming that a fixed proportion of each tank is used for emergency storage, that tanks obey a fixed diameter-height ratio, and by ignoring riser diameters during optimisation. This enables them to specify a tank using only three variables, namely node location, volume and minimum normal operational level.

The following tank parameters may be considered as decision variables in a comprehensive tank model:

1. tank diameter d_{T},
2. tank top elevation t_{T},
3. tank base elevation b_{T},
4. tank node location n_{T},
5. tank riser diameter r_{T}, and
6. minimum normal water level ℓ_{T}.

Assumptions made might be that initial water levels h_{T}^0, are at 90% of the tank height $H_{\mathrm{T}} = t_{\mathrm{T}} - b_{\mathrm{T}}$, that overflow occurs when water reaches the top of the tank, and that tank parameters are within specified limits. The minimum water level may accommodated in two distinct ways. Firstly, if the designer wishes that the tank connection should close upon reaching this level, then this may be done. Otherwise, the specified minimum water level may be ignored during the course of simulation, and the actual simulated minimum water level would replace ℓ_{T}.

The following set of restrictions may be placed upon tank parameters in order to avoid unrealistic designs (such as an extremely tall, thin tank, or a very tiny tank):

[7] It is typically assumed that the demand cycle starts with full tanks just before the morning peak at 6am, in accordance with [238].

[8] Note that tank water level equates to pressure head in a tank.

1. tank diameter to height ratio is bounded as $\mathrm{dh}_{\min} \leq d_{\mathrm{T}}/H_{\mathrm{T}} \leq \mathrm{dh}_{\max}$,
2. tank volume is bounded as $V_{\mathrm{T},\min} \leq V_{\mathrm{T}} = \pi H_{\mathrm{T}} d_{\mathrm{T}}^2/4 \leq V_{\mathrm{T},\max}$,
3. tank base elevation is bounded as $b_{\mathrm{T},\min} \leq b_{\mathrm{T}} \leq b_{\mathrm{T},\max}$,
4. tank node location n_{T} is limited to a set of viable nodes $\mathcal{N}_{\mathrm{T}}$,
5. tank risers are treated as ordinary pipes in the WDS,
6. tanks are limited to a maximum number in the system (*e.g.* 4).

These restrictions on tank parameters have been adapted from Prasad and Tanyimboh [195], although this formulation is slightly less restrictive.

The capital costs of tanks depend on their size (a tank cost function should be available), and may be included in the WDSDO cost objective function. Tanks in a system may also be subject to a number of performance constraints evaluated for a given demand cycle. The following are commonly used. Firstly, the water level at the end of the demand cycle must equal the initial water level, that is $h_{\mathrm{T}}^{\varrho} = h_{\mathrm{T}}^{0}$, where h_{T}^{ϱ} denotes the water level at the end of the final period of a cycle. Secondly, the emergency storage capacity (volume) for all tanks $(V_{\mathrm{T}_1,\min}, \ldots, V_{\mathrm{T}_{n_{\mathrm{T}}},\min})$ must sum to the total required WDS emergency storage S_{E} in m^3 (fire flow plus interruption of mains requirements),

$$\sum_{i=1}^{n_{\mathrm{T}}} V_{\mathrm{T}_i,\min} = \sum_{i=1}^{n_{\mathrm{T}}} (\ell_{\mathrm{T},i} - b_{\mathrm{T},i}) \pi d_{\mathrm{T},i}^2/4 = S_{\mathrm{E}}.$$

Thirdly, for each tank $V_{\mathrm{T}_i,\min}$ must be less than some percentage of the total tank volume to avoid stagnation problems, *i.e.* $V_{\mathrm{T}_i,\min} \leq \varsigma V_{\mathrm{T}_i}$ (Prasad and Tanyimboh [195] use $\varsigma = 0.6$). Finally, no tank should overflow during any period in the simulated demand cycle, but this condition is already satisfied by the first constraint.

If the penalty method were to be used for constraint handling, a penalty function may be constructed for tank operational constraints of the form

$$\widehat{P}_{\mathrm{T}} = \sum_{i=1}^{n_{\mathrm{T}}} \frac{|h_{\mathrm{T}}^{\varrho} - h_{\mathrm{T}}^{0}|}{h_{\mathrm{T}}^{0} - \ell_{\mathrm{T}}} + \left| (S_{\mathrm{E}} - \sum_{i=1}^{n_{\mathrm{T}}} (\ell_{\mathrm{T},i} - b_{\mathrm{T},i}) \frac{\pi d_{\mathrm{T},i}^2}{4}) \right| / S_{\mathrm{E}} + \sum_{i=1}^{n_{\mathrm{T}}} \min[(\varsigma V_{\mathrm{T}_i} - V_{\mathrm{T}_i,\min})/\varsigma V_{\mathrm{T}_i}, 0].$$

EPANET 2 [203] may be used to conduct a simulation of tank hydraulics over the course of a demand cycle. In practice it makes sense to design for a maximum day cycle, especially to ensure that sufficient spare capacity is available for fire flows. It makes good sense to also consider tank performance over the course of an average day, since there is the possibility that tanks may be used inefficiently, which may result in water quality problems. Finally, in practice tank design is usually closely related to pump design and pump scheduling, since gravity flows alone may not fulfil the tank recharge requirements.

4.5.3 Pump Design

Pump design refers to the process of determining the capacity and number of pumps in order to satisfy the hydraulic requirements of a WDS. These pumps would typically be located in a pumping station, positioned after the water treatment facility [169]. A further task regarding pump design is the design of a pumping schedule (typically daily, divided into 24 hours) [108].

Finally, if required, booster pumps may be located in non-looped segments of the WDS [18]. There are typically at least two pumps in a pumping station (perhaps three or four in large systems) in order to provide a level of redundancy. These are often sized similarly in order to simplify maintenance [169].

Pumps have two primary purposes: adding energy to the WDS in order to overcome friction and gravity so as to satisfy consumer hydraulic requirements, and filling tanks as part of a demand cycle analysis. Pump capacities and number may be adjusted during the course of optimisation in order to achieve feasible pressures. Pumps should be able to satisfy a peak hour demand scenario. An additional rule of thumb is that the system should be able to satisfy fire flow demands with the largest pump out of commission [195]. Incorporating pump design as a cost objective requires pump capital costs and the present value of energy costs over the WDS lifetime. This requires simulation of pumping over the course of an extended period analysis (typically maximum day or maximum week), in order to determine pump schedules. The simplest type of scheduling is an on/off setting for individual pumps during each pumping period (usually chosen to coincide with demand periods). Pump costs may then be expressed as

$$C^{\mathrm{P}} = C_c^{\mathrm{P}} + C_e^{\mathrm{P}} = \sum_{i=1}^{n_{\mathrm{pmp}}} C_{c,i}^{\mathrm{P}} + \sum_{y=0}^{Y-1} \left(\frac{1+f_r}{1+i_r}\right)^y 365 \sum_{i=1}^{n_{\mathrm{pmp}}} \sum_{t=1}^{T} \frac{C_{\mathrm{U},t} P_{\mathrm{r},(i,t)}}{\eta_{\mathrm{sr},(i,t)}} D_{(i,t)}, \tag{4.9}$$

where n_{pmp} is the number of pumps, where $C_{c,i}^{\mathrm{P}}$ is the investment cost of the i-th pump, where Y is the number of years considered for the system lifetime (*e.g.* $Y = 20$), where f_r is the inflation rate (*e.g.* $f_r = 0.07$), where i_r is the interest rate (*e.g.* $i_r = 0.1$), where 365 denotes the number of days in year y, where T denotes the number of pump scheduling periods in a day (typically $T = 24$), where $C_{\mathrm{U},t}$ is the energy unit cost of pumping during period t (*e.g.* in units of R/kW-hr), where $P_{\mathrm{r},(i,t)}$, $\eta_{\mathrm{sr},(i,t)}$ and $D_{(i,t)}$ denote the power, efficiency and duration of operation time respectively, of pump i during period t [108]. Note that pump design requires pump performance curves both in terms of pressure and volume, and in terms of efficiency and volume.

4.5.4 Water Quality

Water quality is commonly measured at all outflow and storage points (*e.g.* tanks) in the WDS in terms of the concentration of disinfectant (typically chlorine), which must occur within specified limits [169]. This chemical is generally added at the source (although booster sites may also exist) and decays as it moves through the system, reacting with the pipe walls and pollutants in the water. Undesirable residual byproducts may also form depending on the water characteristics. However, the basic requirements for a water quality model are equations for conservation of chemical mass to simulate chemical dispersion and decay. This may be expressed as

$$\frac{\partial (K_{i,j})_t}{\partial t} = \frac{(q_{i,j})_t}{A_{i,j}} \frac{\partial (K_{i,j})_t}{\partial x_{i,j}} + \Theta (K_{i,j})_t, \qquad i,j \in [1,\ldots,n],\ i<j,$$

where $(K_{i,j})_t$ is the concentration of chemical in pipe i,j as a function of distance and time, where $x_{i,j}$ is the distance along the pipe, where $A_{i,j}$ is the cross-sectional area of pipe i,j, where $\Theta(K_{i,j})_t$ is the reaction (decay) rate of chemical within pipe i,j at time t [108].

EPANET [203] may be used to conduct water quality analyses. In each time period the hydraulic state variables are solved, and then chemicals are routed through the WDS, simultaneously undergoing decay. Decision variables may include the quantity of chemicals introduced at the entry point, the location of chemical booster sites, and the quantity introduced at the booster

sites. Constraints on minimum and maximum chemical concentrations may then be specified at the demand nodes and in storage tanks, in accordance with national water standards [32, 108].

4.5.5 Leakage

Leakage is frequently a large component of water losses. This results in significant revenue losses due to water losses and wasted energy, and affects system hydraulic performance [100]. In general there is a strong correlation between the age of a system and the amount of leakage. Leakage may vary from as little as 5% leakage in new systems to 40% or higher in old systems [251]. Although one is able to combat leakage by means of a leakage detection and repair program, the design of the WDS may also affect leakage. Designing against leakage involves pressure control (higher pressures cause more leakage), and the selection of stronger (larger) pipes. Through the use of a *leakage model*, one is able to incorporate leakage minimisation as an additional objective in WDS design, or instead add lost revenue from leakage to the cost objective function. Leakage models require the use of pressure-driven analysis to accurately estimate pressures in the system across all operational conditions [100]. Todini [229] conducted a review of state-of-the-art WDS leakage models in 2008.

4.5.6 Uncertainty in Pipe Characteristics

Pipe characteristics are also subject to uncertainty, typically experiencing degeneration as the WDS ages. Changes depend on the pipe material, the range of pressures and velocities in the system, and the corrosiveness and mineral content of water. It may be impossible to predict the change of pipe characteristics of a new system accurately as a function of time, yet performance degradation should be taken into account to accommodate future demands. Babayan *et al.* [17] use a uniform distribution for Hazen-Williams roughness coefficients in the range [90,110].

4.5.7 Valves

Valves are essential to the effective operation and maintenance of a WDS. In a system with no valves, the water must be cut off at the source in order to conduct any maintenance operations. Additional valves in the system allows a more accurate isolation of pipe failures in the system, thereby affecting fewer customers. Furthermore, if accurate simulations of the effects of component failure are required, then knowledge of the exact location of valves is also required. Ideally one should be able to isolate each segment of a loop, and this forms a common rule of thumb for valve location. One guideline is placing a minimum of three valves at each intersection, and two valves at each T-junction [103, 112]. The placement of valves constitutes an optimisation problem in its own right, although strictly it is a sub-problem of WDSDO. The cost of a valve obviously increases with valve size, and a valve is sized according to the diameter of the pipe in which it is fitted. Apart from minimizing cost, objectives in valve placement include minimizing the impact of valve closures on hydraulic reliability due to segment maintenance, and minimizing the number of valve closures required to isolate segments (ideally this should never be greater than four). Although valve selection and placement are not addressed in this dissertation, there is no practical reason why the optimisation model cannot be extended to do so in future.

4.5.8 Transient Analysis: A Warning

As mentioned in Chapter 2, failure to consider the effects of transients may cause serious damage to a WDS. It is therefore strongly advised that any important WDS design project include some form of transient analysis, even if only at a basic level. Although commercial software exists for transient analysis, it would be ideal if commonly used hydraulic simulation software (such as EPANET) were adapted to include the ability to conduct at least a crude analysis of maximum pressures during critical transient events. This will enable the designer to select pipe sizes which are rated for transient pressures rather than normal operating pressures alone. In addition to correct operating procedures, transients may be mitigated by means of *accumulators* (such as *air chambers* or open *surge tanks*) connected near a valve, or through the use of pressure relief valves at critical points in the system, which open automatically when pressures exceed safe limits [43]. A broad discussion of transient analysis appears in [251].

4.6 Chapter Summary

This chapter featured a number of topics required in the development of a more realistic WDSDO model, in fulfilment of Dissertation Objective 2 in §1.3 and in partial fulfilment of Objective 3(b). Furthermore, information was presented regarding objectives in addition to cost, in partial fulfilment of Objective 5.

The topics of water demand estimation (including the consideration of uncertainty), demand variation and emergency demands were discussed. These considerations are extremely important during the design of WDSs which are robust under the full range of possible conditions.

The topic of WDS reliability estimation was presented in some detail, focussing on probabilistic reliability (the probability that hydraulic requirements are met under a range of demand conditions and failure events) and reliability surrogate measures. It was concluded that a reduced sampling methodology combined with an evolutionary algorithm (*e.g.* LHS and distributed sampling across generations [141]) may the be best way of approaching the former, while the latter is investigated in the course of this dissertation by comparing the Resilience Index, Network Resilience and Flow Entropy measures.

Finally, several additional topics were discussed which should be considered in a comprehensive WDSDO model, yet could not be included in this dissertation owing to scope limitations. The topics of tank and pump design were presented in some detail, since these are important in a comprehensive model. In addition to capital and operating costs, potential hydraulic performance constraints were discussed. Other topics discussed included uncertainty in terms of other design parameters (including pipe roughnesses and initial reservoir levels), the consideration of running costs over the lifetime of the WDS (in order to merit a long-term design paradigm), water quality analysis, designing for leakage abatement, valve design and transient analysis. This constitutes material for possible future work.

Chapter 5

Multi-objective WDS Design Optimisation

This chapter concerns multi-objective optimisation (MOO) techniques and the essential extension of WDS design to the multi-objective case. It includes a literature review of MOO approaches to the WDS design problem, a generic mathematical formulation of the new problem, and a discussion of MOO in general, including performance evaluation in this context. Additionally, several multi-objective metaheuristics are discussed. There is a strong focus here on so-called Multi-objective Evolutionary Algorithms (MOEAs). However, alternative methodologies are also presented, including a hyperheuristic named AMALGAM. This is a generic evolutionary framework for the simultaneous incorporation of multiple diverse algorithms in the solution process.

5.1 Introduction

Multi-objective optimisation methods allow the user to examine the trade-off between multiple objectives by finding a set of *Pareto-optimal solutions* in objective function space. Multi-objective analysis is considered a more realistic approach in real engineering projects, where cost is very rarely the only consideration. It generally identifies a wider range of alternative solutions and provides more information about these solutions, empowering the decision maker. Water distribution problems may be considered inherently multi-objective. While minimizing capital cost and maximizing capacity are perhaps the most obvious conflicting objectives, other objectives may include minimizing risk, maximizing reliability, minimizing deviations from desired performance levels, maximizing water quality, minimizing operational cost, and so on. In a study of real water distribution system planning situations, Walski *et al.* [248] showed in 2000 that decision makers consistently preferred more robust designs over least-cost designs.

MOO yields a set of compromised solutions, also known as Pareto-optimal solutions. These solutions are *non-dominated* in the sense that no other solution is better with respect to all objectives, and moving from one Pareto-optimal solution to another results in the improvement of one objective but the degradation of another. A typical trade-off between cost and system reliability is shown for a solution set in Figure 5.1. An *inferior solution* is said to be *dominated*, because another solution exists that is better with respect to at least one objective, and no worse in terms of all other objectives. The hyper-curve which may be drawn through the Pareto-optimal solutions in multi-objective solution space is commonly known as the *Pareto-*

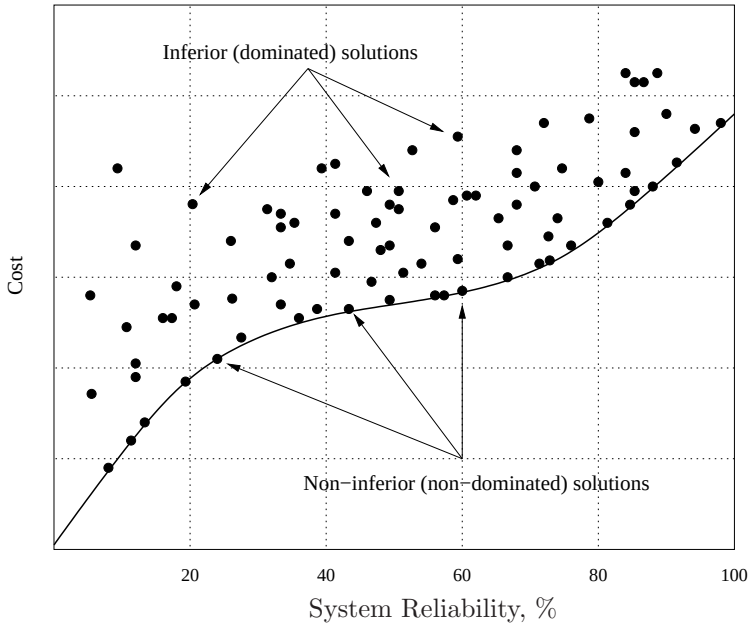


Figure 5.1: *The trade-off between cost and reliability in a WDS design scenario.*

optimal front. There may exist several sub-fronts into which a set of solutions may be classified, each lower front being dominated by the one above it. These fronts are typically ranked in order from best to worst, so that the first front is the current non-dominated front, and the number of the front to which a solution belongs is known as its front depth or *Pareto-rank* [251].

Consider a WDS optimisation problem with the $M = 2$ objectives of maximizing reliability, and maximizing the inverse of cost (see Figure 5.2). Let $\boldsymbol{d} = [d_1, d_2, \ldots, d_\kappa]$ be the decision vector representing a solution in decision space $\boldsymbol{D}$. An objective function $\boldsymbol{f} : \boldsymbol{D} \rightarrow \boldsymbol{Y}$ maps the decision vector to an objective function vector $\boldsymbol{y} = [y_1, y_2]$, representing the fitness of the solution. In accordance with the concept of *Pareto-dominance* for a maximisation problem, objective function vector $\boldsymbol{y}^1$ is said to dominate objective function vector $\boldsymbol{y}^2$ (denoted by $\boldsymbol{y}^1 \prec \boldsymbol{y}^2$) if, for at least one $i \in \{1, 2\}$, $y_i^1 > y_i^2$ and for $j \neq i$, $y_j^1 \geq y_j^2$. Thus, one may say that a solution $\boldsymbol{d}^1$ is better than a solution $\boldsymbol{d}^2$ when $\boldsymbol{f}(\boldsymbol{d}^1) \prec \boldsymbol{f}(\boldsymbol{d}^2)$ (as illustrated in Figure 5.2). The non-dominated set which represents the best possible trade-offs between different objectives is the Pareto-optimal set, $\boldsymbol{D}^* \subseteq \boldsymbol{D}$.

Multi-objective search algorithms must employ limited memory and finite running time; hence they cannot guarantee finding all members (or any for that matter) of the true Pareto-optimal front. The outcome of such a search therefore constitutes an approximation of the full Pareto-optimal set (for a continuous problem, this set has infinite cardinality). The goal of optimisation is to find a good approximation of the Pareto-optimal set $\boldsymbol{A}^* \subseteq \boldsymbol{D}^*$. However, this is an idealization — since the Pareto-set is not available in real world problems, it is not possible to test whether the algorithm has attained any Pareto-set members. More accurately, one may define $\boldsymbol{A}^* \approx \boldsymbol{D}_{\mathrm{S}}^* \subseteq \boldsymbol{D}^*$, where $\boldsymbol{D}_{\mathrm{S}}^*$ is any finite, representative subset of the Pareto-set with the property of good spread along the Pareto-front. That is, one strives to find an $\boldsymbol{A}^*$ that is a suitable $\boldsymbol{D}_{\mathrm{S}}^*$, but there are no guarantees in the metaheuristic domain. There may exist many different approximations, and different algorithms may produce sets of differing quality in terms of their closeness to the true Pareto-optimal front, and their diversity along it. Performance assessment must necessarily take the form of comparative analysis between approximation sets. One Pareto-approximation set A is said to dominate another set B ($A \prec B$) if for each $y \in B$,

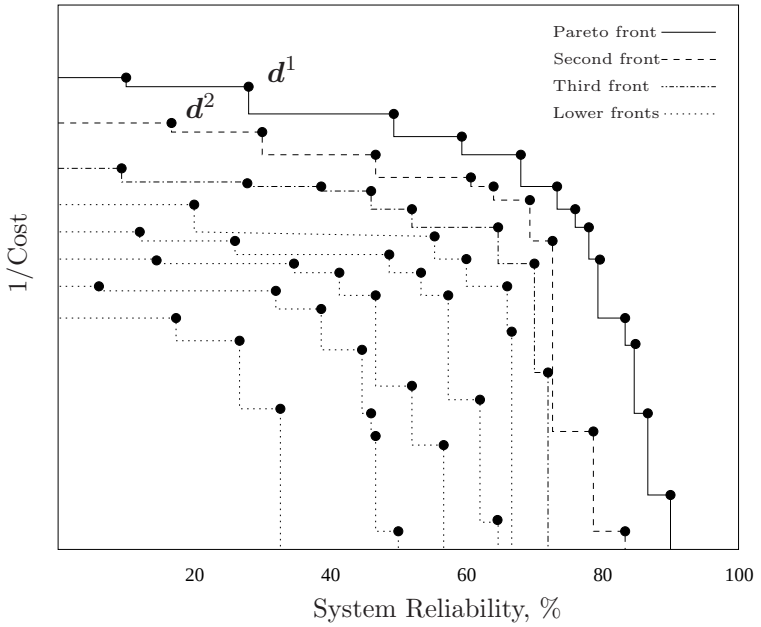


Figure 5.2: *Solution fronts in objective space. Solution* $\boldsymbol{d}^1$ *dominates solution* $\boldsymbol{d}^2$ *(*$\boldsymbol{d}^1 \prec \boldsymbol{d}^2$*).*

there exists an $x \in A$, such that $x \prec y$. Similarly, set A is said to *weakly dominate* set B ($A \lhd B$) if for each $y \in B$, either $y \in A$ or there exists an $x \in A$, such that $x \prec y$. Various measures of quality assessment exist, which will be discussed in §5.5

Traditional methods for MOO include weighted aggregation of objectives, alternating selection between objectives, replacing objectives with constraints [37], and goal programming [259]. In weighted aggregation methods, single-objective optimisation techniques may be used to generate a subset of the Pareto-optimal set by weighing the various objectives in a single lumped function. However, care should be taken when assigning weights to different objectives, especially when they cannot all be expressed in common units (such as monetary value). The weights may be varied systematically over the course of optimisation to obtain different Pareto-optimal solutions. This procedure was formalized by Cohon [37] in 1978 in what is known as the *weighting method*. This method suffers from several drawbacks. The first problem occurs when one objective value differs from another by orders of magnitude, so that large changes in weights may have a negligible effect on the outcome, or alternatively small changes may have a dramatic effect. Another major disadvantage is that it cannot generate all members of the Pareto-optimal front when this front is not convex [251]. In cases where it is difficult to equate objectives in terms of a common unit, it may be desirable to replace an objective with a constraint. This paradigm alludes to the *constraint method*, also by Cohon [37]. It operates by optimizing one objective while constraining the others to a realistic target value, and possibly varying that target value over the search period. Practically this has the effect of restricting the original feasible region. This approach fails to take advantage of certain methods' ability to thoroughly search the solution space, such as that of GAs [251]. Criterion-based methods switch between different objectives in determining which individuals are selected for the mating process. An example of this is the VEGA algorithm developed by Schaffer [213] in 1985, which has since been superseded by improved techniques [278]. Goal-programming involves the assignment of target values to objectives and minimizing the summed deviation from these targets (typically normalized to the objective operating ranges) [259]. These traditional methods suffer from several drawbacks, such as the exclusion of certain members of the Pareto-optimal set when the front is not convex, difficulty in determining an appropriate set of weighting coefficients to specify

the relative importance of objectives, or realistic targets in the case of goal programming, and relatively high computational costs [251].

Current generation MOO algorithms (MOAs) tend to focus on approximating the Pareto set and frequently employ a Pareto-based solution fitness strategy. In particular, there has been a recent deluge of research in the field of *Multi-objective Evolutionary Algorithms* (MOEAs), such as the Non-dominated Sorting Genetic Algorithm II (NSGA-II) [61]. These may be thought of as GAs which have been generalized for multiple objectives, seeking to evolve the population to be a representative approximation of the Pareto-optimal set. MOEAs are capable of searching for multiple non-dominated solutions in a single run. This makes them ideal for solving MOO problems [92].

5.2 History of Multi-objective Optimisation in WDS Design

Multi-objective WDS design came into popular use with the first well-known attempt by Halhal *et al.* [116] in 1997. Several researchers have since made important contributions, such as Xu and Goulter [268] in 1999, Todini [227] in 2000, Dandy and Engelhardt [48] in 2001, Farmani *et al.* [86, 87] in 2003 and 2005, Prasad and Park [194] in 2004, Tolson *et al.* [230] in 2004, Kapelan *et al.* [141] in 2005, Keedwell and Khu [143] in 2006, Olsson *et al.* [185] in 2009, and di Pierro *et al.* [68] in 2009.

In 1997 Halhal *et al.* [116] were the first to consider a multi-objective GA approach to address the problem of WDS rehabilitation under a limited budget. They accommodated the goals of maximizing benefit (carrying capacity, physical integrity and system flexibility) and minimizing cost, using the concepts of *Pareto rank* and *fitness sharing* introduced by Goldberg in 1989 [106]. They used the notion of *incremental solution building* in developing a *structured messy* GA (SMGA), a method which exploits the fact that for a fixed budget, only a small number of the total possible rehabilitation options may be implemented. Whereas the conventional GA represents all decision variables in a chromosome (even if many are inactive), solution representation in SMGA begins with a single decision variable (rehabilitation option), and incrementally builds longer genetic strings up to a maximum number of rehabilitation options, effectively pruning the search space enormously (reduced from order ω^{κ} to $\omega^{k}\binom{\kappa}{k}$, were κ is the number of decision variables, ω is the number of options per variable, and $k \ll \kappa$ is the maximum chromosome length). Their technique is laudable for its dramatic improvements in efficiency and utility. However, their initialization procedure requires a complete enumeration of all single-option solutions, which may not be scalable for large systems. Furthermore, their benefit function is difficult to work with as it requires the specification of weights between a number of incommensurate system properties. Finally, the fitness sharing technique requires user-specified cost neighbourhoods. Also in 1997, Savic and Walters [209] used a standard GA integrated with the EPANET hydraulic solver of Rossman [203], which they tested on three WDS benchmarks from the literature. They found that the results were sensitive to Hazen-Williams head-loss coefficients of the pipes. Halhal *et al.* [254] augmented their SMGA technique in 1999 to enable the selection of pumps and tanks, including the location of the latter at any node and inclusion of new operational constraints for tank water cycles. This was applied to create improved designs for the renowned "Anytown" benchmark (see Walski *et al.* [246]). Halhal *et al.* [116] enforced the structural restriction of recombining chromosomes of the same length only. In 2002, Wu and Simpson [262] demonstrated that this is unnecessary, applying the full version of the fast messy GA (fmGA) designed by Goldberg *et al.* [107] in 1993 to least-cost WDS design. The fmGA employs *probabilistically complete initialization* and

explicit building-block (or schemata) filtering and juxtaposition to find good gene combinations very quickly. They demonstrated dramatically improved performance for the design of a real-world Moroccan WDS. A disadvantage of their method was that it is not formulated for MOO. As opposed to the SMGA it can handle completely new designs. In 2004 Tolson *et al.* [230] combined the first-order reliability method (FORM) of Xu and Goulter [268] with a basic GA to find Pareto-optimal solutions for WDS design by means of goal-programming (numerous single-objective optimisation problems are solved for different reliability goals). This method may work for small problems, but is computationally expensive as it requires intensive calculation of derivatives and matrix inversions. Babayan *et al.* [16] responded in 2005 with a more effective integration-based methodology designed to accommodate uncertainty in demand inputs, which is still limited by its single-objective formulation.

In 2003 Farmani *et al.* [86] compared four MOEAs for WDSDO and concluded that NSGA-II [61] was the best. In 2002 Wu and Simpson [263] investigated a self-adaptive penalty function to pressurize the optimisation search towards the region at the boundary of feasibility where optimal solutions are typically located, thereby boosting performance. This self-adaptive technique was later incorporated into a multi-objective version of the fmGA by Wu and Walski [266] in 2004, and applied to the optimisation of the Hanoi network. In 2004 Nicolini [182] compared three MOEAs (ENGA, NSGA-II, and the controlled elitist NSGA-II [60]) towards the design of the two-loop network introduced by Alperovits and Shamir [9], finding that the latter two outperformed the ENGA. Prasad and Park [194] employed the NSGA in 2004 for MOO using objectives of cost and a novel surrogate measure of reliability called Network Resilience, designed to reward reliable loops in the network explicitly. They found that this produced more robust designs than previous methods. In 2005 Farmani *et al.* [88] compared the NSGA-II to the Strength Pareto Evolutionary Algorithm 2 (SPEA-II) with respect to three WDS benchmarks, and found that the latter produced improved solution quality (at the cost of increased running time). They applied these algorithms to the large Exeter WDS benchmark [84] with three objectives, and concluded that while both algorithms were somewhat successful, further research was needed in locating better Pareto-optimal sets, particularly in high-dimensional objective spaces. In another study [87], they applied the NSGA-II to the multi-objective design of the Anytown WDS [246], which includes the design and placement of tanks, using the Resilience Index as an objective. In 2005 Kapelan *et al.* [141] implemented an adapted robust version of the NSGA-II algorithm (RNSGA-II) which uses *reduced sampling fitness evaluation* (requiring fewer hydraulic simulations) to solve the *stochastic WDS design* problem with the objectives of minimizing cost and maximizing probabilistic hydraulic reliability. They employed this approach to solve the famous NYTUN problem [212] in a multi-objective fashion.

In 2009 Olsson *et al.* [185] compared three *estimation of distribution algorithms* (EDAs), namely the *Hierarchical Bayesian Optimisation* algorithm, the *Chi-square matrix method* for building block identification, and the *Univariate Marginal Distribution* algorithm, the results of which are discussed in §5.7.3. In 2009 di Pierro *et al.* [68] analyzed two modern multi-objective, hybrid algorithms, namely *ParEGO* [147] and *LEMMO* [140], with respect to the design of a real medium-size network in Southern Italy, and a real large-size network in the UK. Both algorithms were designed for dealing with expensive multi-objective optimisation problems, able to operate under a scenario of severely restricted function evaluations, and were compared with the more traditional MOEA *PESA-II* [39]. ParEGO uses a dynamic approximation technique based on Gaussian processes (Kriging) to substitute the expensive objective function evaluation, and LEMMO is a hybrid of a machine learning technique (the C4.5 rule induction algorithm of Quinlan [198]) and the NSGA-II, which learns induction rules in order to modify the child population in the hopes of speeding up the search. They demonstrated that both algorithms were

capable of dramatic speed enhancements, and although ParEGO was shown to be unsuitable for designing large systems, LEMMO could be successfully extended to the efficient design of large-scale WDSs.

A notable problem with all of these studies is the relatively few WDS benchmarks employed, making it difficult to make general claims about algorithmic performance.

5.3 Multi-objective Formulation of the WDS Design Problem

A generic formulation of the multi-objective WDS design problem is given in this section. Consider a WDS system with discrete variables $\boldsymbol{x}^{\mathrm{d}}$ (including n_{p} pipe diameters and other discrete component sizes, settings and locations), continuous variables $\boldsymbol{x}^{\mathrm{c}}$ (including tank dimensions and valve settings), n_{R} reservoirs and n nodes. A complete solution is therefore $\boldsymbol{x} = \{\boldsymbol{x}^{\mathrm{d}}, \boldsymbol{x}^{\mathrm{c}}\}$. The multi-objective WDSDO problem for such a system may be expressed as that of finding an approximation to the Pareto-optimal solution set of designs with objectives of

$$
\left.
\begin{array}{lll}
\text{minimizing} & C = C_{\mathrm{c}}(\boldsymbol{x}) + C_{\mathrm{o}}(\boldsymbol{x}, \boldsymbol{d}, \boldsymbol{e}), & [\mathit{Cost}] \\
\text{minimizing} & \widehat{P} = \widehat{P}(\boldsymbol{d}, \boldsymbol{h}, \boldsymbol{q}, \boldsymbol{o}), & [\mathit{Penalty}] \\
\text{maximizing} & R = R(\boldsymbol{x}, \boldsymbol{d}, \boldsymbol{h}, \boldsymbol{q}), & [\mathit{Reliability}] \\
\text{maximizing} & \mathcal{K} = \{K_1, K_2, \ldots, K_m\}, & [\mathit{Misc}] \\
\text{subject to} & \boldsymbol{x}^{\mathrm{d}} = [x_1^{\mathrm{d}}, x_2^{\mathrm{d}}, \ldots, x_r^{\mathrm{d}}],\ x_i^{\mathrm{d}} \in \mathcal{X}^i, & [\text{Discrete}] \\
 & \boldsymbol{x}^{\mathrm{c}} = [x_1^{\mathrm{c}}, x_2^{\mathrm{c}}, \ldots, x_s^{\mathrm{c}}],\ x_{i,\min}^{\mathrm{c}} \leq x_i^{\mathrm{c}} \leq x_{i,\max}^{\mathrm{c}}, & [\text{Continuous}] \\
 & \boldsymbol{g}(\boldsymbol{h}, \boldsymbol{q}) = \boldsymbol{0}, & [\text{Hydraulic}] \\
 & \boldsymbol{h}(\boldsymbol{d})_{\min} \leq \boldsymbol{h}(\boldsymbol{d}) \leq \boldsymbol{h}(\boldsymbol{d})_{\max}, & [\text{Pressure}] \\
 & \boldsymbol{w}_{\min} \leq \boldsymbol{w}(\boldsymbol{x}, \boldsymbol{h}, \boldsymbol{q}, \boldsymbol{d}, \boldsymbol{o}) \leq \boldsymbol{w}_{\max}, & [\text{Other}]
\end{array}
\right\} \tag{5.1}
$$

where $\boldsymbol{d}$ denotes the demand loading conditions at the nodes, $\boldsymbol{h}$ is a $1 \times n$ vector of computed nodal pressure heads, where $\boldsymbol{q}$ is a $1 \times n_{\mathrm{p}}$ vector of pipe flows, where $\boldsymbol{o}$ denotes other computed hydraulic properties such as tank capacity and water level deviations from those required at the end of a cycle. Here C denotes the cost of the network as a function of decision variables, including capital investment cost C_{c} and operational costs C_{o} (which may include the present value of energy costs $\boldsymbol{e}$ for pumping as well as maintenance and repair costs), and $\widehat{P} = \widehat{P}(\boldsymbol{d}, \boldsymbol{h}, \boldsymbol{q}, \boldsymbol{o})$ is a penalty function which uses the magnitudes of pressure and other constraint violations. Such a penalty function is commonly added to the cost function, using a penalty factor to express violations in terms of cost. The objective R denotes some reliability measure (such as Network Resilience) as a function of components, nodal heads and pipe flows, and $\mathcal{K}$ denotes a set of miscellaneous objective functions such as maximizing water quality and redundant pathways to sources.

The hydraulic constraints $\boldsymbol{g} = \boldsymbol{0}$ ensure *continuity of flow* and *zero head loss around loops*. These are hard constraints and may be satisfied intrinsically by means of a hydraulic solver which is called to evaluate the flows and pressures for every network configuration. EPANET2 [203] is employed for this purpose in this dissertation. The remaining constraints are usually considered *soft constraints*, in that slight violations may still be acceptable. For this reason these constraints are typically handled by means of a penalty function, such as the objective $\widehat{P}$. If the penalty function is zero, then all constraints are satisfied. The nodal pressure constraints specify a vector of minimum heads $\boldsymbol{h}_{\min}$, which ensure a minimum customer service level, and a vector of maximum heads $\boldsymbol{h}_{\max}$, which guards against leakage and component damage. The discrete design constraints specify that every decision variable takes on a value from a discrete set $\mathcal{X}^i = \{x_{\mathrm{o}}^1, \ldots, x_{\mathrm{o}}^{\omega}\}$, where x_{o}^i is the i-th available discrete option. Continuous constraints

impose maximum and minimum limits on $\boldsymbol{x}^{\mathrm{c}}$. The other constraints, denoted by $\boldsymbol{w}$, may include upper limits on flow velocity, tank operational constraints, and budgetary constraints. A truly generic formulation of this problem is difficult, owing to the large variety of potential objectives and constraints, and the option to incorporate phased design over time [251]. The WDSDO formulation presented here is further referred to as the *standard multi-objective formulation*, which is adapted for the various benchmark systems in later chapters.

5.4 Multi-objective Evolutionary Optimisation Concepts

The most important issues pertaining to MOEO are solution fitness assignment, diversity preservation, selection, elitism, population management, constraint handling, and variational operators. A multitude of schemes have been proposed for each of these mechanisms, demonstrating varying levels of success. The NSGA-II [61] and the SPEA-II [277] are two highly successful modern MOEAs which have demonstrated superior performance over many of their peers. As such, these two algorithms will be used to illustrate the algorithm design concepts in the following sections. These algorithms are also used as a benchmark against which to compare new methods [278].

5.4.1 Fitness Assignment

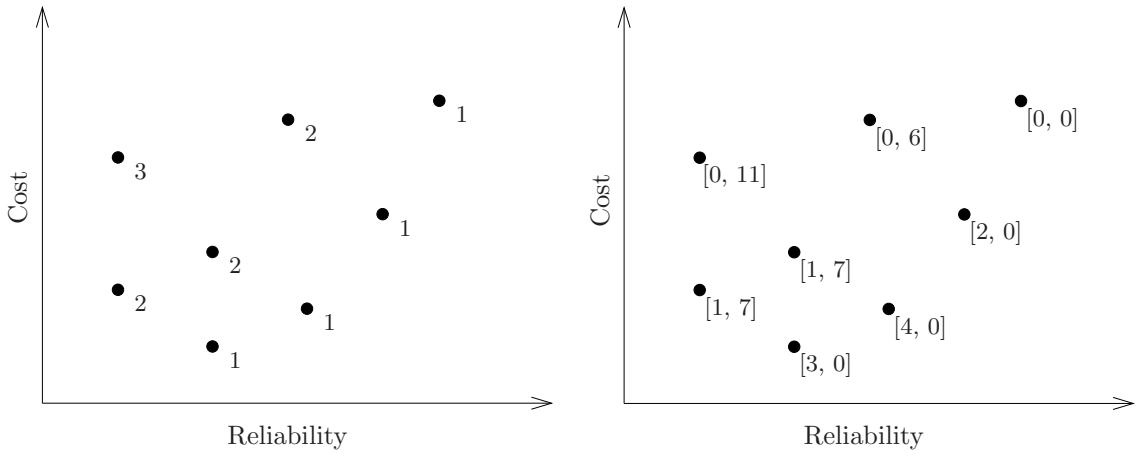
In order to benefit solutions according to the principle of survival of the fittest, one needs to adapt the concept of fitness to accommodate multiple objectives. Most modern MOEAs employ a Pareto-dominance based fitness assignment strategy, whereby the fitness of an individual is calculated in relation to all other population members using some Pareto ranking measurement. This idea was first proposed by Goldberg [106]. Several different approaches to dominance measures have been proposed, unfortunately the naming conventions differ between researchers. In this dissertation the following conventions are used: *dominance count* is the number of solutions by which an individual is dominated, *dominance strength* is the number of solutions an individual dominates, and *dominance depth*, or *Pareto-rank* is the depth of the front to which a solution belongs (generally a population may be divided into several fronts, which successively dominate each other). Various combinations of these measures are also employed [278].

NSGA-II assigns fitness as the Pareto-rank r_p of the solutions, which it determines using a fast non-dominated sorting algorithm to partition solutions into various fronts. A lower rank is better, and the solutions in current non-dominated front have rank 1 [61]. SPEA-II calculates the dominance strength value $\hat{S}$ of each solution, and then assigns the *raw fitness* $\hat{R}$ as the summed strengths of all solutions which dominate a particular individual. Here raw fitness is to be minimized [277].

A graphical comparison of these two fitness assignment schemes is provided in Figure 5.3 for eight solutions in cost-reliability space.

5.4.2 Diversity Preservation

Diversity preservation is the ideal of achieving a representative approximation of the Pareto-set, such that solutions are spread evenly along the Pareto-front. This is typically achieved by using solution density information in decision variable space or objective function space. In order to promote thorough exploration of the search space, individuals will have a greater chance of



(a) *NSGA-II, Pareto-rank fitness* (b) *SPEA-II, strength and raw fitness* $[S, R]$

Figure 5.3*: Graphical comparison of Pareto-based fitness schemes of NSGA-II and SPEA-II.*

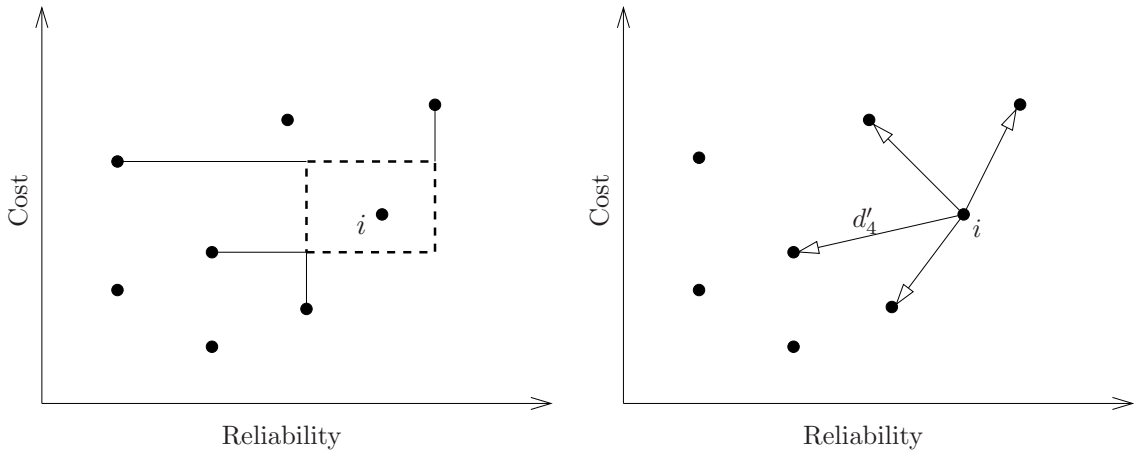
being selected as the number of individuals in their neighbourhood is decreased. Density is most frequently calculated in terms of objective space distances [278].

Kernel methods are sometimes used to quantify density. These methods define the neighbourhood of a point j in terms of a Kernel function K which takes the distance $d'_{j,i}$ to another point i as an argument. The density of solution point j is then $D'_j = \sum_{i=1}^{n} K(d'_{j,i})$. A popular approach using this technique is called *fitness sharing*, as employed by the MOGA and NPGA algorithms. This was proposed by Goldberg [106]. It requires the user to specify a *niche radius* in objective function space, so that the fitness of an individual is diluted by the number of solutions within its niche neighbourhood. The drawback of this method is the difficulty in specifying a good niche radius, which usually requires a trial-and-error approach.

NSGA-II uses a so-called *crowding distance* measure to quantify solution density. This is basically the perimeter of the *isolation hypercube* around the individual in objective function space. Solutions are sorted in increasing order along each objective axis, and the difference between a previous solution's objective function value and a next solution's objective function value is one side of the hypercube. These distances are normalized by the objective range in order to avoid disparities in magnitude. All sides of the hypercube are then added together to yield a crowding measure. Solutions with larger crowding distance values are more isolated and therefore favoured for selection. Finally, boundary solutions with the lowest and highest objective values along each axis are assigned infinite crowding distance to ensure that they are selected [61].

SPEA-II uses the distance to the k-th nearest neighbour as a density estimate. It does so by calculating the objective space distance between every solution, sorting these distances in increasing order, and selecting the k-th largest distance value d'_k. An adapted inverse, $1/(d'_k+2)$, is then added to the raw fitness of the solution, in order to discriminate between solutions of identical raw fitness [277].

A graphical comparison of these two density estimation schemes is provided in Figure 5.4 for a solution in cost-reliability space.



(a) *NSGA-II, crowding distance for solution i (perimeter of dashed rectangle)*

(b) *SPEA-II, distance of k-th nearest neighbour with $k = 4$ for solution i*

Figure 5.4: *Graphical comparison of density estimation schemes of NSGA-II and SPEA-II.*

5.4.3 Selection, Elitism and Population Management

In evolutionary optimisation, selection occurs both for offspring creation (*parent selection*), when potential parent solutions must compete for genetic mastery, and in selection for survival to the next generation (*environmental selection*). It is common practice here to employ some form of elitism so that population degradation is avoided. This is often achieved by the inclusion of an additional population, known as an archive, which typically stores only non-dominated solutions. However, given the constraint of finite memory, it is usually not possible to store all non-dominated solutions, calling for pruning mechanisms to limit population and archive sizes [278].

The NSGA-II uses binary tournament selection for reproduction, such that the solution with the lower Pareto-ranking is chosen as a parent and, in the case of identical rank, the solution with the largest crowding distance is chosen. NSGA-II does not employ an archive. Once offspring have been produced, NSGA-II calculates the Pareto-rank and crowding distance for all solutions in the combined parent and offspring population. Solutions are then selected for survival in an elitist manner, with all the rank 1 solutions being selected first, followed by the rank 2 solutions, and so forth. If including all solutions of the next rank will increase the selected population size beyond the original size N, solutions are selected in order of decreasing crowding distance until the new population size is N [61].

SPEA-II uses binary tournament selection with replacement to define a mating pool. Variational operators are then applied to the solutions in the pool to create offspring. In contrast to NSGA-II, SPEA-II employs an archive of fixed size in which it stores the best solutions. If the non-dominated solutions in each generation are too few to fill the archive, additional solutions are included in order of increasing fitness. Alternatively, if there are more non-dominated solutions than the size of the archive, a truncation operator is applied which iteratively removes individuals. In each iteration the solution with the smallest distance to an archive member neighbour is removed. If two solutions have the same smallest distance to neighbours, then their second smallest distance is compared, and so forth.

Several studies have shown that SPEA-II exhibits improved performance over NSGA-II [87],

especially in terms of diversity and the evenness of solution spread. This may be attributed to the more finely grained truncation analysis conducted by the former algorithm. However, this comes at the expense of higher computational costs, which are quadratic in the size of the population.

5.4.4 Constraint Handling

In evolutionary optimisation it is not normally possible to handle constraints explicitly, except by the *rejection method* whereby infeasible solutions are simply discarded. In the majority of cases this makes it extremely difficult to establish a feasible population, resulting in premature extinction. This is particularly relevant in WDS design, where most constraints can be considered *soft*, in the sense that it may not be possible to satisfy them in all parts of the system.

Both NSGA-II and SPEA-II are formulated for unconstrained optimisation. Constraints have therefore typically been handled by means of a *penalty term* which is added to the basic cost. This penalty is based on the magnitude of the constraint violation, multiplied by a *penalty factor* which scales the violation to the same order of magnitude as the cost. Ideally, this should result in an infeasible solution being slightly more expensive than a feasible one, so that the population evolves towards feasibility. The penalty factor would typically require *trial and error fine tuning*, although some authors have suggested methods for *auto-adaptive penalties* in the single-objective, single constraint case. The auto-adaptive technique of Afshar and Marino [5] functions by attempting to maintain a balance of feasible and infeasible solutions in each generation, effectively guiding the search along the boundary of the feasible region. This provides a significant efficiency enhancement in the single objective case. However, this idea does not scale automatically to multiple objectives, where a static penalty factor was found to be more effective. This is presumably since Pareto-optimal solutions do not necessarily lie on the boundary of the feasible region. Multiple constraints have been dealt with by means of *weighted aggregation*, which further complicates matters by requiring *weighting coefficients* which specify the relative importance of constraints. This may require an additional level of fine tuning.

An example of a penalized cost function for WDS systems is $C(\boldsymbol{x}, \boldsymbol{h}, \boldsymbol{v}) = C_{\mathrm{c}}(\boldsymbol{x}) + \widehat{P}(\boldsymbol{h}, \boldsymbol{v})$, where $C_{\mathrm{c}}(\boldsymbol{x})$ is the capital cost and $\widehat{P}(\boldsymbol{h}, \boldsymbol{v})$ is a penalty term as a function of nodal pressure heads and pipe velocities $\boldsymbol{v}$. The penalty term incorporated in this dissertation takes the form

$$\widehat{P} = p_{\mathrm{f}} \left[\sum_{j=1}^{n_p} \frac{v_j - v_{\min}}{v_{\max} - v_{\min}} - 1, \text{ if } v_j > v_{\max} + \sum_{i=1}^{n} \left\{ \begin{array}{l} -\frac{h_i - h_{\min}}{h_{\max} - h_{\min}}, \text{ if } h_i < h_{\min} \\ \frac{h_i - h_{\min}}{h_{\max} - h_{\min}} - 1, \text{ if } h_i > h_{\max} \end{array} \right. \right], \tag{5.2}$$

where p_{f} is the penalty factor. This formulation penalizes constraint violations normalized by the size of the feasible range of each constraint. This makes specifications of weighting coefficients easier by removing order disparities between different constraints and also enables more meaningful aggregation of minimum and maximum constraint violations. Such a penalty term may be used directly as an indication of feasibility, since it is zero when all velocity and head values are in their feasible ranges. It is normally unnecessary to include pressure limits for individual pipes since most pipes are rated for heads above the typical $h_{\max}$ value (although this may become relevant in the presence of transient effects). Note that velocity here refers to absolute velocity, since flow may occur in either direction. It is not common practice to specify a $v_{\min}$ value other than zero [251]. It is further recommended that pressure head limits only be enforced at nodes with non-zero demand, and velocity limits at pipes which are connected

to such nodes. It is frequently the case that pressures and flows near the sources are outside the ranges (pressure heads are lower and flows much higher) normally required by the end-user (typical in a gravity system).

The penalty factors adopted in this dissertation were determined experimentally with the assistance of an empirical expression, $p_f = C_{max}/0.01$, where C_{max} is the maximum possible cost of a network. This expression ensures that a total head deficit greater than 0.01m will result in a configuration which is more expensive than the most expensive system.

Since the WDS design problem inherently has multiple constraints, it was decided to investigate a second constraint-handling method by Oyama *et al.* [190]. This method uses the concept of *constrained domination*. Solution $\boldsymbol{i}$ is said to constrained-dominate solution $\boldsymbol{j}$ if any of the following are true:

1. Both solutions are feasible, and solution $\boldsymbol{i}$ dominates solution $\boldsymbol{j}$ in objective space.
2. Solution $\boldsymbol{i}$ is feasible, but solution $\boldsymbol{j}$ is infeasible.
3. Both solutions are infeasible, and solution $\boldsymbol{i}$ dominates solution $\boldsymbol{j}$ in constraint space (*i.e.* it is smaller in terms of at least one constraint violation and no greater in terms of any other).

If two solutions are non-constrained-dominated with respect to each other, then diversity information in objective space and/or constraint space may be used to distinguish between them. This is applicable even when both solutions are infeasible and neither dominates the other in constraint space. The two methods used in this dissertation for choosing between two solutions will be called the *crowded comparison with penalty method tournament* (Algorithm 9) and *constrained domination crowded comparison tournament* (Algorithm 10).

5.4.5 Variational Operators and Chromosome Encoding

Variational operators for evolutionary optimisation include *crossover* and *mutation* operators. The purpose of these operators is to apply variation to an existing solution pool in order to further explore the solution space with the goal of improving the average population fitness. It is therefore desirable that the operators capitalize on existing solution structure, and are able to uncover novel genetic encodings of improved fitness. Gene values and gene substrings which are present in individuals of high fitness are known as *building-blocks*, and several algorithms have been designed to work directly with building blocks. Traditional GA variational operators are based on similar mechanisms in nature; gamete exchange and gene recombination during sexual reproduction, and imperfect gene copying which is termed mutation [106].

Crossover operators allow the exchange of genetic information between two or more solutions in order to produce a new (offspring) solution. They should constitute an effective mechanism for exploring recombinations of the existing advantageous structure embodied in the current population. Mutation operators cause the random alteration of gene values, thereby providing fresh ideas and mobility to explore new regions of the search space, in the hope of finding serendipitous new encodings and avoiding premature convergence. Both operator types are associated with probabilities of occurrence. Crossover operators typically have a high probability of occurrence, as they are considered the primary mechanism for variation. Mutation typically occurs with a low probability, as too much would result in population degradation.

Algorithm 9 Crowded Comparison Tournament with Penalty Method

Input: Solutions $\boldsymbol{a}$ and $\boldsymbol{b}$ with their objective vectors and constraint violation data
Output: Superior of $\boldsymbol{a}$ and $\boldsymbol{b}$ in terms of objective and diversity information

if $\boldsymbol{a}$ is feasible and $\boldsymbol{b}$ infeasible **then**
  return $\boldsymbol{a}$
else if $\boldsymbol{b}$ is feasible and $\boldsymbol{a}$ infeasible **then**
  return $\boldsymbol{b}$
else if both $\boldsymbol{a}$ and $\boldsymbol{b}$ are infeasible **then**
  if penalty term of $\boldsymbol{a}$ is smaller than penalty term of $\boldsymbol{b}$ **then**
    return $\boldsymbol{a}$
  else
    return $\boldsymbol{b}$
  end if
else if both $\boldsymbol{a}$ and $\boldsymbol{b}$ are feasible **then**
  if $\boldsymbol{a}$ dominates $\boldsymbol{b}$ in objective space **then**
    return $\boldsymbol{a}$
  else if $\boldsymbol{b}$ dominates $\boldsymbol{a}$ in objective space **then**
    return $\boldsymbol{b}$
  else
    return solution with least objective space crowding C_{rO}.
  end if
end if

The traditional variation operators for binary-coded GAs are the *single-point crossover*, *uniform crossover* and *bitwise mutation*. Both the single- (or *multi-point*) and uniform crossovers take two parent solutions as input. In single-point crossover, a point along the genetic chromosome string is selected uniformly, the two genetic codes are cut at this point, and the heads and tails of these code strings are exchanged to form two new solutions. This operator is based on a similar phenomenon in nature, pertaining to DNA strings. It has the advantage of preserving building blocks since the solution is only broken at a single point. The uniform crossover creates a new child by assigning bit values (or whole gene values) from either parent with an equal probability (0.5). It achieves maximum allele mixing, which works well if there is no interaction between variables, but which is unsuitable for most real-world problems. A common mutation operator is the *bitwise mutation* for a binary-coded genetic string, whereby bits are flipped from 0 and 1 and *vice versa*, with a low probability (typically of the order $1/n$ where n is the number of bits) [106].

It is well-documented that the use of binary-coded strings to represent real values performs poorly when applying traditional variation operators [58]. Real-coded variables differ from those in conventional GAs in that the genes (*genotype*) and the decision variable values (*phenotype*) are identical, so that no gene decoding is necessary. Several crossover operators have been developed especially for real-coded chromosomes. The Simulated Binary Crossover (SBX) was developed by Deb and Agrawal [58] in 1994, in order to match the *search power* of the single-point crossover for binary encodings. The SBX operator is now widely used in continuous evolutionary optimisation, and is the operator of choice for the NSGA-II algorithm [61]. The SBX operator is applied between similar genes in different solutions. If $x_i^{(1,t)}$ and $x_i^{(2,t)}$ are variables denoting gene i for two different solutions at time t, and their derived offspring values at time $t+1$ are $x_i^{(1,t+1)}$ and $x_i^{(2,t+2)}$, then a *spread factor* Ψ_i is defined as the ratio of the

Algorithm 10 Constrained Domination Crowded Comparison Tournament

Input: Solutions $\boldsymbol{a}$ and $\boldsymbol{b}$ with their objective vectors and constraint violation data

Output: Superior of $\boldsymbol{a}$ and $\boldsymbol{b}$ in terms of objective or constraint dominance and diversity information

```
if a is feasible and b infeasible then
    return a
else if b is feasible and a infeasible then
    return b
else if both a and b are infeasible then
    if a dominates b in constraint space then
        return a
    else if b dominates a in constraint space then
        return b
    else
        return solution with least constraint space crowding C_rC.
    end if
else if both a and b are feasible then
    if a dominates b in objective space then
        return a
    else if b dominates a in objective space then
        return b
    else
        return solution with least objective space crowding C_rO.
    end if
end if
```

absolute difference between the offspring gene values to that of the parent gene values,

$$\Psi_i = \left| \frac{x_i^{(1,t+1)} - x_i^{(1,t+2)}}{x_i^{(1,t)} - x_i^{(2,t)}} \right|.$$

The SBX operator uses a probability distribution for Ψ_i which has been designed to mimic the spread of a binary crossover applied to a single gene, and such that 'near-parent' solutions are monotonically more likely to be chosen as offspring than solutions distant from parents. This distribution is

$$f(\Psi_i) = \begin{cases} 0.5(n_c+1)\Psi_i^{n_c}, & \text{if } \Psi_i \leq 1, \\ 0.5(n_c+1)\frac{1}{\Psi_i^{n_c+2}}, & \text{otherwise,} \end{cases} \tag{5.3}$$

where n_c is a non-negative real number called the *distribution index* which affects the probability of selecting offspring close to parent solutions. A high value of n_c yields a higher probability of near-parent solutions, thereby focussing the search around the parents, and a small value does the opposite. In the literature a value of $n_c = 2$ is typically used. Although schemes have been suggested to adapt n_c during the course of optimisation [63], they were found to perform poorly compared to a static index in MO WDS design. In order to generate offspring, a random number $u_i \in [0, 1]$ is first generated. Then a value of Ψ_i is found such that the area under the distribution (5.3) equals u_i. This value of Ψ_i is then used to generate offspring as

$$\begin{aligned} x_i^{(1,t+1)} &= 0.5\left[(1+\Psi_i)x_i^{(1,t)} + (1-\Psi_i)x_i^{(2,t)}\right], \\ x_i^{(2,t+1)} &= 0.5\left[(1-\Psi_i)x_i^{(1,t)} + (1+\Psi_i)x_i^{(2,t)}\right]. \end{aligned}$$

Differential Evolution is a simple yet powerful optimisation algorithm that uses a *differential crossover operator*, employing three solution vectors. These solutions are selected at random from the population, and the first (or base) vector is adapted by some factor of the difference vector between the next two solutions. The new solution is then compared to the base solution, and replaces it if it constitutes an improvement [152]. The Simplex Method may also be used as a crossover operator, as is the case in SCE (described in Chapter 3). Another interesting crossover operator is the *Jumping Gene* (JG) mechanism, whereby genes or gene substrings called *transposons* are allowed to change position stochastically within a chromosome. This mimics the JG transposition phenomenon discovered by Nobel Laureate, Barbara McClintock, in her work on corn genetics [171, 172]. Transposons may be cut or copied and pasted in different positions within the same genome or between two different genomes, either overwriting the genes in that position, or causing them to shift. Chan *et al.* [27] performed extensive analysis of a jumping gene MOEA, and demonstrated the superiority of this algorithm over six other popular MOEAs for the majority of 13 common benchmark problems. In preliminary trials for this dissertation, the JG operator was applied to WDS design, but the results were unsatisfactory. It is thought that JG is better suited to problems where variable structure is more homogenous or balanced. Pipe networks tend to form a natural hierarchy of large diameters near reservoirs, with smaller pipes at the system outskirts. On average, this hierarchy is disrupted by the JG operator.

Mutation operators are harbingers of disruption, typically disturbing building blocks. As such, it is essential that they are used as secondary variation mechanisms, and are applied with a low probability. Despite their semi-chaotic effect, it is still desirable that mutation is 'smooth', such that mutated gene values have a higher probability of occurring close to the original values. The classical bitwise mutation operator suffers from the problem of *Hamming cliffs*, where certain bit flips may cause extreme alterations in value, owing to each bit having a different order of magnitude [51].

For this dissertation a novel mutation operator, based on the triangular distribution (TD), was developed for real and integer coded genes. The TD function involves three parameters a, b, and c, where a is the lowest possible value of a range of variation, b is the mode, and c is the highest value attainable. The probability density function (PDF) of the TD is graphed in Figure 5.5(a), and is formulated as

$$f(x|a,b,c) = \begin{cases} \frac{2(x-a)}{(c-a)(b-a)}, & \text{for } a \leq x \leq b, \\ \frac{2(c-x)}{(c-a)(c-b)}, & \text{for } b \leq x \leq c. \end{cases}$$

The cumulative distribution function (CDF) associated with the TD is shown in Figure 5.5(b), and is given by

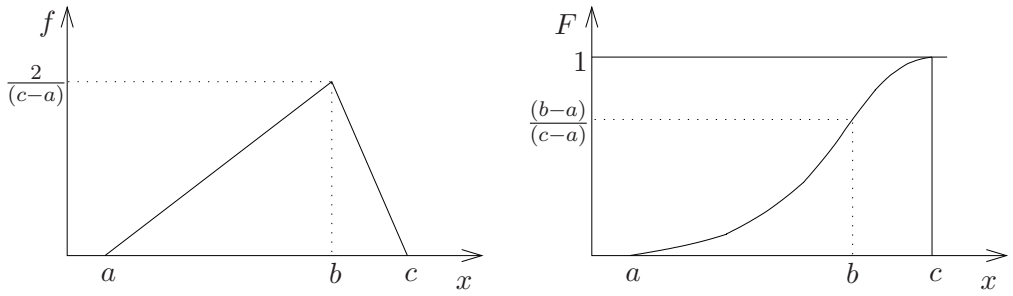
$$P(x|a,b,c) = \begin{cases} \frac{(x-a)^2}{(c-a)(b-a)}, & \text{for } a \leq x \leq b, \\ 1 - \frac{(c-x)^2}{(c-a)(c-b)}, & \text{for } b \leq x \leq c. \end{cases}$$

The TD is applied with the current gene value assigned to the mode b, with a and c taken as the lower and higher limits of gene values. In order to mutate a gene $x \rightarrow x'$, a random number $u \in [0,1]$ is generated and the inverse CDF applied as follows,

$$x' = \begin{cases} a + \sqrt{(c-a)(x-a)u}, & \text{for } u \leq \frac{(x-a)}{(c-a)}, \\ c - \sqrt{(c-a)(c-x)(1-u)}, & \text{for } u > \frac{(x-a)}{(c-a)}. \end{cases}$$

Polynomial mutation is a popular operator developed in conjunction with the SBX method [58]. In 2008 Deb and Tiwari [64] presented a modified form of polynomial mutation

$$\delta_1 = \frac{x_p - x_\ell}{x_u - x_\ell}$$



(a) *Triangular Distribution PDF, f* (b) *Triangular Distribution CDF, F*

Figure 5.5*: Probability density and cumulative density functions for the Triangular Distribution.*

$$
\begin{aligned}
\delta_2 &= \frac{x_u - x_p}{x_u - x_\ell} \\
\delta_q &= \begin{cases} \left[2r + (1-2r)(1-\delta_1)^{n_m+1}\right]^{\frac{1}{(n_m+1)}} - 1 & \text{if } r < 0.5 \\ 1 - \left[2(1-r) + 2(r-0.5)(1-\delta_2)^{n_m+1}\right]^{\frac{1}{(n_m+1)}} & \text{if } r \geq 0.5 \end{cases} \\
x_c &= x_p + \delta_q (x_u - x_\ell),
\end{aligned}
$$

for their *Omni-optimizer* algorithm which improves on its predecessor. Here x_p is the parent solution (a real variable), x_c is the child solution (a real variable), x_ℓ is a lower bound on the solution variable, x_u is an upper bound on the solution variable, r is a random number uniformly distributed in [0, 1], and η_m is the index for polynomial mutation. This formulation uses two different probability distributions in two different regions (x_ℓ to x_p and x_p to x_u) of the search space, thereby assigning a non-zero probability of generating an offspring in the entire search space, even if the parent is near a boundary value [64].

Another example of a mutation operator for real-coded chromosomes is *Non-uniform mutation*. Here, offspring y_i^{t+1} is generated from parent x_i^{t+1} as

$$y_i^{t+1} = x_i^{t+1} + \tau(x_i^{\mathrm{u}} - x_i^{\mathrm{l}})(1 - u_i^{(1-t/t_{\max})^b}),$$

where x_i^{u} and x_i^{l} are the upper and lower limits on the values of gene x_i, where the factor τ is 1 or -1 with an equal probability, where $u_i \in [0, 1]$ is a random number, where t is the generation counter, where $t_{\max}$ is the maximum number of generations, and where b is a user-defined parameter. This operator has the property of increasing the probability of generating near-parent solutions as time progresses [200].

A modern revolution in population based optimisation which departs from classical genetic operators was spawned by the so-called *estimation of distribution algorithms* (EDAs) or *probabilistic building block algorithms*. These algorithms attempt to build probability distributions for chromosome values from the current population, sometimes incorporating gene linkage information by means of joint probability distributions. These distributions are then used to sample gene values in order to create new offspring solutions. A simple *Univariate Marginal Distribution* (UMD) algorithm is described later in this chapter, as well as a variation on this, the *Partitioned UMD*, developed by the author. Another interesting approach is the so-called *Fast Messy Genetic Algorithm* which attempts to identify good gene schemata or building-blocks explicitly, by employing a messy or partial representation of chromosomes [107]. These building blocks are then recombined using traditional operators with the help of a template solution to fill in

the missing information. This may be generalized to MOO by considering a diverse archive of non-dominated template solutions.

Chromosome encoding obviously restricts which operators may be applied. In this dissertation only integer coded genes were employed, however, the software model was designed to additionally allow for real coded genes. An emphasis was therefore placed on using variational operators that are applicable to both integer and real gene encodings. Hence, the answer was to investigate real-coded operators combined with a discretization scheme in order to map results to integer values.

In preliminary trials on the simplified pipe network optimisation problem, the SBX operator with simple rounding (applied at random to each corresponding gene pair with a probability of 0.5) was found to outperform single- and double-point crossover. Triangular mutation was similarly found to be superior to polynomial mutation for use in WDSDO. These variation schemes were therefore employed in this dissertation for many of the algorithms tested (*e.g.* within the NSGA-II and SPEA-II frameworks).

5.4.6 Population Sizing

Traditionally population sizing for genetic algorithms was performed on an *ad hoc* basis, with a size in the region of 50–100 solutions most commonly used [164]. Population size is a critical parameter with respect to the correct functioning of evolutionary algorithms. This is due to the requirement for sufficient genetic schemata (solution building blocks), allowing a multi-modal search to cover the search space adequately. This means that the required population size depends on the size and complexity of the problem. If the population is too small, then the algorithm will become trapped in localized regions of the search space. However, if the population is too large, then the algorithm will unnecessarily waste computational resources [165]. The majority of current MOEAs and other multi-objective metaheuristics employ static population sizes, and hence sensitivity analysis should be conducted with respect to the population size used, since it will have an effect on the performance of a particular algorithm.

In 1999 Harik and Lobo [119] introduced the *parameter-less GA* that incorporates adaptive population sizing using multiple competing populations of consecutively increasing sizes, which undergo optimisation simultaneously and independently. At any one time there are two populations, the larger of which is double the size of the smaller. Populations are evolved for an equal amount of time (or perhaps benefiting the smaller population by some factor $m > 1$). The average fitnesses of the populations are compared, and if the larger population has a superior average fitness, then the smaller population is discarded and a new population is introduced whose size is again doubled. Harik and Lobo argue that since the smaller population receives the same (or a larger) allotment of computational resources, its average fitness being lower is compelling evidence that it contains insufficient building blocks. This process is repeated, evolving the larger population for an equivalent period of time until either the smaller population is again discarded or outperforms the larger one — evidence that the larger population is wasting computational resources unnecessarily. The current smaller of the two populations then represents the 'optimized' population size [164]. For multi-objective optimisation some other measure of fitness is required which summarizes the quality of the entire population.

In this dissertation the following convention with respect to static population sizing has been applied: For each algorithm-benchmark pair, the above procedure was executed by starting with populations of size $2^5 = 32$ and $2^6 = 64$, and for larger populations progressively incrementing the size by doubling the previous size. A time budget consisting of the time to conduct 20 000

hydraulic simulations of a particular benchmark was used for each population. The measure of fitness employed is the population *hypervolume* (a measure of the area (or volume) dominated by a population in multiple objective space). The sizing process was repeated 10 times with randomized initial populations for each algorithm-benchmark pair. It was found that the 'optimal' population size of an algorithm for a particular benchmark varied from sizing to sizing. However, there was a tendency for a particular population size to dominate in the sizing outcomes for each algorithm-benchmark combination. In the case of a tie, the population size that produced the higher average hypervolume was chosen. To analyze the robustness of this sizing scheme, trials were performed using the times for 10 000 and 30 000 hydraulic simulations respectively. Here it was found that different 'optimal' population sizes were occasionally produced for a particular algorithm.

There are two primary concerns with population sizing, namely (1) that the population is large enough to provide sufficient representative schemata, and (2) that a particular algorithm employs the population size that optimizes its performance. The first concern is addressed by the sizing method above. The optimal size for algorithmic performance is a more thorny issue. There are two primary components of algorithmic performance: speed and solution quality. There are also two general observations to be made regarding the effect of population size. Firstly, in general, the larger the population, the slower the convergence to a relatively stable Pareto-optimal approximation set. Secondly, it was observed empirically that larger populations tend to produce higher quality Pareto-set approximations once they have converged. Therefore, an algorithm with a smaller 'optimal' population size will usually have a speed advantage over another algorithm but produce approximation sets of inferior quality.

By assigning a similar population size to all algorithms and conducting *convergence analyses*, one can observe the relative speed of the algorithms. Then, taking the longest convergence time amongst the algorithms, one may allow all algorithms to execute for a similar length of time in *fair time trials*, and proceed to compare the quality of the solutions produced (without concerns about differences in the size of the approximation sets produced by the different algorithms).

In this dissertation, algorithms tested were divided into two classes, depending on their population size requirements (the classes of EDAs and MOEAs). For each class, the sizing methodology described above was applied and the largest 'optimal' population size found amongst all sizing trials for a particular benchmark was identified. This population size was then employed for *all* algorithms in a class in both the convergence trials and the fair time trials.

5.4.7 Epsilon-domination and Grid-based Optimisation Schemes

Epsilon (ϵ-) dominance is a variation on the ordinary Pareto-dominance concept with several different interpretations. The most common definition is that an $n \times 1$ vector $\boldsymbol{u}$ ϵ-dominates a vector $\boldsymbol{v}$ if and only if $(1+\epsilon)u_i \geq v_i \quad i = 0, \ldots, n$ (maximisation problem). This is commonly used in solution archiving schemes, by preserving the subset of non-dominated solutions that ϵ-dominates the rest of the Pareto-front. What makes this technique useful is that the population will converge to an ϵ-approximation of the Pareto-optimal set whose size is finite and depends on the value of ϵ. This is extremely beneficial in cases where the true Pareto-optimal set grows exponentially with the size of the problem. Horoba and Neumann [125] have, however, demonstrated cases where the use of such an $\epsilon-$dominance scheme can impede the optimisation process significantly.

Epsilon dominance may also be used in the simplified context of cell/grid-based dominance, whereby the user is able to specify the desired precision for each objective in a multi-objective

problem. For example, consider a bi-objective problem with objectives of cost C and reliability R. One may then specify a precision ϵ_C for the cost objective (*e.g.* $\epsilon_C = \text{R}10\,000$) and precision ϵ_R for the reliability objective (*e.g.* $\epsilon_R = 1\%$). The notion of ϵ-dominance then applies a hypergrid in objective function space, with cell sizes defined by user-specified ϵ-values. The ϵ-dominance scheme uses this hypergrid to determine which solutions are dominant, based on entire grid cells rather than on individual objective vectors. Solutions are mapped to ϵ-cells, and associated with the most non-dominated corner (NDC) of the given cell. Solutions in other cells whose NDC coordinates are dominated by the NDC coordinates of the given solution are said to be ϵ-dominated. Strict-ϵ-domination would require that both NDC coordinates be dominating. If only a single solution is retained per cell, then the one with the smallest Euclidean distance to the NDC may be retained. In this manner it is possible to reduce the number of population members under consideration significantly (*e.g.* during the selection phase), in the hope of achieving computational cost savings. However, one must balance this saving with the reality of having to update the supporting data-structures, and the loss of precision intrinsic to this scheme.

Another use of such a grid scheme is to store solution quality information in the form of one or more epsilon hypergrids, instead of associating it with the solutions directly, such that the solutions interact with the objective-space hypergrid rather than directly with each other. Potential computational savings may then be achieved, depending on the precision used, since a very fine precision will require numerous grid cells to be updated for every solution during each generation. This is discussed in more detail in the context of the *Dynamic Multi-objective Algorithm* later in this chapter.

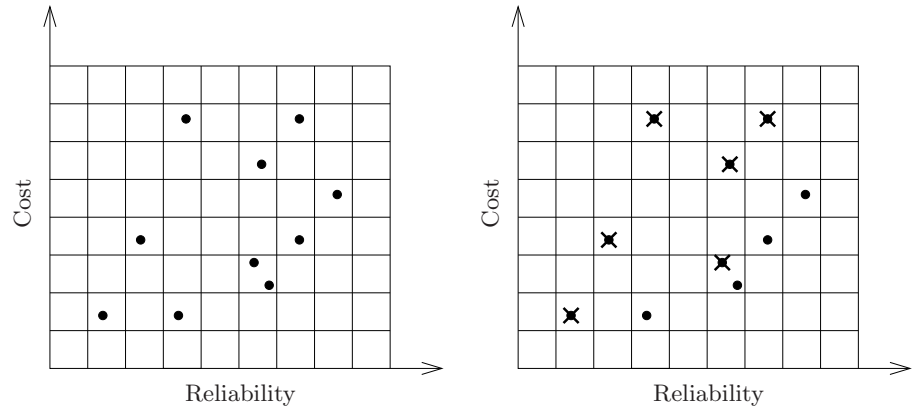
The notion of cellular ϵ-domination is demonstrated in Figure 5.6.

Since there is only one solution per grid cell, this method has the further advantage of preventing solution clustering, and ultimately producing a more evenly distributed approximation set. Larger values of ϵ result in a coarser grid, with fewer solutions, and faster processing times. Furthermore, an epsilon-hypergrid is a useful concept since it has the potential of replacing cryptic algorithmic parameters with objective axis precisions, which users may able to understand and use more easily. This also enables the use of various adaptive mechanisms. For instance, it may be possible to conduct a coarse search initially, and gradually increase the ϵ-precision as the search progresses.

5.4.8 Adaptive Population Sizing

Adaptive population sizing is featured in several MOEAs. In 2007 Kollat and Reed [149] conducted a performance evaluation of two such algorithms, namely ϵ-NSGA-II and ϵ-MOEA, which they compared to the ordinary NSGA-II. The algorithms work by storing an archive of ϵ-non-dominated solutions and conducting a series of connected optimisation runs. The search is periodically restarted with a larger population, and the initial population is preconditioned by seeding it each time with archive solutions, along with randomly generated solutions. The size of the archive dictates the size of the population. For example, ϵ-NSGA-II uses a population four times larger than the current archive size, requiring 75% randomly generated solutions. This is effective in introducing new chromosome building blocks with each restart, and preventing premature convergence to local Pareto-fronts. The following scheme for automatic termination is used by ϵ-NSGA-II:

1. The ϵ-archive is updated at the end of every generation using the current population.



(a) *Epsilon domination scheme — pre elimination*

(b) *Epsilon domination scheme — post elimination (crossed out solutions have been eliminated)*

Figure 5.6*: A cellular ϵ-dominance scheme applied to solution selection using a hypergrid in objective function space.*

2. At the end of each lag window of c generations, the difference in size of the archive between the start and finish of the lag window is examined, and a new sub-run is initiated (with a newly seeded population) if it has failed to grow more than a specified percentage Δ. A new sub-run is also initiated if a maximum number of generations $G_{\max}$ is reached (Kollat and Reed [149] recommended values of $c = 10$, $\Delta = 0.1$, and $G_{\max} = 50$).

3. If the archive size fails to grow more than $\Delta\%$ between successive sub-runs, then the number of archive replacements during the sub-run is examined. If this fails to be larger than a specified percentage of the archive (*e.g.* 2%), then the search terminates; otherwise a new sub-run is initiated.

ϵ-NSGA-II was found to outperform the other algorithms, being able to accommodate difficult problems that the other methods failed to solve. A useful feature of this algorithm is that its archiving and adaptive population sizing schemes are applied *after the fact*, so that they may be used in almost any MOO scenario. Detailed descriptions of ϵ-dominance are given by Laumanns *et al.* [161] and Deb *et al.* [62]. A preliminary investigation of ϵ-NSGA-II in this dissertation for use in WDSDO failed to demonstrate significant improvements above the ordinary NSGA-II with replacement of duplicate solutions by random alternatives.

Other mechanisms for adapting population size have been proposed, such as that of the Dynamic Multi-objective Evolutionary Algorithm (DMOEA) [271] which uses explicit rank and density goals in cells of an epsilon hypergrid, coupled with a population growth and decline strategy. Another technique is simply making the population size a function of the number of non-dominated solutions, such as in the FastPGA algorithm [82] (although this may cause exponential population growth). A better method is to make the population size a function of the current ϵ-Pareto-front size. Possibly the most rationally sound method of population sizing is through a race of multiple populations of differing sizes (typically sized in powers of two), determining the best population as the one with the best average fitness. This approach is adopted in the parameter-less GA [164]. The major disadvantage of this method is its high computational expense.

5.5 Performance Evaluation for MOAs

In 2000, Zitzler *et al.* [276] noted three main aims in MOO: (i) to maximize the number of elements of the Pareto-optimal set found, (ii) to minimize the distance of the Pareto-front produced by an algorithm with respect to the global Pareto-front, assuming its location is known, and (iii) to maximize the spread of solutions found (*i.e.* maximize how smoothly and uniformly a Pareto approximation set is distributed). Another evident goal is that the algorithm should be able to provide such results within a reasonable time-frame.

Numeric measures of solution quality are subject to limitations. Firstly, it is often impossible to determine the exact location of the true Pareto-front in practical, real-world problems (as is the case in WDS optimisation), which compels the user to conduct comparative analysis between different approximation sets. Also, the goal of global convergence requires that the sequence of Pareto-set approximations $\boldsymbol{A}^t$ produced by the algorithm converges to the true Pareto-set $\boldsymbol{D}^*$ as the number of generations t grows. However, it is intuitively apparent that this is not practically feasible as it requires unlimited memory resources (the cardinality of the Pareto-set may be arbitrarily large [278]).

Secondly, the nature of the three aspects listed above makes their simultaneous assessment by a single performance measure tricky, especially one which provides only a unary quantification of some quality aspect. Therefore, it is recommended that more than one quality measure is used and, in particular, that at least one *binary performance indicator* be used, which allows the direct quantitative comparison of two approximation sets [278]. Furthermore, performance measures used should be *Pareto-compliant*. Pareto-compliance means that an indicator should only give preference to one approximation set, $\boldsymbol{A}$, over another set, $\boldsymbol{B}$, if $\boldsymbol{B}$ does not weakly dominate $\boldsymbol{A}$. Most unary indicators are non-Pareto-compliant [148]. One of the popular exceptions is the *hypervolume* metric [276] (discussed later). However, these performance assessment measures may still be unsatisfactory, such as when algorithms are equally good at locating solutions in certain regions of the search space, but vary dramatically in other regions. The simplest assessment method remains a graphical comparison of results, which may reveal additional information hidden by numeric measures. It is desirable to include graphics in addition to ordinary quality measures, although this may only be possible for at most tri-objective problems.

Other issues regarding performance measurement for MOO are: convergence and optimisation termination criteria, optimisation parameter settings, equitable comparative analysis, and solution quality measures. These shall be discussed in the following subsections.

5.5.1 Convergence

Since the true Pareto-set is not available, convergence must be defined in terms of convergence to a static population. However, since this may take an exorbitant length of time, it is reasonable to define convergence as the event that the percentage improvement in the approximation set falls below a specified threshold for a required number of consecutive generations. In this dissertation a threshold of 0.05% change in hypervolume per generation is used, and this must occur for two hundred consecutive generations for the algorithm to have 'converged'.

5.5.2 Parameter Tuning

Most optimisation algorithms have a number of parameters which may be tuned to adjust algorithm performance. One focus in this dissertation is using algorithms which require minimal

parameter setting by users. This is achieved either by employing algorithms which are robust under a wide range of different parameter values (such GAs), or by using algorithms which self-adapt parameter values, such as during adaptive population sizing. Where this is not possible, the general approach in this dissertation is to consider a range of parameter values for each algorithm, and conduct preliminary sensitivity analysis in order to determine which set of values to implement in the final algorithms. The parameter settings for the specific algorithms used in this dissertation are provided and motivated in Chapter 6.

5.5.3 Algorithm Comparison

It is traditional to compare the performance of evolutionary algorithms by executing different algorithms with similar population sizes for an equal number of generations [16, 209, 218]. This allows results from different studies to be compared independently of the computer software and hardware used. However, this method becomes impractical when using algorithms which require different population sizes (such as EDAs), or adaptive population sizing (ϵ-NSGA-II). Additionally, it may be highly unfair, considering that an evolutionary generation may entail an arbitrary amount of numerical processing, possibly including a local search subcomponent, resulting in very different generational processing times from one algorithm to the next. Numerous studies have erroneously reported results where algorithms are compared on this basis, where one is an order of magnitude worse than another in terms of execution time.

One solution might be to impose a maximum budget of objective function evaluations (or hydraulic simulations in the case of WDS design), particularly if the processing bottleneck is caused by this stage of the optimisation. However, there may also be a trade-off between the number of objective function evaluations and computation in the rest of an algorithm. It is conceivable that an algorithm with a high degree of function evaluations is relatively efficient in all other respects, resulting in unfair penalization.

Therefore, the only truly fair mechanism would seem to be to impose a time budget, such that an algorithm must do its best within an allotted time-frame. Of course, one could argue that this is only valid if all algorithms demonstrate similar behaviour with regards to their fitness improvement profiles. An algorithm with a linear average fitness improvement profile (*e.g.* a greedy heuristic) may outperform one with an exponential improvement profile (*e.g.* a GA) if the time budget is too brief.

One solution to this conundrum is to execute all algorithms being compared independently until they converge (as defined in §5.5.1), and then to use the longest average running time as the maximum time limit. This may be excessive if certain algorithms are very slow, although it may be argued that these algorithms are not of interest for practical implementation. This practical philosophy is applied in this dissertation, employing a 90% percentile with respect to running times, and discarding the top ten percent of longest average running times.

This is in line with the requirement of maximizing efficiency in what is already a very computationally demanding problem. In order to make this *time to convergence* comparison technique more robust, several runs (*i.e.* 30) of each algorithm are executed in order to calculate their average running times. This method may also be used to transfer experiments to a new computer platform (*i.e.* attaining new time limits for the new system), provided similar convergence thresholds are employed.

5.5.4 Solution Quality Assessment

In this dissertation, three measures of solution quality are employed. The use of multiple performance measures, including ones that are Pareto-compliant, is recommended by Knowles *et al.* [148] in an extensive tutorial on performance assessment for MOEAs.

The first performance measure is a *dominance rank* quantifier, advocated in [148] as the first step in comparing algorithm results. The Pareto-compliance of dominance ranking follows directly from its definition. The second measure is the unary *hypervolume metric*, which is extremely useful in that it simultaneously quantifies dominance and spread, and is also Pareto-compliant. The third measure is a *novel unary metric for diversity / spread quantification* based on an ϵ-dominance scheme. This third measure is non-Pareto-compliant, and should therefore be considered only after the first two. The order of these performance measures conveys in some sense the order of their importance. An algorithm producing approximation sets with the highest average dominance ranking for a statistically significant number of trials is almost certainly superior.

The *dominance rank* quantifier uses a *binary weak-domination indicator* which compares one approximation set to another and determines whether or not it weakly dominates it. One may say that set $\boldsymbol{A}$ is better than set $\boldsymbol{B}$ if it weakly dominates $\boldsymbol{B}$, and is dissimilar from $\boldsymbol{B}$. Mathematically this is expressed by $\boldsymbol{A} \triangleleft \boldsymbol{B}$. A given approximation set $\boldsymbol{A}$ is compared to every other approximation set in the total pool $\mathcal{P}$ of approximation sets produced by the various algorithms. The dominance rank of $\boldsymbol{A}$ is then

$$\text{rank}(\boldsymbol{A}) = 1 + |\{\boldsymbol{B}_i \in \mathcal{P} : \boldsymbol{B}_i \triangleleft \boldsymbol{A}\}| \,.$$

The lower the ranking, the better the approximation set is with respect to the entire set pool. Three approximation sets along with their rankings are shown in Figure 5.7(a). The rank must be calculated for each set $\boldsymbol{A}_i^j,\ i = 1, \ldots, m$ produced by each algorithm $j = 1, \ldots, n$, forming a set of rank samples for each algorithm,

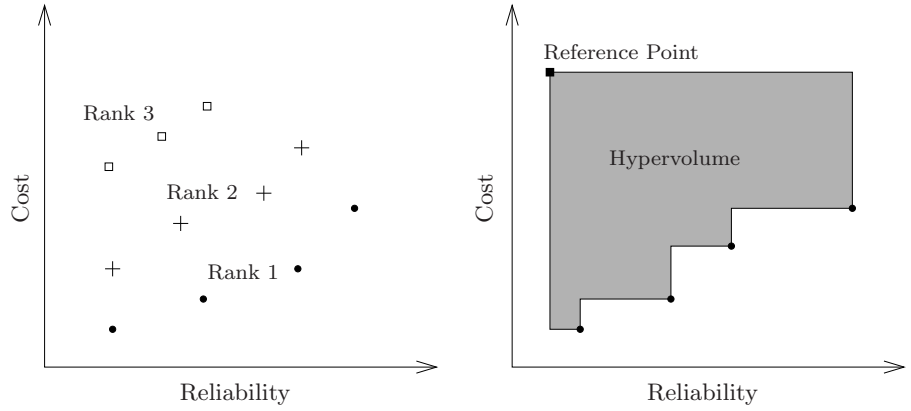
$$\mathcal{D}_{\text{R}} = \left[\left\{\text{rank}(\boldsymbol{A}_1^1), \text{rank}(\boldsymbol{A}_2^1), \ldots, \text{rank}(\boldsymbol{A}_m^1)\right\}, \ldots, \{\text{rank}(\boldsymbol{A}_1^n), \text{rank}(\boldsymbol{A}_2^n), \ldots, \text{rank}(\boldsymbol{A}_m^n)\}\right] .$$

A statistical rank test may be used to determine whether the ranks assigned to one algorithm are significantly smaller than the ranks assigned to another. In this dissertation, the average and standard deviation of dominance rank was reported for the approximation sets produced by each algorithm.

The unary hypervolume metric by Zitzler and Thiele [279] is a fine-grained analysis which measures the total hypervolume of the objective space dominated by a given approximation set, relative to a reference point. Higher hypervolumes are more desirable, since more space is dominated. This metric has the advantage of representing both closeness to the true Pareto-front and solution diversity. Any reference point used should be dominated by the entire set of known solutions [279]. For a maximisation problem with positive valued objectives, the natural choice of reference might be the zero vector. Figure 5.7(b) shows a hypervolume region for a given approximation set in cost-reliability space. It may be sensible to normalize hypervolumes to the range [0,1] for presentation purposes, as was done in this dissertation by normalizing the hypervolume of each approximation set by the hypervolume of the best Pareto-set found by combining every approximation set produced by all algorithms. Average dominance rank is the primary measure employed to rank algorithms in this dissertation (taking standard deviation of dominance rank into account in the case of a tie), with average hypervolume attainment used to distinguish between two algorithms with identical dominance statistics.

The unary diversity metric used is defined simply as the size of the final ϵ-archive for each approximation set. If the algorithm under consideration does not use an ϵ-archive, then each output approximation set is simply inserted into one of identical precision at the end of the optimisation run. This archive size is equivalent to the number of evenly spaced ϵ-dominance hypergrid cells containing solutions. If two algorithms consistently converge to a common front (*e.g.* they both have dominance rank 1), but one algorithm has a larger ϵ-archive, then its solutions cover more of the Pareto-front and are therefore better distributed.

Finally, results shall also be presented graphically, in the form of so-called *attainment fronts*, which includes all the non-dominated solutions from the combined approximation sets of an algorithm. This is shown as a stepped hypervolume region, as per Figure 5.7(b).



(a) *Dominance rank* (b) *Hypervolume metric*

Figure 5.7*: Multi-objective solution quality assessment mechanisms.*

5.6 Multi-objective Evolutionary Algorithms

MOEAs are population based search procedures designed to produce an approximation to the Pareto-optimal set of solutions in multiple objective space. A generic algorithmic formulation for MOEAs may appear as follows:

1. *Initialization*: Generate an initial population (randomly or otherwise) and evaluate their fitness.

2. *Parent Selection*: Select a mating pool from the current population, favouring solutions of greater Pareto-dominance.

3. *Solution Creation*: Create a population of offspring solutions from the mating pool using recombination and/or mutation operators.

4. *Fitness Evaluation*: Evaluate the Pareto-based fitness of offspring solutions.

5. *Environmental Selection*: Select solutions for survival to the next generation from the combined parent and offspring populations, favouring solutions of greater Pareto-dominance. Update the current population. This stage may include interaction with a non-dominated archive or the use of diversity preservation mechanisms.

6. *Termination Check*: If the termination condition is satisfied, then stop and output the current population / archive; otherwise return to Step 2.

The working of three popular MOEAs mentioned previously, namely NSGA-II, SPEA-II, and Differential Evolution is described in this section. Each of these algorithms is employed for WDSDO in the following chapter of this dissertation.

5.6.1 NSGA-II

The Non-dominated Sorting Genetic Algorithm II (NSGA-II) was developed in 2002 by Deb *et al.* [61] in an attempt to improve upon the performance of its predecessor the NSGA, as well as several competing MOEAs. Its design objectives were to reduce the number of optimisation parameters, improve its elitism scheme, and improve upon the computational complexity of the non-dominated sorting algorithm used in most MOEAs at the time (which was $\mathrm{O}(MN^3)$, where M denotes the number of objectives and N denotes the population size). It is still considered a state-of-the-art MOEA, and has outperformed many of its brethren for numerous optimisation problems. NSGA-II is also frequently used as a benchmark in MOO studies [86, 88, 141, 185, 194].

Farmani *et al.* [86] conducted a study in 2003 in which they compared the performance of four modern MOEAs — the Multi-objective Genetic Algorithm (MOGA), the Pareto Archived Evolution Strategy (PAES), the Niched Pareto Genetic Algorithm (NPGA), and NSGA-II — in solving the deterministic multi-objective WDS design problem. They concluded that the NSGA-II is the best of the four.

The NSGA-II is notable for the relatively low computational complexity of its Pareto-rank sorting algorithm, called the Fast Non-dominated Sorting Algorithm (FNSA), which is $\mathrm{O}(MN^2)$[1]. The FNSA works by first calculating the dominance count $d_{\boldsymbol{i}}^{\mathrm{c}}$ for each solution $\boldsymbol{i}$ (*i.e.* the number of solutions which dominate $\boldsymbol{i}$), and $\mathcal{S}_{\boldsymbol{i}}$, the set of solutions which $\boldsymbol{i}$ dominates. This stage requires $\mathrm{O}(MN^2)$ comparisons. The solutions in the first non-dominated front have a dominance count of zero, and are assigned Pareto-rank 1. The second stage of the sorting begins by placing all solutions having a dominance count $d_{\boldsymbol{i}}^{\mathrm{c}} = 0$ in a separate set $\mathcal{F}_1$, and cycling through this set. For each $\boldsymbol{i} \in \mathcal{F}_1$, the algorithm visits each solution $\boldsymbol{j} \in \mathcal{S}_i$, and decrements its $d_{\boldsymbol{j}}^{\mathrm{c}}$ value, effectively discounting the effect of $\boldsymbol{i}$ on $\boldsymbol{j}$'s dominance count. In this manner, all the rank 1 solutions' effects on dominance count are discounted for the remainder of the population. All rank 2 solutions now have a dominance count of zero. These are then placed in a separate set $\mathcal{F}_2$, and the algorithm proceeds by cycling through this set. This process continues for successive fronts until all solutions have been ranked. This stage requires at most $\mathrm{O}(MN^2)$ steps. A pseudocode listing of the FNSA appears in Algorithm 12 [61].

Suppose there are h solutions in a population. In order to calculate the crowding distance density measure one must sort the solutions in order of increasing value along each objective axis. Let $Y[i]|k$ represent the objective function value of the i-th solution in the sorted list for the k-th objective. The crowding distance i_{dist} for the solutions at the endpoints, $Y[1]|k$ and $Y[h]|k$, are assigned a value of infinity, and the crowding distance of the intermediate solutions i are incremented by the distance between their neighbours on either side, that is a value of $i_{\mathrm{dist}}|k + (Y[i+1]|k - Y[i-1]|k)/(k_{\max} - k_{\min})$ is assigned to $i_{\mathrm{dist}}|k$, where each distance value has been normalized by the length of objective k's range. Crowding distance is accumulated for

[1] It should be noted that the latest algorithm for non-dominated sorting is able to achieve a computational complexity of $\mathrm{O}(MN \log_{M-1} N)$ [138].

Algorithm 11 Non-dominated Sorting Genetic Algorithm II (NSGA-II) [61]

Input: A MOO problem where solutions are an assignment of values to decision variables $\boldsymbol{x}$, a population size N, a set of constraints and violation magnitude functions, M objective functions to produce entries for the $M \times 1$ objective vector $\boldsymbol{y}$, a maximum number of generations $G_{\max}$.

Output: An approximation of the Pareto-optimal solution set in multi-objective space, $\boldsymbol{A}^*$.

1: Randomly generate an initial population of solutions $\boldsymbol{P}_0$ of size N.
2: Rank and sort $\boldsymbol{P}_0$ using the FNSA [Algorithm 12].
3: Calculate the crowding distance of the solutions in $\boldsymbol{P}_0$ [Algorithm 13]
4: Create population $\boldsymbol{Q}_0$ of size N using binary tournament selection (with the crowded comparison operator $\succ_{\mathrm{c}}$) from $\boldsymbol{P}_0$, crossover and mutation.
5: $t \leftarrow 0$
6: **while** $t < G_{\max}$ **do**
7: $\quad \boldsymbol{R}_t \leftarrow \boldsymbol{P}_t \cup \boldsymbol{Q}_t$
8: $\quad$ Partition $\boldsymbol{R}_t$ into fronts $\mathcal{F}_1, \mathcal{F}_2, \ldots$ by means of the FNSA.
9: $\quad \boldsymbol{P}_{t+1} \leftarrow \emptyset$ and $i \leftarrow 1$
10: $\quad$ **while** $|\boldsymbol{P}_{t+1}| < N$ **do**
11: $\quad\quad$ **if** $|\mathcal{F}_i| + |\boldsymbol{P}_{t+1}| <= N$ **then**
12: $\quad\quad\quad \boldsymbol{P}_{t+1} \leftarrow \boldsymbol{P}_{t+1} \cup \mathcal{F}_i$
13: $\quad\quad$ **else if** $|\mathcal{F}_i| + |\boldsymbol{P}_{t+1}| > N$ **then**
14: $\quad\quad\quad$ Calculate crowding distance for all solutions in $\mathcal{F}_i$.
15: $\quad\quad\quad$ Sort $\mathcal{F}_i$ members in order of decreasing crowding distance.
16: $\quad\quad\quad \boldsymbol{P}_{t+1} \leftarrow \boldsymbol{P}_{t+1} \cup \{$ the first $(N - |\boldsymbol{P}_{t+1}|)$ elements of $\mathcal{F}_i\}$
17: $\quad\quad$ **end if**
18: $\quad\quad i \leftarrow i + 1$
19: $\quad$ **end while**
20: $\quad$ Calculate the crowding distance for each $\boldsymbol{x} \in \boldsymbol{P}_{t+1}$.
21: $\quad$ Create $\boldsymbol{Q}_{t+1}$ of size N using crowded comparison selection, crossover and mutation.
22: $\quad t \leftarrow t + 1$
23: **end while**
24: $\boldsymbol{A}^* = \boldsymbol{P}_{G_{\max}}$

each objective. The complexity of this process is dominated by the sorting procedure which is $\mathrm{O}(MN \log N)$. A pseudocode listing for the crowding distance calculation appears in Algorithm 13. The crowding distances may now be used to estimate solution density, with a higher value indicating a more isolated solution [61].

Pareto-rank and crowding distance measures give rise to the so-called *crowded comparison operator* ($\succ_{\mathrm{c}}$) for use in binary tournament selection. This operator defines the fitter of two solutions ($\boldsymbol{i} \succ_{\mathrm{c}} \boldsymbol{j}$) firstly as the one with the lower rank ($\boldsymbol{i}_{\mathrm{rank}} < \boldsymbol{j}_{\mathrm{rank}}$) or, provided the ranks are equal, the one with the largest crowding distance (if $\boldsymbol{i}_{\mathrm{rank}} = \boldsymbol{j}_{\mathrm{rank}}$, then $\boldsymbol{i}_{\mathrm{dist}} > \boldsymbol{j}_{\mathrm{dist}}$). This favours exploration of solutions in less crowded regions [61].

A pseudocode listing for the main NSGA-II loop appears in Algorithm 11. The initial population $\boldsymbol{P}_0$ is generated randomly. In order to produce the first offspring population $\boldsymbol{Q}_0$, $\boldsymbol{P}_0$ is sorted using the FNSA, and crowding distance values are calculated for each solution. Offspring creation proceeds by binary tournament selection from $\boldsymbol{P}_0$ using the crowded comparison operator, solution crossover, and mutation. For binary-coded chromosomes Deb *et al.* [61] used single-point crossover and bitwise mutation with a probability $1/\kappa$ (where κ is the number of

Algorithm 12 Fast Non-dominated Sorting Algorithm [61]

Input: A population of solutions $\boldsymbol{P}$, where each solution is a specific assignment of values to decision variables $\boldsymbol{x}$, a vector $\boldsymbol{y}$ of two or more computed objective function values for each solution.

Output: The original population partitioned into successively dominating fronts, $\mathcal{F}_1, \ldots, \mathcal{F}_n$.

```
1: F_1 ← ∅
2: for all p ∈ P do
3:    S_p ← ∅                                  [S_p is set of solutions which p dominates]
4:    d_p^c ← 0                                [d_p^c is count of how many solutions dominate p]
5:    for all q ∈ P do
6:       if p ≺ q then
7:          S_p ← S_p ∪ {q}                   [p dominates q]
8:       else if q ≺ p then
9:          d_p^c = d_p^c + 1                 [increment the domination counter of p]
10:      end if
11:   end for
12:   if d_p^c = 0 then
13:      p_rank ← 1                            [p belongs to the first (non-dominated) front]
14:      F_1 ← F_1 ∪ p
15:   end if
16: end for
17: i ← 1
18: while F_i ≠ ∅ do
19:   Q ← ∅
20:   for all p ∈ F_i do
21:      for all q ∈ S_p do
22:         d_q^c ← d_q^c − 1                  [discount effect of i-th front on q's domination count]
23:         if d_q^c = 0 then
24:            q_rank ← i + 1                  [q is a member of the next front]
25:            Q ← Q ∪ {q}
26:         end if
27:      end for
28:   end for
29:   i ← i + 1
30:   F_i ← Q
31: end while
```

decision variables), and for real-coded chromosomes, they used SBX crossover with polynomial mutation. Once the first offspring population has been created, t is set equal to 1, and the following steps are iterated for a number of generations until $t = G_{\max}$:

1. The parent and offspring populations are combined ($\boldsymbol{R}_t \leftarrow \boldsymbol{P}_t \cup \boldsymbol{Q}_t$).
2. $\boldsymbol{R}_t$ is ranked using FNSA, and sorted into non-dominated fronts $\mathcal{F}_1, \ldots, \mathcal{F}_k$ and crowding distances are calculated for each solution.
3. The next population $\boldsymbol{P}_{t+1}$ is created by including all solutions in the first front $\mathcal{F}_1$, the second front, and so forth, until including the next front's solutions will increase the population size beyond N. This front is then sorted in order of decreasing crowding distance and solutions are added until $|\boldsymbol{P}_{t+1}| = N$.

Algorithm 13 Crowding Distance Assignment Algorithm [61]

Input: A population of solutions $\boldsymbol{P}$, where each solution is a specific assignment of values to decision variables $\boldsymbol{x}$, a vector $\boldsymbol{y}$ of two or more computed objective function values for each solution.

Output: The crowding distance of population member, $\boldsymbol{P}[1]_{\text{dist}}, \ldots, \boldsymbol{P}[n]_{\text{dist}}$.

1: $h =\mid \boldsymbol{P} \mid$ [h is the number of solutions in $\boldsymbol{P}$]
2: **for all** $i \in \boldsymbol{P}$ **do**
3: $\quad \boldsymbol{P}[i]_{\text{dist}} \leftarrow 0$ [initialize crowding distance]
4: **end for**
5: **for all** M objectives **do**
6: $\quad \boldsymbol{P} = \text{sort}(\boldsymbol{P}, k)$ [sort population using value of objective k]
7: $\quad \boldsymbol{P}[1]|k \leftarrow \infty$ [so that boundary points are always selected]
8: $\quad \boldsymbol{P}[h]|k \leftarrow \infty$
9: $\quad$ **for** $i = 2\text{to}(h-1)$ **do**
10: $\quad\quad \boldsymbol{P}[i]_{\text{dist}}|k \leftarrow \boldsymbol{P}[i]_{\text{dist}}|k + (\boldsymbol{P}[i+1]|k - \boldsymbol{P}[i-1]|k)/(k_{\max} - k_{\min})$
11: $\quad$ **end for**
12: **end for**

4. Offspring population $\boldsymbol{Q}_{t+1}$ is created using $\succ_{\text{c}}$ binary tournament selection, crossover, and mutation.

5. The value of t is incremented and the process is repeated from Step 1.

In order to accommodate constrained optimisation, Deb *et al.* [61] developed their own notion of *constrained-domination*. However, their definition is a watered down version of the one provided in §5.4.4. A solution $\boldsymbol{i}$ is said to dominate a solution $\boldsymbol{j}$ if:

1. Both solutions are feasible, and $\boldsymbol{i}$ dominates $\boldsymbol{j}$ in terms of the standard crowded comparison operator, or

2. Solution $\boldsymbol{i}$ is feasible and solution $\boldsymbol{j}$ is infeasible, or

3. Both solutions are infeasible, and solution $\boldsymbol{i}$ has the lowest overall constraint violation.

The notion of 'overall lowest constraint violation' requires that constraint violations be normalized and summed (although weighting coefficients may also be used). This method was employed in this dissertation.

5.6.2 SPEA-II

SPEA-II was developed by Zitzler *et al.* [277] in order to improve upon its predecessor and take advantage of new MOEA techniques. It has proven very competitive versus the NSGA-II algorithm, particularly in terms of solution diversity.

In a 2005 study on MOEAs for WDS design, Farmani *et al.* [88] concluded that NSGA-II is outperformed by SPEA-II, especially in terms of the evenness of solution spread. However, this comes at the cost of increased computational time.

SPEA-II employs a population $\boldsymbol{P}$ of size N and a fixed-size archive $\overline{\boldsymbol{P}}$ of size $\overline{N}$ to store non-dominated solutions. If there are not enough non-dominated solutions available to fill the archive

to the required size, then it is filled by dominated solutions. Diversity preservation is achieved by the fitness assignment method, along with the manner in which dominated solutions are selected, as well as a *truncation method* which is invoked when there are more non-dominated solutions than the archive size. This operator has been designed to prevent clustering and preserve boundary solutions [277].

SPEA-II uses a finely grained fitness assignment strategy in order to improve diversity. Fitness assignment begins by calculating the dominance strength of each solution $\boldsymbol{i} \in \boldsymbol{P} \cup \overline{\boldsymbol{P}}$ as $\hat{S}(\boldsymbol{i}) = \left|(\boldsymbol{j} \in \boldsymbol{P} \cup \overline{\boldsymbol{P}}) \wedge (\boldsymbol{i} \prec \boldsymbol{j})\right|$, where $\prec$ is the Pareto-dominance relation. Let $\mathcal{S}_{\boldsymbol{i}}$ be the set of solutions which dominates $\boldsymbol{i}$. The raw fitness of $\boldsymbol{i}$ is then calculated as $\hat{R}(\boldsymbol{i}) = \sum_{\boldsymbol{j} \in \mathcal{S}_{\boldsymbol{i}}} \hat{S}(\boldsymbol{j})$. Note that raw fitness is to be minimized, and all non-dominated solutions have a raw fitness of zero. Density information is incorporated into the fitness quantification in order to discriminate between solutions of identical raw fitness. The Euclidean distance to the k-th nearest neighbour ($\sigma_{\boldsymbol{i}}^k$) is calculated by finding the distances in objective function space to every other solution and sorting this list in increasing order. The k-th entry in the sorted list is the sought-after distance $\sigma_{\boldsymbol{i}}^k$. The value of k is commonly selected as the square root of the sample size, $\sqrt{N + \overline{N}}$. The adapted inverse $\hat{I}(\boldsymbol{i}) = \frac{1}{\sigma_{\boldsymbol{i}}^k + 2}$ is added to the raw fitness to achieve a total fitness $F(\boldsymbol{i}) = \hat{R}(\boldsymbol{i}) + \hat{I}(\boldsymbol{i})$. Note that $0 \leq \hat{I}(\boldsymbol{i}) \leq 0.5$ [277].

The archive selection process works as follows: First all the non-dominated solutions in $\boldsymbol{M}_t = \boldsymbol{P}_t \cup \overline{\boldsymbol{P}_t}$ are copied into the new archive $\boldsymbol{P}_{t+1}$. If $\left|\overline{\boldsymbol{P}}_{t+1}\right| = \overline{N}$, then this step is complete; otherwise there are two distinct cases. Firstly, if $\left|\overline{\boldsymbol{P}}_{t+1}\right| < \overline{N}$, then the remaining solutions in $\boldsymbol{M}_t$ are sorted in order of increasing fitness value, and the first $\overline{N} - \left|\overline{\boldsymbol{P}}_{t+1}\right|$ solutions for which $F(i) \geq 1$ are inserted into the archive. Alternatively, if $\left|\overline{\boldsymbol{P}}_{t+1}\right| > \overline{N}$, then a truncation operator is iteratively invoked to eliminate $\left|\overline{\boldsymbol{P}}_{t+1}\right| - \overline{N}$ solutions from the archive. In each iteration, an individual $\boldsymbol{i}$ is selected for removal such that $\boldsymbol{i} \leq_{\mathrm{d}} \boldsymbol{j}$ for all $\boldsymbol{j} \in \boldsymbol{P}_{t+1}$, where

$$\begin{aligned} \boldsymbol{i} \leq_{\mathrm{d}} \boldsymbol{j} \quad \Leftrightarrow \quad & \forall\, 0 < k < |\boldsymbol{P}_{t+1}| : \sigma_{\boldsymbol{i}}^k = \sigma_{\boldsymbol{j}}^k \quad \vee \\ & \exists\, 0 < k < |\boldsymbol{P}_{t+1}| : (\forall\, 0 < \ell < k : \sigma^{\ell}{}_{\boldsymbol{i}} = \sigma_{\boldsymbol{j}}^{\ell}) \wedge \sigma_{\boldsymbol{i}}^k < \sigma_{\boldsymbol{j}}^k. \end{aligned}$$

This operator basically means that the solution with the smallest distance to another solution is chosen for removal, and where these distances are tied, their second smallest distances are compared, and so forth. This guarantees preservation of boundary solutions, since the solution closest to a boundary solution will always have some k-th distance to another solution which is less than that of the boundary solution's k-th distance.

Only archive members participate in the mating process, and any suitable operators may be substituted for the recombination and mutation steps. A pseudocode listing of the SPEA-II appears in Algorithm 14.

Although SPEA-II has a similar computational complexity to NSGA-II (O(MN^2) in every generation)), it generally performs slower than NSGA-II in comparative studies [88]. This is due to the requirement of sorting solution distance vectors for each solution. However, SPEA-II generally achieves better solution distribution. Advanced data-structures and algorithms exist for reducing the computational complexity of both SPEA-II and NSGA-II [138], however, this author was only made aware of them towards the end of his research.

5.6.3 Differential Evolution

Differential evolution (DE) was first proposed by Storn and Price [221] in 1997, as a generic metaheuristic for the optimisation of nonlinear and non-differentiable continuous space func-

Algorithm 14 Strength Pareto Algorithm II (SPEA-II) [277]

Input: A MOO problem where solutions are an assignment of values to decision variables $\boldsymbol{x}$, a population size N, an archive size $\overline{N}$, a set of constraints and violation magnitude functions, M objective functions to produce entries for the $M \times 1$ objective vector $\boldsymbol{y}$, a maximum number of generations $G_{\max}$.

Output: An approximation of the Pareto-optimal solution set in multi-objective space, $\boldsymbol{A}^*$.

1: Randomly generate an initial population of solutions $\boldsymbol{P}_0$ of size N and create an empty archive $\overline{\boldsymbol{P}}_0$.
2: $t \leftarrow 0$
3: Calculate the strength value $\hat{S}(\boldsymbol{i})$ for each solution $\boldsymbol{i} \in \boldsymbol{P}_t$ as $\hat{S}(\boldsymbol{i}) = \left|\boldsymbol{j} \in \boldsymbol{P} \cup \overline{\boldsymbol{P}} \wedge \boldsymbol{i} \prec \boldsymbol{j}\right|$.
4: For every $\boldsymbol{i} \in \boldsymbol{P}_t$, determine $\mathcal{S}_{\boldsymbol{i}}$, the set of solutions which dominates i. The raw fitness of $\boldsymbol{i}$ is then calculated as $\hat{R}(\boldsymbol{i}) = \sum_{\boldsymbol{j} \in \mathcal{S}_{\boldsymbol{i}}} \hat{S}(\boldsymbol{j})$.
5: For every $\boldsymbol{i} \in \boldsymbol{P}_t$, calculate the Euclidean distances $\sigma_{\boldsymbol{i},\boldsymbol{j}}$ in objective function space to every other solution, $\boldsymbol{j} = 1 \ldots N$. Sort these distances in increasing order and identify $\sigma_{\boldsymbol{i}}^k$ as the k-th distance value. Set $\hat{I}(\boldsymbol{i}) = \frac{1}{\sigma_{\boldsymbol{i}}^k + 2}$.
6: Set the fitness of every $\boldsymbol{i}$ as $F(\boldsymbol{i}) = \hat{R}(\boldsymbol{i}) + \hat{I}(\boldsymbol{i})$.
7: Copy all the non-dominated solutions ($F(\boldsymbol{i}) < 1$) in $\boldsymbol{M}_t = \boldsymbol{P}_t \cup \overline{\boldsymbol{P}}_t$ to archive $\overline{\boldsymbol{P}}_{t+1}$.
8: **if** $\mid \overline{P}_{t+1} \mid > \overline{N}$ **then**
9: Iteratively employ the truncation operator ($\leq_{\mathrm{d}}$) to remove solutions until $\mid \overline{\boldsymbol{P}}_{t+1} \mid = \overline{N}$.
10: **else if** $\mid \overline{\boldsymbol{P}}_{t+1} \mid < \overline{N}$ **then**
11: Sort $\boldsymbol{M}_t$ in order of increasing fitness values and fill $\overline{\boldsymbol{P}}_{t+1}$ with the first $\overline{N} - \mid \overline{\boldsymbol{P}}_{t+1} \mid$ solutions having $F(i) \geq 1$.
12: **end if**
13: **if** $t \geq G_{\max}$ **then** $\boldsymbol{A}^* = \overline{\boldsymbol{P}}_{t+1}$ and Terminate.
14: Perform binary tournament selection with replacement on $\overline{\boldsymbol{P}}_{t+1}$ in order to fill the mating pool.
15: Apply recombination and mutation operators to the mating pool and set $\boldsymbol{P}_{t+1}$ to the resulting population.
16: Set $t \leftarrow t + 1$ and return to Step 3.

tions, and has proven very robust and competitive with respect to other evolutionary algorithms. At the heart of its success lies a very simple differential operator, whereby a trial solution vector is generated by mutating a random target vector by some multiple of the difference vector between two other random population members. For three distinct random indices i, j and k, this has the form

$$\boldsymbol{y}_i = \boldsymbol{x}_i + \hat{f} \times (\boldsymbol{x}_j - \boldsymbol{x}_k),$$

where $\boldsymbol{x}_i$ is the target vector, $\boldsymbol{y}_i$ is the trial vector and $\hat{f}$ is a constant factor in the range $[0, 2]$ which controls the amplification of differential variation, typically taken as 0.5. If the trial vector has a better objective function value, then it replaces its parent vector. Storn and Price also included a crossover operator between the trial vector and the target vector in order to improve convergence.

The original DE method was formulated for single-objective optimisation only. Several adaptations of DE have been proposed in order to extend it for multi-objective optimisation. These have included the *Pareto-based DE approach* [28], the Pareto DE Approach (PDEA) [167], DE for Multi-objective Optimisation (DEMO) [202], Generalized DE (GDE) in three different versions [151, 152, 156].

Several variants of multi-objective DE were compared. The version used in this dissertation is a slight variation on the first GDE, where trial vectors are generated and saved to an offspring population only if they are non-dominated with respect to their target vectors. Once the offspring population is the same size as the parent population, the two populations are combined and the NSGA-II environmental selection mechanism is used to produce the next generation. A pseudocode listing of GDE appears in Algorithm 15.

Multi-objective DE has been used successfully in the context of water resources management for the operational policy design of a large reservoir system by Reddy and Kumar [201] in 2007, with objectives of minimizing flood risk, maximizing hydropower generation, and minimizing irrigation deficits.

Algorithm 15 Generalized Differential Evolution Algorithm [156]

Input: A MOO problem where solutions are an assignment of values to decision variables $\boldsymbol{x}$, a population size N, a set of constraints and violation magnitude functions, M objective functions to produce entries for the $M \times 1$ objective vector $\boldsymbol{y}$, a maximum number of generations $G_{\max}$, a difference factor $\hat{f} \in [0.4, 2]$, and a probability of crossover $p_c \in [0, 1]$.

Output: An approximation of the Pareto-optimal solution set in multi-objective space, $\boldsymbol{A}^*$.

Randomly generate an initial population of solutions $\boldsymbol{P}_0$ of size N.
$t \leftarrow 0$
while $t < G_{\max}$ **do**
  $i \leftarrow 1$
  while $i \leq N$ **do**
    Generate three distinct random indices $i, j, k \in [1, N]$ in order to select solutions $\boldsymbol{x}_i, \boldsymbol{x}_j, \boldsymbol{x}_k$ from $\boldsymbol{P}_t$.
    Calculate $\boldsymbol{y}_i = \boldsymbol{x}_i + \hat{f} \times (\boldsymbol{x}_j - \boldsymbol{x}_k)$.
    if $\boldsymbol{y}_i \preceq \boldsymbol{x}_i$ **then**
      $\boldsymbol{x}_i \leftarrow \boldsymbol{y}_i$
    end if
  end while
  Set $t \leftarrow t + 1$
end while
$\boldsymbol{A}^* \leftarrow \boldsymbol{P}_t$.

5.7 Alternative Multi-objective Algorithms

In this section several metaheuristics for multi-objective WDS design optimisation are discussed as a contrast to traditional MOEAs. These include a novel Greedy Algorithm that mimics decision strategies typically employed by a human engineer, Multi-objective Particle Swarm Optimisation, two estimation of distribution algorithms, namely the Univariate Marginal Distribution Algorithm (UMDA) and the novel Partitioned UMDA, and finally two dynamic adaptive MOEAs, called Another Dynamic MOEA and ANIMA.

5.7.1 A Multi-objective Greedy Algorithm

A greedy WDS design heuristic was developed by the author for specific use in this dissertation, named the *WDS Greedy Algorithm* (GREEDY). It is adapted from four prior WDS design

heuristics, namely those of Keedwell and Khu [142], and Afshar *et al.* [4], which were designed for single objective least-cost optimisation, that of Todini [227], which employs a goal-programming approach using *Resilience Index* to generate a trade-off curve, and a similar approach using the concept of *Pipe Unitary Power*, which appears in the work by Saldarriga *et al.* [205]. In addition to these heuristics, it also employs several practical adjustment steps to improve performance based on engineering judgement. The new combined algorithm is greedy in the sense that it conducts a neighborhood search in which the best improvement step is followed for each of the different heuristic rules. Owing to its greedy nature, there is a danger that the search may become trapped at local optima. However, the practice of incorporating different search mechanisms reduces this probability. The advantage of using this algorithm is that it incorporates design strategies similar to those that might be used by an engineer during the course of a manual design procedure. It may be considered a local search component within the framework of a broader evolutionary search.

In 2000 Todini [227] presented a goal-programming design heuristic for rapidly approximating the Pareto-optimal curve in cost/resilience space (see the Resilience Index defined in §4.3.2). For a given solution, if no pressure deficit occurs, a reduction of diameters is performed with respect to the pipe p_i^* for which the largest decrease in cost per unit of power dissipation occurs during a single step reduction in pipe diameters ($j^\lambda \rightarrow j^{\lambda-1}$). That is,

$$p_i^* = \max_{i=1,\ldots,p} \left\{ -\frac{C_i^{\lambda-1} - C_i^{\lambda}}{P_{\mathrm{r},i}^{\lambda-1} - P_{\mathrm{r},i}^{\lambda}} \right\},$$

where C_i^λ and $P_{\mathrm{r},i}^\lambda$ are the cost and power respectively of pipe i at the current diameter $x_i = j^\lambda$. Todini applies three tests before a diameter reduction may occur (hydraulic simulation is not performed to calculate the pressure and flow conditions at the lower diameter). Firstly, velocity constraints may not be exceeded at the lower pipe diameter (using an upper bound of 2 m/s). Velocity may be approximated (using the current value of flow Q) as $\bar{v} = Q/A = Q/(\pi D^2/4)$. Secondly, the Resilience Index, which depends on flow and head loss in the pipe, may not fall below a currently specified target. Finally, Todini makes use of a failure index which equates to stating that pressure deficit may not occur at any of the nodes. This cost-efficient reduction of pipe diameters continues iteratively until one of these three tests fails. Given a solution where pressure deficit does indeed occur, the increase of pipe diameters proceeds according to the largest decrease of internal power dissipation per unit cost,

$$q_i^* = \max_{i=1,\ldots,p} \left\{ -\frac{P_{\mathrm{r},i}^{\lambda} - P_{\mathrm{r},i}^{\lambda-1}}{C_i^{\lambda} - C_i^{\lambda-1}} \right\},$$

as a function of the increase in diameter x_i from $j^{\lambda-1}$ to j^λ. These diameter increase operations continue iteratively until the Resilience Index is above the specified target value. Once a new configuration has been determined, hydraulic simulation is performed, and the above process is repeated. Todini's method must be applied for a range of target resilience values in order to attain a Pareto-optimal approximation.

A similar procedure may be followed for the Unitary Power metric [205], which is defined as pipe discharge q_i, multiplied by the difference between the pressure head at the pipes initial ($h_{i,\mathrm{init}}$) and final ($h_{i,\mathrm{fin}}$) nodes, such that $h_{i,\mathrm{init}} - h_{i,\mathrm{fin}} > 0$. The pipe unitary power is therefore $P_{\mathrm{ru},i} = q_i(h_{i,\mathrm{init}} - h_{i,\mathrm{fin}})$. The calculation of this property is less computationally intensive than that of Resilience Index. In this method, in each iteration the diameter of the pipe with the highest $P_{\mathrm{ru},i}$ may be increased to the next diameter size, and similarly, the pipe with the smallest unitary power may be decreased to the next smallest commercial diameter.

The heuristic of Afshar *et al.* [4], proposed in 2005, appears as a sub-component of a larger algorithm and is used in the context of converting a continuous diameter solution to discrete diameters. Although they only consider the problem of converting an infeasible solution to a feasible one, this may easily be adapted in the reverse direction when a solution is over-specified. For a given solution, a node with the maximum head deficit is identified. All paths conveying water from any source to this node are established. This may easily be done by considering the flow direction supplied by the hydraulic simulator. It is obvious that increasing the diameter of any pipe on these paths should increase the head at the node in question. The pipe whose diameter is selected for increase is the one that results in the maximum decrease in a total penalized system cost (for which they employed a function similar to, but less complex than (5.2)). Once a pipe diameter has been increased, the system is hydraulically simulated, and the process is repeated until all nodal head constraints are satisfied. Given a solution with no head deficit, but with at least one maximum velocity constraint violation, a search is conducted to find a pipe with the largest maximum velocity violation. Then all the pipes which take flow away from the first node of this pipe are established. The pipe amongst these whose diameter is selected for increase is again one that results in the largest decrease in total penalized cost.

In 2006 Keedwell and Khu [142] developed a cellular automata approach towards initializing WDS optimisation searches with healthy designs instead of using random initial configurations. The method is named the Cellular Automaton Network Design Algorithm (CANDA). It considers each demand node as a cell in an automaton and iteratively evaluates the head deficit or excess of that node (based on some target pressure). If a node experiences a pressure deficit, all the pipes supplying water to that node are increased to the next largest size. Similarly, if a node experiences a head excess, the incoming pipes are downsized. These changes are implemented for every node in the network before the next hydraulic simulation is conducted. It was shown that this method converges rapidly to semi-realistic configurations, but was of limited use in further refining designs [142]. This heuristic performs macroscopic changes to a configuration and was selected to assist in generating replacement solutions for duplicates in the population.

Finally, the additional heuristic steps implemented by the author are: incrementing the diameter of the pipe which has the largest head loss, decrementing the diameter of the pipe which has the smallest head loss, incrementing the diameter of the pipe which has the largest head loss per unit length (or *head loss gradient*), decrementing the diameter of the pipe which has the smallest head loss gradient, and similar steps for pipe unitary power. In order to apply these steps in a multi-objective setting, the search is conducted in both directions (both increasing and reducing reliability) for every solution considered. A pseudocode listing of the combined greedy algorithm appears as Algorithm 16. The heuristic techniques are referred to as a *Cost-Power Benefit* step (for which a pseudocode listing appears in Algorithm 17), an *Efficient-Path* step (for which a pseudocode listing appears in Algorithm 18), and the *CANDA* replacement method (for which a pseudocode listing appears in Algorithm 19), inspired respectively by the design heuristics of Todini, Afshar *et al.* and Keedwell and Khu, described above. Note that the offspring are only accepted as valid population members provided they are not dominated by their parent solutions.

5.7.2 Multi-objective Particle Swarm Optimisation

Although highly elaborate versions of the PSO algorithm exist for multi-objective optimisation (MOPSO — see, for example, [36]), a more basic version was selected for inclusion in this dissertation. This version is almost identical to the version discussed in §3.7.9, except that PSO fitness is calculated as the *crowding distance of a solution divided by the square root*

Algorithm 16 Greedy WDS Design Heuristic

Input: A population of solutions $\boldsymbol{P}$, where each solution is a specific assignment of values to decision variables $\boldsymbol{x}$, a vector $\boldsymbol{y}$ of and two or more computed objective function values for each solution, computed hydraulic parameters (pressure, flow, velocity) for every solution, and a number of offspring $N^{[1]}$ which must be generated.

Output: An offspring population $\boldsymbol{O}$ of size $N^{[1]}$.

1: $\boldsymbol{O} \leftarrow \emptyset$
2: **while** $|\boldsymbol{O}| < N^{[1]}$ **do**
3: Select a parent solution $\boldsymbol{x}$ from $\boldsymbol{P}$ by means of binary tournament selection.
4: Decrease the diameter of the pipe in $\boldsymbol{x}$ with the smallest head loss to yield $\boldsymbol{x}'$
5: **if not** $\boldsymbol{x} \prec \boldsymbol{x}'$ **then** $\boldsymbol{O} \leftarrow \boldsymbol{O} \cup \boldsymbol{x}'$
6: Increase the diameter of the pipe in $\boldsymbol{x}$ with the largest head loss to yield $\boldsymbol{x}'$
7: **if not** $\boldsymbol{x} \prec \boldsymbol{x}'$ **then** $\boldsymbol{O} \leftarrow \boldsymbol{O} \cup \boldsymbol{x}'$
8: Decrease the diameter of the pipe in $\boldsymbol{x}$ with the smallest unitary power to yield $\boldsymbol{x}'$
9: **if not** $\boldsymbol{x} \prec \boldsymbol{x}'$ **then** $\boldsymbol{O} \leftarrow \boldsymbol{O} \cup \boldsymbol{x}'$
10: Increase the diameter of the pipe in $\boldsymbol{x}$ with the largest unitary power to yield $\boldsymbol{x}'$
11: **if not** $\boldsymbol{x} \prec \boldsymbol{x}'$ **then** $\boldsymbol{O} \leftarrow \boldsymbol{O} \cup \boldsymbol{x}'$
12: Execute a *Cost-Power Benefit* step [Algorithm 17] on $\boldsymbol{x}$ to yield solutions $\{\boldsymbol{x}'\}$
13: **for all** $\boldsymbol{z} \in \{\boldsymbol{x}'\}$ **if not** $\boldsymbol{x} \prec \{\boldsymbol{z}\}$ **then** $\boldsymbol{O} \leftarrow \boldsymbol{O} \cup \boldsymbol{z}$
14: Execute an *Efficient-Path* step [Algorithm 18] on $\boldsymbol{x}$ to yield solutions $\{\boldsymbol{x}'\}$
15: **for all** $\boldsymbol{z} \in \{\boldsymbol{x}'\}$ **if not** $\boldsymbol{x} \prec \{\boldsymbol{z}\}$ **then** $\boldsymbol{O} \leftarrow \boldsymbol{O} \cup \boldsymbol{z}$
16: **end while**
17: Sort the solutions in $\boldsymbol{O}$ in terms of increasing cost.
18: Any consecutive solutions which are identical are replaced in $\boldsymbol{O}$ by a random initial solution which has been modified using the *CANDA* algorithm [19].

of its Pareto-rank. No global best position is used, *only a local best*, which is identified for each dominated individual as *the Pareto-solution which yields the highest PSO fitness value normalized by the Euclidean distance between the solutions in objective space*. Particle collisions are accommodated by randomly generating a new solution with a random initial velocity. In this dissertation, MOPSO is implemented using an inertial weight of $w = 0.75$ and learning factors $c_1 = c_2 = 2$. Finally, the algorithm was adapted to round continuous positions to discrete component values.

5.7.3 Univariate Marginal Distribution Algorithm

The *Univariate Marginal Distribution* algorithm (UMD / UMDA) is possibly the simplest EDA, assuming no interaction between variables. In each generation it builds anew separate probability distributions for each gene using allele frequencies in the population. These univariate probability density functions are then stochastically sampled in order to generate new gene values for offspring creation. The probability of generating a particular individual is the product of the individual's allele probabilities. This simple technique is surprisingly effective when combined with a Pareto-based selection scheme and an anti-crowding mechanism, such as that of the NSGA-II which is employed in this dissertation.

In 2009 Olsson *et al.* [185] compared three EDAs for MO WDSDO, including the *Hierarchical Bayesian Optimisation* (HBO) algorithm, developed by Pelikan and Goldberg [191] in 2002, the *Chi-square matrix method* (CSM) for building block identification, developed by Aporntewan

Algorithm 17 Cost-Power Benefit Step

Input: A single solution $\boldsymbol{x}$ which is a specific assignment of values to decision variables $\boldsymbol{x}$, its computed hydraulic parameters (pressure, flow, velocity), and a target pressure head h.

Output: Offspring solutions $\boldsymbol{x}'_1$ and $\boldsymbol{x}'_2$.

1: Execute a decreasing diameter step as follows:
2: Calculate the head loss $h_{L(i)}$ for every pipe p_i of $\boldsymbol{x}$ at the diameter $j^{\lambda-1}$ which is one size lower than its current diameter (use Hazen-Williams or Darcy-Weisbach head loss equation with fixed flow q)
3: Calculate the power $P^{\lambda}_{\mathrm{r},i}$ and $P^{\lambda-1}_{\mathrm{r},i}$ at the exit node of each pipe for the current and lower diameters respectively ($P_{\mathrm{r},i} = \gamma q_i h_i$)
4: Calculate $w_i = -\frac{C_i^{\lambda-1} - C_i^{\lambda}}{P^{\lambda-1}_{\mathrm{r},i} - P^{\lambda}_{\mathrm{r},i}}$ for every pipe, where $C_i^{\lambda-1} - C_i^{\lambda}$ is the difference in pipe cost between the higher and lower diameters.
5: Identify the pipe p_i with the maximum value of w_i. Set $\boldsymbol{x}' = \boldsymbol{x}$ and decrement the diameter of p_i in $\boldsymbol{x}'$
6: For solution $\boldsymbol{x}'$, calculate the new pipe velocity $v_i = q_i/\pi(d_i^2/4)$ for a fixed flow q_i.
7: **if** $v_i < v_{max}$ **then**
8: Output solution $\boldsymbol{x}'_1 = \boldsymbol{x}'$
9: **end if**
10: Execute an increasing diameter step as follows:
11: Calculate the head loss $h_{L(i)}$ for every pipe p_i of $\boldsymbol{x}$ at the diameter j^{λ} which is one size larger than its current diameter
12: Calculate the power $P^{\lambda}_{\mathrm{r},i}$ and $P^{\lambda-1}_{\mathrm{r},i}$ at the exit node of each pipe for the larger and current diameters respectively
13: Calculate $w_i = -\frac{P^{\lambda}_{\mathrm{r},i} - P^{\lambda-1}_{\mathrm{r},i}}{C_i^{\lambda} - C_i^{\lambda-1}}$ for every pipe
14: Identify the pipe p_i with the maximum value of w_i. Set $\boldsymbol{x}' = \boldsymbol{x}$ and increment the diameter of p_i in $\boldsymbol{x}'$
15: For solution $\boldsymbol{x}'$, calculate the new pipe velocity $v_i = q_i/\pi(d_i^2/4)$ for a fixed flow q_i.
16: **if** $v_i < v_{max}$ **then**
17: Output solution $\boldsymbol{x}'_2 = \boldsymbol{x}'$
18: **end if**

and Chongstitvatana [12] in 2004, and the UMDA proposed by Mhlenbein [177] in 1997. As a benchmark optimizer, they also included NSGA-II in the analysis. The first two algorithms both use some mechanism for the consideration of inter-variable dependencies. EDAs typically use population sizes an order of magnitude larger than traditional MOEAs. However, they require fewer generations to converge to stable probability distributions.

The findings of Olsson *et al.* were as follows. Whilst the HBO algorithm was highly effective for designing small WDS systems, the algorithm's performance deteriorated completely for large, real-world systems, likely due to exponential growth in multiplicity of possible variable interactions. CSM maintained good performance for the larger systems, demonstrating a better solution spread than UMDA. However, UMDA was clearly the best algorithm overall in terms of Pareto-dominance. Both latter algorithms also significantly outperformed NSGA-II, but they exhibited largely reduced diversity. In view of the superiority of UMDA, it was selected for inclusion in this dissertation. A pseudocode listing of the UMDA algorithm appears in Algorithm 20.

A variant of the UMDA, called the *Partitioned UMDA* (PUMDA), was developed by the author,

Algorithm 18 Efficient-Path Step

Input: A single solution $\boldsymbol{x}$, its computed hydraulic parameters (pressure, flow, velocity), and a target pressure head h.

Output: Offspring solutions $\boldsymbol{x}'_1$ and $\boldsymbol{x}'_2$.

if for any pipe in $\boldsymbol{x}$, $v_i > v_{\max}$ **then**
  Identify pipe p_v in $\boldsymbol{x}$ with largest maximum velocity constraint violation and node n_v which it accepts flow from
  Identify set of pipes $\mathcal{V}$ which take flow away from node n_v
  For every system with $p_v \in \mathcal{V}$ at a higher diameter, calculate the penalized cost (5.2)
  Select the pipe amongst these for which a diameter increment results in the largest decrease, and let $\boldsymbol{x}'_1$ be the resulting system
  Output solution $\boldsymbol{x}'_1$
else
  Execute a decreasing diameter step as follows:
  Identify node n_i in $\boldsymbol{x}$ with largest pressure excess
  Identify $\mathcal{C}$, the set of pipes constituting paths from all sources to n_i
  Approximate v_c, and $h_{L(c)}$ for every $c \in \mathcal{C}$ whose diameter has been decremented one size
  For every system defined by c at a lower diameter, calculate the penalized cost (5.2)
  Select the pipe for diameter decrement which yields the smallest increase in total penalized cost, and let $\boldsymbol{x}'_1$ be the resulting system
  Output solution $\boldsymbol{x}'_1$
end if
Execute an increasing diameter step as follows:
Identify node n_i in $\boldsymbol{x}$ with largest pressure deficit
Identify $\mathcal{C}$, set of pipes constituting the paths from all sources to n_i
Approximate v_c, and $h_{L(c)}$ for every $c \in \mathcal{C}$ whose diameter has been incremented one size
For every system defined by pipe c at a higher diameter, calculate the penalized cost (5.2)
Select pipe for diameter increment which yields the largest decrease in total penalized cost, and let $\boldsymbol{x}'_2$ be the resulting system
Output solution $\boldsymbol{x}'_2$

Algorithm 19 CANDA Replacement Method

Input: N/A

Output: A CANDA improved offspring solution $\boldsymbol{x}$.

Generate a random initial solution $\boldsymbol{x}$, and simulate its hydraulics.
Generate a uniform random variable $u \in [5, 10]$; $i \leftarrow 0$
while $i < u$ **do**
  for all nodes $n_i \in \boldsymbol{x}$ **do**
    if $h_i > h_{i,\max}$ **then**
      Decrement diameters of all incident pipes with incoming flow to node n_i
    else if $h_i < h_{i,\min}$ **then**
      Increment diameters of all incident pipes with incoming flow to node n_i
    end if
  end for
  $i \leftarrow i + 1$
end while
return $\boldsymbol{x}$

whereby in each generation the objective space is partitioned along the reliability axis. The size of each partition is generated independently using half of the absolute value of a normal distribution sample with a mean of zero and a standard deviation equal to one third of the reliability range ($r_{\min}$,$r_{\max}$), sampled iteratively until the full range is partitioned. For each sub-range of the reliability range, all the solutions that fall into that partition are then employed to generate a UMD probability model for that partition. In order to generate new solutions, a partition is selected at random and its UMD model sampled to produce offspring. The philosophy behind this method is that different levels of reliability correspond to different design paradigms (described by critical solution schemata), which can only be exploited fully by honing in on these regions of the objective space. By allowing variable-sized partitions and redefining them during each generation, PUMDA allows for paradigm overlap and mixing. PUMDA was found to outperform the UMDA for many of the test cases.

Algorithm 20 UDM Algorithm [177, 185]

Input: A MOO problem with solutions $\boldsymbol{x}$, a population size N, a set of constraints and violation magnitude functions, M objective functions, a maximum number of generations $G_{\max}$.

Output: An approximation of the Pareto-optimal solution set in multi-objective space, $\boldsymbol{A}^*$.

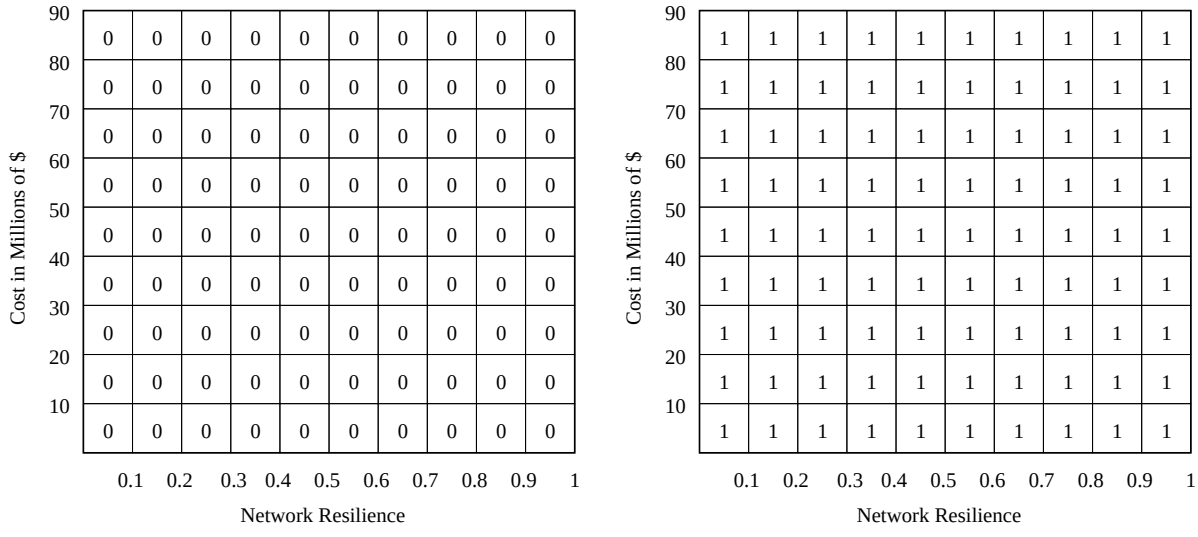
1: Randomly generate an initial population of solutions $\boldsymbol{P}_0$ of size N.
2: Rank and sort $\boldsymbol{P}_0$ using the FNSA [Algorithm 12].
3: Calculate the crowding distance of the solutions in $\boldsymbol{P}_0$ [Algorithm 13]
4: Create population $\boldsymbol{Q}_0$ of size N using binary tournament selection (with the crowded comparison operator $\succ_c$) from $\boldsymbol{P}_0$, crossover and mutation.
5: $t \leftarrow 0$
6: **while** $t < G_{\max}$ **do**
7: $\quad \boldsymbol{R}_t \leftarrow \boldsymbol{P}_t \cup \boldsymbol{Q}_t$
8: $\quad$ Partition $\boldsymbol{R}_t$ into fronts $\mathcal{F}_1, \mathcal{F}_2, \ldots$ by means of the FNSA.
9: $\quad \boldsymbol{P}_{t+1} \leftarrow \emptyset$ and $i \leftarrow 1$
10: $\quad$ **while** $|\boldsymbol{P}_{t+1}| < N$ **do**
11: $\quad\quad$ **if** $|\mathcal{F}_i| + |\boldsymbol{P}_{t+1}| \leq N$ **then**
12: $\quad\quad\quad \boldsymbol{P}_{t+1} \leftarrow \boldsymbol{P}_{t+1} \cup \mathcal{F}_i$
13: $\quad\quad$ **else if** $|\mathcal{F}_i| + |\boldsymbol{P}_{t+1}| > N$ **then**
14: $\quad\quad\quad$ Calculate crowding distance for all solutions in $\mathcal{F}_i$, and sort $\mathcal{F}_i$ members in order of decreasing crowding distance.
15: $\quad\quad\quad \boldsymbol{P}_{t+1} \leftarrow \boldsymbol{P}_{t+1} \cup \{$ the first $(N - |\boldsymbol{P}_{t+1}|)$ elements of $\mathcal{F}_i\}$
16: $\quad\quad$ **end if**
17: $\quad\quad i \leftarrow i + 1$
18: $\quad$ **end while**
19: $\quad$ Calculate the crowding distance for each $\boldsymbol{x} \in P_{t+1}$.
20: $\quad$ For each gene g_k in the solution chromosome, generate a probability distribution $f_k(a) = \frac{a_n}{N}$, where a_n is a count of the number of times allele a occurs for gene g_k in the population. Use $f_k(a)$ to generate a cumulative density function $F_k(a)$.
21: $\quad$ Create N solutions by sampling randomly from the inverse function $F_k^{-1}(a)$ for each gene value, producing population Q_{t+1}.
22: $\quad t \leftarrow t + 1$
23: **end while**
24: $\boldsymbol{A}^* = \boldsymbol{P}_{G_{\max}}$

5.7.4 Dynamic Multi-objective Evolutionary Algorithm

The *Dynamic Multi-objective Algorithm* (DMOEA) developed by Yen and Lu [271] in 2003 is an attempt to establish a cellular multi-objective evolutionary algorithm. Such an algorithm is cellular in the sense that the objective space is divided into a grid of a user-specified (epsilon-based) granularity with the intention of improving algorithmic efficiency (§5.4.7). This is achieved in DMOEA by using the grid as an environmental model to store solution quality information (Pareto-rank and density) organized per grid cell, such that the solutions need not be compared directly to one another, but rather interact only with the grid in order to determine their comparative quality. Grid cells may dominate each other in the usual epsilon-dominance fashion, and a solution takes on as rank the domination count of the cell in which it lies. Similarly, the number of solutions in a particular grid cell provides a density estimate for these solutions. The DMOEA has a slightly inauspicious name, since this could easily pertain to an entire class of MOEAs that exhibit some dynamic behaviour. Certainly this is the case for this particular algorithm, which employs population growth and decline strategies in order to obtain a so-called 'optimal' population size, although the optimality of this size is defined in this case in terms of user solution density preferences rather than in terms of algorithmic performance.

The cellular solution quality storage method is illustrated in Figures 5.8 and 5.9. The grid for storing cell density information, which is initially zero for all cells before any solutions are generated, is shown in Figure 5.8(a). The rank of every cell, is initialized to 1, as shown in Figure 5.8(b).

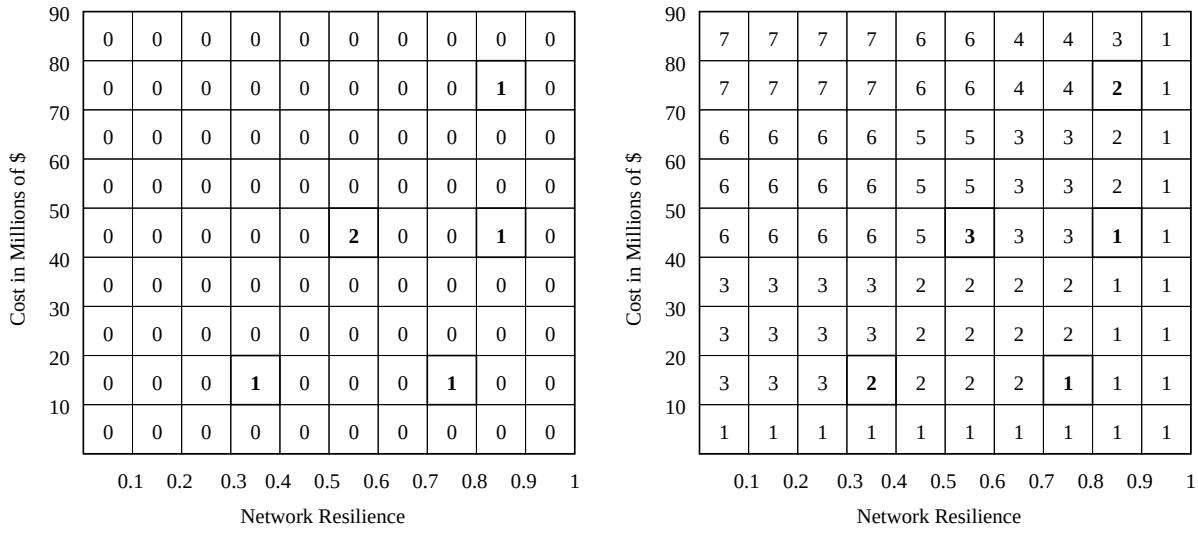
When a solution is added to or removed from the population, the solution is mapped to objective space coordinates which lie in a particular cell. The density and rank grids must then be updated. In the case of an addition, the density grid is updated by incrementing the value of the relevant cell, and the rank grid is updated by incrementing the rank value of each cell dominated by the relevant cell. Solution removal results in the same effect, except that it is a decrement operation instead. Figures 5.9(a) and 5.9(b) show the effect of adding five solutions to the population, two of which fall into the same cell.



(a) *Cell density (initial state)*

(b) *Cell rank (initial state)*

Figure 5.8*: Cellular grid for solution quality storage (initial state).*



(a) *Cell density (generation 1)*

(b) *Cell rank (generation 1)*

Figure 5.9*: Cellular grid for solution quality storage (generation 1).*

An algorithm was developed for this dissertation based on the original DMOEA [271], using an identical grid model for the cellular rank and density information, but differing in terms of the population growth and decline strategies, since the original ones failed to produce satisfactory convergence towards reasonable population sizes. This algorithm shall be called *Another DMOEA* (ADMOEA) to distinguish it from its forerunner. This algorithm incorporates a number of advanced features, including:

1. A growth strategy whereby a number of new solutions are generated as a function of the grid-size and the current Pareto-set size.

2. This growth strategy incorporates three different search mechanisms and a probability vector to control mechanism selection, updated during each generation, depending on each mechanism's success rate.

3. A solution age that is incremented during each generation. Solutions may not be removed from the population before they reach a certain age, allowing them sufficient time to propagate their genes. This age threshold gradually decreases as a function of convergence.

4. A population decline strategy that selects solutions for removal on the basis of their age, cellular rank and density.

5. A regeneration strategy that recreates the entire population using PUMD once limited improvement has occurred for fifty generations.

6. An external epsilon archive to hold Pareto-optimal solutions, updated before each regeneration.

7. Compression and growth mechanisms to alter the dimensions of the grid in order to zoom in on the important region of the objective space, or to accommodate new solutions outside of the current grid dimensions.

In this dissertation, ADMOEA is implemented using the following parameter values: $a_t = 5$, $n_c = n_r = 200$, difference factor $f = 0.7$, and SBX exponent $n_c = 2$. The ADMOEA algorithm appears in pseudo-code form as Algorithm 21.

Algorithm 21 Another Dynamic Multi-objective Evolutionary Algorithm (ADMOEA) (Adapted from [271])

Input: A MOO problem where solutions are an assignment of values to decision variables $\boldsymbol{x}$, a set of constraints on these decision variables, and M objective functions to produce entries for the $M \times 1$ objective vector $\boldsymbol{y}$, a maximum number of generations $G_{\max}$, initial estimates of the minimum and maximum reliability and cost values, $r_{\min}$, $r_{\max}$, $c_{\min}$, $c_{\max}$ ($\Delta r = r_{\max} - r_{\min}$ and $\Delta c = c_{\max} - c_{\min}$) for rank and density grid dimensions, the number of grid cells in each dimension n_{r} and n_{c} (such that $\epsilon_{\mathrm{c}} = \Delta c / n_{\mathrm{c}}$ and $\epsilon_{\mathrm{r}} = \Delta r / n_{\mathrm{r}}$), and an age threshold a_t.

Output: An approximation of the Pareto-optimal solution set in multi-objective space, $\boldsymbol{A}^*$.

1: Initialize the density and rank grids, $\boldsymbol{D}$ and $\boldsymbol{R}$, with zeros and ones respectively.
2: Initialize the probability vector as $\boldsymbol{p}_{\mathrm{v}} = [\frac{1}{4}, \frac{1}{4}, \frac{1}{4}]$.
3: Generate an initial population of solutions $\boldsymbol{P}_0$ of size N by means of LHS.
4: Set $t \leftarrow 0$
5: **while** population converged = FALSE **and** $t < G_{\max}$ **do**
6: *Cellular Density Update*: Update the density $\boldsymbol{D}$ by incrementing the value of each cell in which a solution lies. Grid coordinates for solution $\boldsymbol{x}$ are found as $x = \mathrm{Max}(\mathrm{Min}(\lfloor(\boldsymbol{x}.\mathrm{reliability} - r_{\min})/\epsilon_{\mathrm{r}}\rfloor, n_{\mathrm{c}} - 1), 0)$ and $y = \mathrm{Max}(\mathrm{Min}(\lfloor(\boldsymbol{x}.\mathrm{cost} - c_{\min})/\epsilon_{\mathrm{c}}\rfloor, n_{\mathrm{r}} - 1), 0)$
7: *Cellular Rank Update*: Update the rank $\boldsymbol{R}$ by incrementing the value of every cell epsilon-dominated by each cell in which a solution lies.
8: Compute the number of children to be generated as $g = \sqrt{(n_{\mathrm{c}} + n_{\mathrm{r}} + |\boldsymbol{P}^*|)/2}$, where $|\boldsymbol{P}^*|$ is the number of solutions having rank 1 in the current population.
9: Initialize the child population as $\boldsymbol{Q} \leftarrow \emptyset$.
10: Increment the age of every solution in $\boldsymbol{P}$ by one.
11: Begin *Growth Strategy* [Algorithm 22]
12: Begin *Decline Strategy* [Algorithm 23]
13: Determine the minimum and maximum reliability values in the population $\boldsymbol{P}_{t+1}$, r_{pmin} and r_{pmax}, and the minimum and maximum cost values in the population, c_{pmin} and c_{pmax}.
14: **if** improvement in hypervolume less than 0.01% for 50 generations **then**
15: Execute *Compression and Regeneration Strategy* [Algorithm 27]
16: **end if**
17: Set $t \leftarrow t + 1$
18: **end while**
19: Return $\boldsymbol{A}^* \leftarrow \boldsymbol{P}_t \cup \boldsymbol{A}$.

5.7.5 ANIMA: A Self-adaptive Evolutionary Algorithm

ANIMA is an auto-adaptive MOEA based on the NSGA-II framework and was developed for this dissertation. It employs two different variation mechanisms, namely the SBX crossover with triangular mutation and a differential evolution operator. What makes ANIMA unique is that it encodes the variation operator parameters (along with the solution genes) effectively, making each solution an agent carrying both the search instructions and the solution information. These

Algorithm 22 ADMOEA Growth Strategy

Input: A population of solutions $\boldsymbol{P}_t$ where each is an assignment of values to decision variables $\boldsymbol{x}$, and their associated objective function vectors consisting of reliability and cost values, a search mechanism probability vector $\boldsymbol{p}_{\mathrm{v}}$, and cellular density and rank grids $\boldsymbol{D}$ and $\boldsymbol{R}$.

Output: The union $\boldsymbol{P}_{t+0.5}$ of the original population and the newly created offspring population, an updated search mechanism probability vector $\boldsymbol{p}_{\mathrm{v}}$, and updated cellular density and rank grids $\boldsymbol{D}$ and $\boldsymbol{R}$.

1: Initialise generated offspring as $n_i = 0$ for $i = 1, \ldots, 3$, where i indicates the search mechanism.
2: Initialize successful offspring as $s_i = 0$ for $i = 1, \ldots, 3$, where i indicates the search mechanism.
3: Build a PUMD model by creating partitions (using the represented range of $\boldsymbol{P}_t$) and generating UMD sub-models for each partition.
4: **while** $|\boldsymbol{Q}| < g$ **do**
5: Generate a uniform random number $r \in [0, 1]$
6: For r and the inverse CDF of $\boldsymbol{p}_{\mathrm{v}}$, select search mechanism $m_{\mathrm{n}} \in [1, 2, 3]$.
7: **if** $m_{\mathrm{n}} = 1$ [*Grid Search*] **then**
8: Increment n_1
9: Call Grid Search Step [Algorithm 24]
10: **if** any offspring are successful **then** increment s_1
11: **else if** $m_{\mathrm{n}} = 2$ [*Differential Evolution*] **then**
12: Increment n_2
13: Call Differential Evolution Search Step [Algorithm 25]
14: **if** any offspring are successful **then** increment s_2
15: **else if** $m_{\mathrm{n}} = 3$ [*PUMD Search*] **then**
16: Increment n_3
17: Call PUMD Search Step [Algorithm 26]
18: **if** any offspring are successful **then** increment s_3
19: **end if**
20: *Note: If any successful child is created which falls outside of the grid dimensions ($r_{\min}$, $r_{\max}$)-($c_{\min}$, $c_{\max}$) then an abberation counter is incremented, and the solution quality information is mapped to the edge of the grids. Once the abberation counter reaches one hundred, the grids $\boldsymbol{D}$ and $\boldsymbol{R}$ are resized to accommodate all solutions, re-initialized and updated using the union of the $\boldsymbol{P}$, $\boldsymbol{Q}$.*
21: **end while**
22: Compute the total of the search mechanism success rates as $T = \sum_i s_i/n_i$ for all $n_i \neq 0$.
23: Update the probability vector $\boldsymbol{p}_{\mathrm{v}}$ for each mechanism where $n_i \neq 0$ as $\boldsymbol{p}_{\mathrm{v}}(i) \leftarrow 0.9 \times \boldsymbol{p}_{\mathrm{v}}(i) + 0.1 \times (s_i/n_i/T)$.
24: Normalize $\boldsymbol{p}_{\mathrm{v}}$.
25: $\boldsymbol{P}_{t+0.5} \leftarrow \boldsymbol{P}_t \cup \boldsymbol{Q}$.

parameter values are generated randomly within reasonable ranges for the initial population and any duplicate replacement solutions, but are passed on to newly created offspring by their parent solutions. At the time of initial or replacement solution creation, a solution is randomly assigned an evolution state — indicating either the SBX operator or the DE operator.

During reproduction, a dominant parent is selected according to a uniform distribution. Two additional parent solutions are then selected using two binary tournaments. Variation is achieved by conducting an SBX crossover with polynomial mutation between the dominant parent and

Algorithm 23 ADMOEA Decline Strategy

Input: A population of solutions $\boldsymbol{P}_{t+0.5}$ where each is an assignment of values to decision variables $\boldsymbol{x}$, and cellular density and rank grids $\boldsymbol{D}$ and $\boldsymbol{R}$.

Output: The reduced population $\boldsymbol{P}_{t+1}$ and updated cellular density and rank grids $\boldsymbol{D}$ and $\boldsymbol{R}$.

for all $\boldsymbol{x}_i \in \boldsymbol{P}_{t+0.5}$ **do**
  if Cell rank $\boldsymbol{x}_i > 1$ **and** Age($\boldsymbol{x}_i$) > Age threshold a_t **then**
    $\ell_1 \leftarrow 1 - 1/Rank(\boldsymbol{x}_i)$
    Generate $r \in [0, 1]$ according to a uniform distribution.
    if $\ell_1 > r$ **then**
      Remove $\boldsymbol{x}_i$ from $\boldsymbol{P}_{t+0.5}$ and update $\boldsymbol{D}$ and $\boldsymbol{R}$.
    end if
  end if
end for
for all $\boldsymbol{x}_i \in \boldsymbol{P}_{t+0.5}$ satisfying $\boldsymbol{D}(\boldsymbol{x}_i) > 1$ **do**
  Set $a = \frac{\text{Age}(\boldsymbol{x}_i) - a_t}{\text{Age}(\boldsymbol{x}_i)+1}$
  $\ell_2 \leftarrow 1 - 1/Rank(\boldsymbol{x}_i) \times \boldsymbol{D}(\boldsymbol{x}_i) \times a$
  Generate $r \in [0, 1]$ according to a uniform distribution.
  if $\ell_2 > r$ **then**
    Remove $\boldsymbol{x}_i$ from $\boldsymbol{P}_{t+0.5}$ and update $\boldsymbol{D}$ and $\boldsymbol{R}$.
  end if
end for
for all $\boldsymbol{x}_i \in \boldsymbol{P}_{t+1}$ with rank 1 **do**
  if $\boldsymbol{D}(\boldsymbol{x}_i) > 1$ **then**
    Perform non-dominated sorting amongst the solutions in the cell of $\boldsymbol{x}_i$ as per NSGA-II, and remove all dominated individuals from $\boldsymbol{P}_{t+0.5}$.
    Update $\boldsymbol{D}$ and $\boldsymbol{R}$.
  end if
end for
$\boldsymbol{P}_{t+1} \leftarrow \boldsymbol{P}_{t+0.5}$
Determine the minimum age and maximum rank amongst all the solutions in $\boldsymbol{P}_{t+1}$
if Minimum age > a_t **and** Maximum rank = 1 **then**
  Execute *Compression and Regeneration Strategy* [Algorithm 27]
end if

the second parent, using the search parameter value (SBX difference index) of the dominant parent. This creates two new offspring, each of which may undergo triangular mutation with a low probability (probability provided by the dominant parent). Then a differential evolution vector displacement occurs by subtracting the difference of the solution vectors of the second two parents from the each of the offspring, again using the search parameter value (difference factor) from the dominant parent. The evolution state is copied directly to the offspring from the dominant parent.

If the evolution state of the dominant parent is that of the SBX operator, then the difference factor parameter is copied directly to the offspring and the SBX difference index is either copied directly or varied, and may also undergo mutation. The search parameter value indicated by the evolution state is copied directly with a high probability (80% chance) to the offspring, and is otherwise varied by allowing the search parameter of the dominant parent and the second parent to undergo an SBX crossover (again using the dominant parent's SBX parameter value). Triangular mutation on the parameter value then occurs with a low probability (probability

supplied by the dominant parent). The reason for separation of the evolution states is that evolving multiple variation parameters simultaneously is unstable — the parameters do not settle on good values. Good parameter value combinations are found by rather randomizing and fixing one parameter value, and adjusting the second by means of evolutionary variation in order to suit the first. The standard NSGA-II environmental section process is applied and the entire procedure is repeated until convergence occurs. The pseudo-code for the ANIMA offspring generation process appears as Algorithm 28 (within the framework of NSGA-II (Algorithm 11), with replacement solutions assigned random parameter vectors as per the population initialization (Step 1) in Algorithm 28).

The following parameter ranges were used in this dissertation: for the difference index a minimum value of $p_{\text{sbxmin}} = 1$ and a maximum $p_{\text{sbxmax}} = d/2$ were selected, where d is the size of the design variable range (*e.g.* $d = |\mathcal{X}^i|$ for discrete variable x_i). For the difference factor the range $[p_{\text{dfmin}}, p_{\text{dfmax}}] = [0.6, 1.2]$ was chosen, and for the triangular mutation probabilities the range $[p_{\text{tmin}}, p_{\text{tmax}}] = [0, 0.02]$ was used.

Algorithm 24 ADMOEA Grid Search Step

Input: A population of solutions $\boldsymbol{P}_t$ where each is an assignment of values to decision variables $\boldsymbol{x}$, offspring population $\boldsymbol{Q}$, cellular density and rank grids $\boldsymbol{D}$ and $\boldsymbol{R}$, and an age threshold a_t.

Output: At most one offspring solution, an updated cellular density and rank grid $\boldsymbol{D}$ and $\boldsymbol{R}$.

1: Select parent solution $\boldsymbol{x}_1$ using constraint competent binary tournament selection (NSGA-II), except that Pareto-rank is replaced with cellular rank and crowding distance is replaced with the sum total of cellular density in a 3×3 neighbourhood of cells.
2: Find all solutions with a cellular rank in the next two lower categories than $\boldsymbol{x}_1$; *i.e.* if the rank of $\boldsymbol{x}_1$ equals 3 then find all solutions of rank 1 and 2. Randomly select solution $\boldsymbol{x}_2$ amongst these as second parent. If the cell rank of $\boldsymbol{x}_1$ equals 1, then select a random solution of rank 1.
3: Perform an SBX crossover with polynomial mutation to produce two offspring solutions.
4: Select one of two offspring solutions with a 50% chance and either a rank or density test with a 50% chance. If a rank test is used, compare the offspring to its parents and add it to $\boldsymbol{Q}$ if it has a rank smaller than or equal to either of the parents' ranks. If a density test is used, then add it to $\boldsymbol{Q}$ if it is not strictly dominated (*i.e.* its grid coordinates are not both lower than those of its parents) and it has a cell density of 1. If the first offspring solution fails the test, then perform a similar test for the second one. Update $\boldsymbol{D}$ and $\boldsymbol{R}$.

Algorithm 25 ADMOEA DE Search Step

Input: A population of solutions $\boldsymbol{P}_t$ where each is an assignment of values to decision variables $\boldsymbol{x}$, offspring population $\boldsymbol{Q}$, cellular density and rank grids $\boldsymbol{D}$ and $\boldsymbol{R}$, a difference factor f, and an age threshold a_t.

Output: At most one offspring solution, updated cellular density and rank grids $\boldsymbol{D}$ and $\boldsymbol{R}$.

1: Randomly select three distinct solutions $\boldsymbol{x}_1$, $\boldsymbol{x}_2$, and $\boldsymbol{x}_3$ from the population $\boldsymbol{P}$.
2: Create an offspring solution by adding the difference vector of the second two parents to the first, *i.e.* $\boldsymbol{x}_4 = \boldsymbol{x}_1 + f(\boldsymbol{x}_2 - \boldsymbol{x}_3)$, where f is the difference factor.
3: **if** $\boldsymbol{x}_4$ is not dominated by $\boldsymbol{x}_1$ **then** add it to $\boldsymbol{Q}$ and update $\boldsymbol{D}$ and $\boldsymbol{R}$, else
4: Create an offspring solution by subtracting the difference vector of the second two from the first, *i.e.* $\boldsymbol{x}_5 = \boldsymbol{x}_1 - f(\boldsymbol{x}_2 - \boldsymbol{x}_3)$.
5: **if** $\boldsymbol{x}_5$ is not dominated by $\boldsymbol{x}_1$ **then** add it to $\boldsymbol{Q}$ and update $\boldsymbol{D}$ and $\boldsymbol{R}$.

Algorithm 26 ADMOEA PUMD Search

Input: A population of solutions $\boldsymbol{P}_t$ where each is an assignment of values to decision variables $\boldsymbol{x}$, offspring population $\boldsymbol{Q}$, cellular density and rank grids $\boldsymbol{D}$ and $\boldsymbol{R}$, and an age threshold a_t.

Output: At most one offspring solution, updated cellular density and rank grids $\boldsymbol{D}$ and $\boldsymbol{R}$.

1: Randomly select a solution $\boldsymbol{x}_1$ from $\boldsymbol{P}$.
2: Sample from the prebuilt PUMD model up to a maximum of three times in order to produce a maximum of three offspring. The first offspring solution that dominates $\boldsymbol{x}_1$ is then added to $\boldsymbol{Q}$, $\boldsymbol{D}$ and $\boldsymbol{R}$ are updated, and the process is halted.

Algorithm 27 ADMOEA Compression and Regeneration Strategy

Input: A population of solutions $\boldsymbol{P}_{t+1}$ where each is an assignment of values to decision variables $\boldsymbol{x}$, cellular density and rank grids $\boldsymbol{D}$ and $\boldsymbol{R}$, and an age threshold a_t.

Output: An updated archive $\boldsymbol{A}$, a regenerated population $\boldsymbol{P}$ and updated cellular density and rank grids $\boldsymbol{D}$ and $\boldsymbol{R}$, as well as a new value of the age threshold a_t.

1: Determine minimum and maximum reliability values represented in population $(r_{\text{pmin}}, r_{\text{pmax}})$, and minimum and maximum cost values from population $(c_{\text{pmin}}, c_{\text{pmax}})$.
2: Decrement a_t to a minimum of 1.
3: **if** $(r_{\text{max}} - r_{\text{pmax}}) > 0.1 \times (r_{\text{max}} - r_{\text{min}})$ **or** $(r_{\text{pmin}} - r_{\text{min}}) > 0.1 \times (r_{\text{max}} - r_{\text{min}})$ **then**
4: compressreliability $\leftarrow$ TRUE
5: **else**
6: compressreliability $\leftarrow$ FALSE
7: **end if**
8: **if** $(c_{\text{max}} - c_{\text{pmax}}) > 0.1 \times (c_{\text{max}} - c_{\text{min}})$ **or** $(c_{\text{pmin}} - c_{\text{min}}) > 0.1 \times (c_{\text{max}} - c_{\text{min}})$ **then**
9: compresscost $\leftarrow$ TRUE
10: **else**
11: compresscost $\leftarrow$ FALSE
12: **end if**
13: **if** compressreliability = TRUE **then**
14: $r_{\text{max}} \leftarrow (r_{\text{max}} + r_{\text{pmax}})/2$
15: $r_{\text{min}} \leftarrow (r_{\text{min}} + r_{\text{pmin}})/2$
16: **end if**
17: **if** compresscost = TRUE **then**
18: $c_{\text{max}} \leftarrow (c_{\text{max}} + c_{\text{pmax}})/2$
19: $c_{\text{min}} \leftarrow (c_{\text{min}} + c_{\text{pmin}})/2$
20: **end if**
21: **if** compressreliability = TRUE **or** compresscost = TRUE **then**
22: Re-initialize extent and values of cellular density and rank grids $\boldsymbol{D}$ and $\boldsymbol{R}$.
23: **end if**
24: Update the epsilon archive $\boldsymbol{A}$ with the current population.
25: Generate a PUMD probability model using the current population.
26: Recreate the entire population by sampling from the PUMD model.
27: Update the cellular density and rank grids $\boldsymbol{D}$ and $\boldsymbol{R}$.

5.8 AMALGAM: An Evolutionary Hyperheuristic

The AMALGAM algorithm by Vrugt and Robinson [244] is a generic evolutionary meta-algorithm framework which incorporates k sub-algorithms in the solution of a MOO problem. The algorithm employs a population of N solutions, whose offspring are created in a *genetically adaptive* manner by dividing the creation of N offspring amongst the sub-algorithms, proportional to the success of these sub-algorithms during previous generations. *Global information sharing* means that at all times each sub-algorithm has access to the entire population to generate its share of the offspring. Theory and numerical experiments have shown that it is impossible to develop a single algorithm which performs efficiently over a diverse set of problems. The philosophy behind AMALGAM is that the strengths of different meta-heuristics can be combined and exploited dynamically to produce a faster, more reliable search than is possible with any one of the algorithms on its own.

AMALGAM borrows largely from the Non-dominated Sorting Genetic Algorithm (NSGA-II) by Deb *et al.* [61] in 2002, using its concepts of *Pareto-rank fitness*, *crowding distance*, the *Fast Non-dominated Sorting Algorithm* (FNSA), and an *elitist selection strategy* to construct a generic multi-method framework.

Vrugt and Robinson [244] used four sub-algorithms within the AMALGAM framework, namely NSGA-II, Adaptive Metropolis, Particle Swarm Optimisation and Differential Evolution, each with differing strengths and search techniques. They applied this formulation of AMALGAM to a benchmark suite of continuous optimisation problems, and for some of the most challenging problems exhibited a tenfold performance improvement versus any of the sub-algorithms used individually.

To the best knowledge of the author, this is the first study in which AMALGAM is applied in a solution of a real-world problem (WDS design).

The steps of the AMALGAM algorithm are shown in pseudo-code form in Algorithm 30. The first step is the generation of an initial population P_0 of size N, for which *Latin Hypercube Sampling* (LHS) is employed to provide a well-distributed initial population. In Step 2 of AMALGAM, the population is ranked using the FNSA, which is given in pseudo-code as Algorithm 12. The Pareto-rank may be used by a sub-algorithm as a measure of fitness. In Step 3, each sub-algorithm uses the initial population to generate N/k offspring in order to create a new child population $\boldsymbol{Q}_0$. The algorithm then proceeds iteratively for $G_{\max}$ generations, selecting consecutive populations from the union of the parents and offspring of each generation t, *i.e.* $\boldsymbol{R}_t = \boldsymbol{P}_t \cup \boldsymbol{Q}_t$. This is achieved by partitioning $\boldsymbol{R}_t$ into its various fronts $\mathcal{F}_1, \mathcal{F}_2, \ldots$ by means of the FNSA. The next population $\boldsymbol{P}_{t+1}$ is constructed by filling up N spaces, starting with the members of $\mathcal{F}_1$ and proceeding in increasing order of fronts. When adding all the members of the front $\mathcal{F}_i$ will take the size of $\boldsymbol{P}_{t+1}$ beyond N, this front is ordered in terms of decreasing crowding distance and the first $N - |\boldsymbol{P}_{t+1}|$ elements of $\mathcal{F}_i$ are selected. A procedure for efficiently computing crowding distance is provided in pseudo-code as Algorithm 13. This selection procedure automatically guarantees elitism since none of the non-dominated solutions are lost, unless the first front has a size larger than N, in which case those solutions in the most crowded regions are discarded. The next stage is to reward sub-algorithms which have demonstrated the most reproductive success during the current generation. For this purpose it is required to count, for each sub-algorithm, the number of solutions S_{t+1}^j contributed by the j-th sub-algorithm to the population P_{t+1}.

If N_t^j is the number of offspring that sub-algorithm j must generate during generation t, then

$$N_{t+1}^j = N\left(S_{t+1}^j / N_t^j\right) \Bigg/ \sum_{h=1}^{k}\left(S_{t+1}^h / N_t^h\right), \tag{5.4}$$

where the ratio of the number of sub-algorithm j's successful offspring in the new population to the number generated is scaled to the combined success ratios of the entire algorithm set. The implementation by Vrugt and Robinson employed a minimum size of $N_t^j = 5$ to avoid inactivating any of the algorithms. New offspring are then generated by each algorithm to create the child population Q_{t+1}, and the process repeats from Step 5 of Algorithm 30 [244].

It is recommended that all sub-algorithms used within the AMALGAM framework should be capable of solving the relevant optimisation problem in their own capacity, otherwise their internal administration will waste precious computational time.

There are some unclarities regarding the implementation of AMALGAM. Firstly, environmental selection is conducted by the AMALGAM FNSA and Crowding Distance routines, and the sub-algorithms are only required to generate offspring, using their selection and variational operators, and apparently selecting solutions from the globally shared population. However, many metaheuristics require different environmental selection schemes, or model-building techniques, such that an entire sub-model must be built for that sub-algorithm in every generation. For example, SPEA-II would require the maintenance of an internal archive, and would need an additional level of environmental selection utilizing SPEA-II computed fitness. PSO presents even more of a challenge since it requires the tight coupling of meta-data to solutions in the form of solution velocities, such that an additional PSO population would be required, and velocities would have to be assigned randomly or heuristically if solutions are to be included from the shared population. On the other hand, NSGA-II and DE may use the shared population directly without additional environmental selection or internal model building. For these algorithms, one is only utilizing their offspring creation subroutines. An illustration of the problems with this idea comes from the fact that NSGA-II and SPEA-II essentially use similar mechanisms to generate offspring, and *differ primarily in terms of how they conduct environmental selection.*

Assume that metaheuristics may be categorized either as using *regenerative models* (EDAs, MOEAs, Greedy) or *persistent models* (PSO), where regenerative models require only the current populations (parent / offspring / archive) and some internal variables to generate the next population, and where representation in persistent models requires continuous monitoring and refinement of population meta-data (meta-parameters are tightly coupled to solutions and are assigned values only by the relevant metaheuristic), such as the case of PSO. The original formulation of AMALGAM seems only to be suitable for regenerative models, since there is a break in the continuity of persistent model parameters. However, Vrugt and Robinson [244] implemented a naive version of PSO as an AMALGAM sub-algorithm in their article, failing to explain how they solved the discontinuity problem. Two possible ways to accommodate this may include *meta-parameter generalization* (whereby all sub-algorithms supply values for all tightly coupled meta-parameters), or *persistent model reformulation* (whereby solutions obtained from other sub-algorithms must be incorporated and assigned meta-parameter values by the host sub-algorithm). This problem was bypassed in this dissertation by only considering regenerative models.

Alternative formulations for AMALGAM were devised for this dissertation, in order to address some of the shortcomings exposed during testing. These included the arbitrariness of the environmental selection scheme, and various issues pertaining to the offspring partitioning formula. The AMALGAM scheme need not necessarily be deployed within the NSGA-II framework.

An alternative formulation was developed, called *AMALGAMS*, which employs the SPEA-II environmental selection strategy.

One issue of importance not addressed in AMALGAM is the problem of sub-algorithm efficiency. Currently a very slow sub-algorithm which produces the best offspring will dominate the search. However, it may be possible that a faster algorithm is capable of producing equally good solutions if allocated a larger portion of the offspring, resulting in significant speed enhancements overall. The offspring partitioning formula (5.4) may be adapted to

$$N_{t+1}^j = N\left(S_{t+1}^j/(N_t^j T_t^j)\right) \Bigg/ \sum_{h=1}^{k}\left(S_{t+1}^h/(N_t^h T_t^h)\right)$$

in order to include the individual running times of the sub-algorithms such that relatively longer running times are penalized. This variant was called *TAMALGAM*.

A more intelligent offspring partitioning scheme might include the use of performance metrics, such as hypervolume or dominance rank information, to better quantify offspring successes. Two additional formulations were devised. In the first formulation, *AMALGAMI*, the offspring success count per algorithm S_{t+1}^j is replaced by the sum of the squared inverse dominance rankings of the offspring that survive to the next generation, *i.e.*

$$\hat{S}_{t+1}^j = \sum_{i \in O_{t+1}^j} \left(\frac{1}{\text{rank}(i)}\right)^2,$$

where O_{t+1}^j is the set of successful offspring produced by algorithm j. The second formulation, *AMALGAMJ*, also replaces N_t^j by $(N_t^j)^{\frac{1}{2}}$ in order to reduce the importance of the number of offspring generated.

Finally, the two considerations of sub-algorithm efficiency and finely-grained performance evaluation may be combined to produce additional variants *TAMALGAMI* and *TAMALGAMJ*. Numerous other formulations were investigated, but the four best of those included in the final study were AMALGAMS, TAMALGAM, AMALGAMI, and TAMALGAMJ.

5.9 Chapter Summary

An overview to the topic of multi-objective optimisation in the context of WDS systems was provided in this chapter, in fulfilment of Dissertation Objective 4 in §1.3. and partially addressing Objective 3(b). Furthermore, the algorithms named in Dissertation Objectives 6 and 7 were introduced and discussed in some detail. The basic theory of MOO was presented, and a generic mathematical model for multi-objective WDS design optimisation was developed. A history of MOO for WDS design was included, detailing the developments over the last decade.

Important design concepts for MOEAs (and multi-objective metaheuristics in general) were discussed, such as Pareto-based fitness assignment, diversity preservation using solution density information, constraint handling with penalty factors or constrained-dominance, selection schemes, elitism, variational operators and solution encoding. Various population sizing strategies were discussed, as well as the use of an ϵ-domination scheme of user-defined precision. Performance evaluation in a MOO context was considered, covering topics such as optimisation search convergence, parameter tuning, experimental design for fair algorithm comparison, and solution quality measures. Three quality measures for comparing approximation sets were also

identified: a dominance ranking quantifier, the hypervolume metric, and a new measure based on the size of an output ϵ-archive.

Three popular MOEAs from the literature were discussed in detail, namely NSGA-II, SPEA-II and DE. Several alternative population-based metaheuristics were also presented, including a novel greedy algorithm based on engineering judgement, a multi-objective PSO algorithm, and an EDA based on the Univariate Marginal Distribution, as well as a novel variant of this, named the Partitioned UMD. A self-adaptive evolutionary algorithm called ANIMA was also developed, which evolves search parameters in addition to solutions. Finally, a hyperheuristic named AMALGAM was presented which simultaneously incorporates multiple diverse metaheuristics within a single optimisation framework in the hope of effectively combining their strengths. Several variants on the original AMALGAM formulation were proposed.

This chapter serves the purpose of preparing the reader for the chapters that follow, which deal with the implementation of the algorithms mentioned above for WDS design and actual benchmark testing in which the performances of these metaheuristics are compared.

Algorithm 28 ANIMA Self-adaptive MOEA

Input: An initial random population of solutions $\boldsymbol{P}_t$ of size N where each solution is an assignment of values to decision variables $\boldsymbol{x}$ and parameter values $\boldsymbol{p} = (p_{\text{sbx}}, p_{\text{df}}, p_{\text{t}})$ representing the SBX parameter (difference index), the Differential Evolution parameter (difference factor), and the triangular mutation probability parameter, respectively. Also required is the difference index value range $[p_{\text{sbxmin}}, p_{\text{sbxmax}}]$, the difference factor range $[p_{\text{dfmin}}, p_{\text{dfmax}}]$, and the triangular mutation probability parameter range $[p_{\text{tmin}}, p_{\text{tmax}}]$, as well as a maximum number of generations G_{max}.

Output: An approximation of the Pareto-optimal solution set in multi-objective space, $\boldsymbol{A}^*$.

1: Initialize the parameter values for each solution in $\boldsymbol{P}_t$ by sampling uniformly from the permissable parameter ranges. Also assign an evolution state to each solution as SBX or DE with an equal probability.
2: Set $t \leftarrow 0$
3: **while** Population converged = FALSE **and** $t < G_{\text{max}}$ **do**
4: $\quad \boldsymbol{Q}_t \leftarrow \emptyset$
5: $\quad$ **while** $|\boldsymbol{Q}_t| < N$ **do**
6: $\quad\quad$ Select the dominant parent $\boldsymbol{x}_1$ uniformly from $\boldsymbol{P}_t$. Select next two parents $\boldsymbol{x}_2$ and $\boldsymbol{x}_3$ by means of two constraint competent binary tournaments.
7: $\quad\quad$ Perform an SBX crossover between $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ using the p_{sbx} of the dominant parent. This produces two offspring solutions $\boldsymbol{x}_4$ and $\boldsymbol{x}_5$.
8: $\quad\quad$ Conduct variable-wise triangular mutation on these offspring with the probability p_{t} of the dominant parent.
9: $\quad\quad$ Perform differential evolution vector operations as $\boldsymbol{x}_4 \leftarrow \boldsymbol{x}_4 - f(\boldsymbol{x}_2 - \boldsymbol{x}_3)$, and $\boldsymbol{x}_5 \leftarrow \boldsymbol{x}_5 + f(\boldsymbol{x}_2 - \boldsymbol{x}_3)$, where $f = p_{\text{df}}$ of the dominant parent.
10: $\quad\quad$ *NSGA-II Environmental Selection* to generate $\boldsymbol{P}_{t+1}$ (replacing any duplicate solutions in the problem domain with UMD generated solutions whose parameter vectors are generated as per Step 1).
11: $\quad\quad$ Conduct *Offspring Parameter Creation* [Algorithm 29]
12: $\quad$ **end while**
13: $\quad$ Set $t \leftarrow t + 1$
14: **end while**

Algorithm 29 ANIMA Parameter Generation

Input: Parent solutions $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ along with their parameter vectors $\boldsymbol{p}_1$ and $\boldsymbol{p}_2$, offspring solutions $\boldsymbol{x}_4$ and $\boldsymbol{x}_5$, valid parameter ranges $[p_{\text{sbxmin}}, p_{\text{sbxmax}}]$, $[p_{\text{dfmin}}, p_{\text{dfmax}}]$ and $[p_{\text{tmin}}, p_{\text{tmax}}]$.
Output: Offspring parameter vectors $\boldsymbol{p}_4$ and $\boldsymbol{p}_5$.

if evolstate($\boldsymbol{x}_1$) = SBX **then**
  for all offspring $\boldsymbol{x}_i$ **do**
    Generate random number $u_1 \in [0, 1]$
    if $u_1 < 0.8$ **then**
      Copy $\boldsymbol{p}_1$ directly to the offspring.
    else
      Offspring parameters p_{sbx} and p_{t} are generated as SBX crossovers of those values for $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ using the p_{sbx} of $\boldsymbol{x}_1$. Parameter p_{df} is copied directly from $\boldsymbol{x}_1$.
    end if
    Generate random number $u_2 \in [0, 1]$.
    if $u_2 < p_{\text{t}}$ of dominant **then**
      Allow offspring parameters p_{sbx} and p_{df} to undergo triangular mutation.
    end if
  end for
else if evolstate($\boldsymbol{x}_1$) = DE **then**
  for all offspring $\boldsymbol{x}_i$ **do**
    Generate random number $u_1 \in [0, 1]$
    if $u_1 < 0.8$ **then**
      Copy $\boldsymbol{p}_1$ directly to the offspring.
    else
      Offspring parameter p_{df} is generated as an SBX crossover of this parameter for $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ using the p_{sbx} of $\boldsymbol{x}_1$. Parameters p_{sbx} and p_{t} are copied directly from $\boldsymbol{x}_1$.
    end if
    Generate random number $u_2 \in [0, 1]$.
    if $u_2 < p_{\text{t}}$ of dominant **then**
      Allow offspring parameter p_{df} to undergo triangular mutation.
    end if
  end for
end if

Algorithm 30 Amalgam Hyperheuristic [244]

Input: A MOO problem consisting of an initial population of solutions $\boldsymbol{P}$, where each solution is an assignment of values to decision variables $\boldsymbol{x}$, a set of constraints on these decision variables, and M objective functions to produce entries for the $M \times 1$ objective vector $\boldsymbol{y}$, a maximum number of generations $G_{\max}$.

Output: An approximation of the Pareto-optimal solution set in multi-objective space, $\boldsymbol{A}^*$.

Generate an initial population of solutions $\boldsymbol{P}_0$ of size N by means of LHS.
Rank and sort population using the FNSA [Algorithm 12].
Create population $\boldsymbol{Q}_0$ using k sub-algorithms, each contributing N/k offspring.
$t \leftarrow 0$
while $t < G_{\max}$ **do**
  $\boldsymbol{R}_t \leftarrow \boldsymbol{P}_t \cup \boldsymbol{Q}_t$
  Partition $\boldsymbol{R}_t$ into fronts $\mathcal{F}_1, \mathcal{F}_2, \ldots$ by means of the FNSA [Algorithm 12].
  $\boldsymbol{P}_{t+1} \leftarrow \emptyset$ and $i \leftarrow 1$
  while $|\boldsymbol{P}_{t+1}| < N$ **do**
    if $|\mathcal{F}_i| + |\boldsymbol{P}_{t+1}| <= N$ **then**
      $\boldsymbol{P}_{t+1} \leftarrow \boldsymbol{P}_{t+1} \cup \mathcal{F}_i$
    else if $|\mathcal{F}_i| + |\boldsymbol{P}_{t+1}| > N$ **then**
      Calculate crowding distance in $\mathcal{F}_i$ [Algorithm 13].
      Sort $\mathcal{F}_i$ members in order of decreasing crowding distance.
      $\boldsymbol{P}_{t+1} \leftarrow \boldsymbol{P}_{t+1} \cup \{$ the first $(N - |\boldsymbol{P}_{t+1}|)$ elements of $\mathcal{F}_i\}$
    end if
    $i \leftarrow i + 1$
  end while
  Calculate the crowding distance for each $\boldsymbol{x} \in \boldsymbol{P}_{t+1}$.
  Calculate the number of offspring, $\boldsymbol{S}^j_{t+1}$, contributed by sub-algorithm j to $\boldsymbol{P}_{t+1}$.
  $j \leftarrow 1$
  while $j \leq k$ **do**
    Calculate $N^j_{t+1} = N \left(S^j_{t+1}/N^j_t \right) / \sum_{h=1}^{k} \left(S^h_{t+1}/N^h_t \right)$
    $j = j + 1$
  end while
  Create $\boldsymbol{Q}_{t+1}$ by generating N^j_{t+1} offspring with each algorithm $j = 1, \ldots, k$.
  $t \leftarrow t + 1$
end while
$\boldsymbol{A}^* = \boldsymbol{P}_t$

Chapter 6

Implementation of Multi-objective WDSDO

This chapter contains a discussion of the implementations of the multi-objective algorithms presented in the previous chapter, as well as the development of a testing strategy for comparing the performance of these algorithms in terms of their ability to conduct WDSDO. In order to provide global perspective, the overall testing strategy is presented first, followed by the WDS benchmark descriptions and required adaptations of the generic WDSDO problem formulation for each benchmark. Finally, technical details are presented with respect to global optimisation requirements and specific algorithmic implementations.

6.1 Algorithm Testing Strategy for WDSDO

The following algorithms were selected for comparison in terms of their ability to conduct WDSDO on a number of WDS benchmarks that are documented in the literature:

1. ADMOEA
2. $\text{AMALGAM}_{\text{ndp}}$
3. $\text{AMALGAM}_{\text{ndu}}$
4. $\text{AMALGAM}_{\text{ndug}}$
5. $\text{AMALGAMI}_{\text{ndp}}$
6. $\text{AMALGAMI}_{\text{ndu}}$
7. $\text{AMALGAMI}_{\text{ndug}}$
8. $\text{AMALGAMS}_{\text{ndp}}$
9. $\text{AMALGAMS}_{\text{ndu}}$
10. ANIMA
11. DE (GDE)
12. GREEDY
13. NSGA-II
14. PSO (MOPSO)
15. PUMDA
16. SPEA-II

17. TAMALGAM$_{\text{ndp}}$
18. TAMALGAM$_{\text{ndu}}$
19. TAMALGAM$_{\text{ndug}}$
20. TAMALGAMJ$_{\text{ndp}}$
21. TAMALGAMJ$_{\text{ndu}}$
22. TAMALGAMJ$_{\text{ndug}}$
23. UMDA

In each case, the AMALGAM variants are implemented with sub-algorithms indicated by the first-letter subscript (*i.e.* subscript 'n' denotes NSGA-II, 'd' denotes DE, 'u' denotes UMDA, 'p' denotes PUMDA, and 'g' denotes GREEDY). AMALGAM has been implemented with three different selections of sub-algorithms, the first two comprising three sub-algorithms ('ndu' and 'ndp'), and the final selection comprising four sub-algorithms ('ndug').

In this dissertation nine standard WDS benchmarks documented in the literature were employed to test the relevant algorithms. These benchmarks were introduced in Chapter 3, and include TLN, HANOI, NYTUN, TRP, EXNET, BLACK, FOSS, PESCA and MOD. Summarised descriptions of these benchmarks appear later in this chapter. Full benchmark specifications are also provided as EPANET2 input files (files named '*.inp') and accompanying optimisation data files (files named '*.opt') in an electronic supplement to this dissertation (see Appendix E). These WDS benchmarks represent different aspects of real-world systems, varying in numbers and dimensions of components such as pipes and reservoirs. Amongst these are included instances of real-world WDSs.

The testing strategy is divided into four phases. Phase 1 is applicable to the TRP, TLN, HANOI, NYTUN, BLACK, FOSS, PESCA, MOD and EXNET benchmarks, while Phases 2 and 3 apply to the same benchmarks, excluding EXNET. These testing phases address the performance comparison of the various algorithms, constraint handling techniques and surrogate reliability measures. The test results and accompanying analysis for each of these phases appear in Chapters 7 and 8. Furthermore, in Phase 4, a real South African WDS case study, the R21 Corridor, is considered in Chapter 9. The testing phases are described as follows.

Phase 1: WDSDO Algorithm Comparison

The purpose of the first phase of testing is to evaluate the performance of the various algorithms with respect to WDSDO on the named WDS benchmarks as presented in the literature. The following optimisation setup is employed:

1. The WDS benchmarks are utilized with deterministic demands as stated in their problem descriptions.
2. The penalty factor method is used, as per §5.4.4.
3. The Network Resilience surrogate measure is used.
4. Population sizes are calculated for each algorithm and benchmark combination in accordance with Goldberg's method, as explained in §5.4.6.
5. Fair time trials are conducted, as explained in §5.5.1, using a *convergence time trial* followed by a *full time trial*, which employs the time limits obtained in the convergence trial.
6. Thirty different initial populations are randomly generated and used to conduct 30 optimisation runs for each algorithm.

7. The *dominance rank*, *hypervolume*, and ϵ-*archive size* performance measures (described in §5.5.4) are calculated for each approximation set.
8. The WDS benchmarks considered during this phase are TRP, TLN, NYTUN, HANOI, BLACK, FOSS, PESCA, MOD and EXNET.

The results of Phase 1 are reported in Chapter 7.

Phase 2: Constraint Handling Technique Comparison

The purpose of the second phase is to identify which of the penalty factor or constrained domination techniques for constraint handling produce superior results in terms of the ordinary performance metrics as well as the proportion of feasible solutions in the final populations. The following optimisation setup is employed:

1. The WDS benchmarks are utilized with deterministic demands as stated in their problem descriptions.
2. The NSGA-II algorithm is combined with the two different constraint handling techniques in order to produce two algorithms.
3. Thirty fair time trials are conducted as in Phase 1.
4. The *dominance rank*, *hypervolume*, and ϵ-*archive size* performance measures are calculated for each approximation set.
5. The WDS benchmarks considered during this phase are TRP, TLN, NYTUN, HANOI, BLACK, FOSS, PESCA and MOD.

The results of Phase 2 are also reported in Chapter 7.

Phase 3: Reliability Measures Comparison

The third phase entails comparing reliability measures (the Resilience Index, the Network Resilience, Flow Entropy, and the Mixed Surrogate) in order to determine which one produces the most robust designs in terms of probabilistic reliability (with uncertain demands) and the ability to respond gracefully to pipe failures. The following optimisation setup is employed:

1. The WDS benchmarks are utilized with deterministic demands using the surrogate measures, and uncertain demands with the LHS sampling method.
2. The NSGA-II algorithm is executed with four different reliability schemes providing four different result sets.
3. Thirty convergence trials per algorithm are conducted, and the approximation sets produced are combined to produce four attainment sets for each benchmark.
4. Probabilistic reliability is estimated for the solutions in the attainment sets of each algorithm using LHS for nodal demands (distributed normally with deterministic peak demand as the mean and 30% of the mean as the standard deviation) employing 1 000 samples. Furthermore, these demands are correlated at a correlation coefficient of 0.5, as per the method described in §4.2.6.
5. A failure analysis is conducted for all n_{p} single-pipe failure events.
6. Pressure-driven analysis is conducted using a demand adaptation process.

7. Average demand satisfaction percentages (ADSU and ADSF, as defined in §4.3.1) are calculated for uncertain demands analysis and failure analysis.
8. Linear regression analysis is performed in order to determine whether there is a significant relationship between the reliability estimation methods and the ADS values for probabilistic and failure reliability.
9. The WDS benchmarks considered during this phase are TRP, TLN, NYTUN, HANOI, BLACK, FOSS, PESCA, and MOD.

The results of Phase 3 are presented in Chapter 8.

Phase 4: South African Case Study (R21 WDS)

Phase 4 is the subject matter of Chapter 9, in which the design of a real-world South African WDS, the R21 Corridor WDS, is considered, using the redundant layout technique and the best design method from Phase 1. Significant improvement on a preliminary engineering design is demonstrated.

6.2 WDS Benchmarks documented in the literature

It is worth special mention that available studies on WDSDO typically examine a small number of WDS benchmarks, sometimes even considering only a single network [88, 141, 182, 194]. While this might demonstrate potential for a certain optimisation methodology, it fails to allow any general conclusions. However, employing a large number of benchmarks is troublesome due to the high computational requirements of the WDSDO problem and the lack of readily available WDS benchmarks in the public domain.

Four of the WDS benchmarks considered in this dissertation are taken from the free online repository of the *Exeter Centre for Water Systems* (UK) [84], namely TRP, NYTUN, HANOI and EXNET. The remaining WDS benchmarks are taken from the website of the *DEIS Operations Research Group* at the University of Bologna [65], namely TLN, BLACK, FOSS, PESCA and MOD.

Only the basic problem description, including the demand loading conditions and pipe sizing / rehabilitation options for some of the benchmarks are presented here. Detailed WDS design specifications of all benchmarks, including the physical dimensions and layout of each WDS, are made available in standard EPANET input file format on a compact disc which accompanies this dissertation (and which also contains the code and optimisation software employed in this dissertation).

6.2.1 Benchmark 1 — The Two Reservoir Problem

The *Two Reservoir Problem* (TRP) was first proposed by Gessler [99] in 1985, and later studied by Simpson *et al.* [218] in 1994. A schematic of the TRP pipe layout appears in Figure 6.1, with the pipes numbered for identification. The objective is to minimize the cost of the design which requires the addition of five new pipes (represented by the dashed lines numbered 6, 8, 11, 13 and 14) to an existing system, and rehabilitation options (duplication or cleaning or leaving unchanged) for three of the existing pipes (represented by the bold lines numbered 1, 4 and 5). The available diameters for new pipes and rehabilitation options for the three existing pipes

appear in Table 6.2, together with their associated unit costs. The size of the decision space is $10^3 \times 8^5 \approx 10^{7.5}$, which makes TRP the smallest combinatorial problem considered in this dissertation. Three demand patterns must be satisfied from the two available supply sources, as specified in Table 6.1, for given minimum head requirements. The Hazen-Williams roughness coefficients are equal to 120 for new and cleaned pipes, and the coefficients for existing pipes are variable.

The TRP may be implemented using the standard multi-objective formulation (5.1) with five discrete variables for new pipes (which may be assigned one of eight values 1–8, indexing the different pipe options) and three discrete variables for rehabilitation options (which may be assigned one of ten options — leave unchanged, clean or duplicate in one of eight sizes). Only minimum head violations are included in the penalty function, normalized by a maximum head limit of 90 m.

6.2.2 Benchmark 2 — The Two-loop Network

The *Two-loop Network* (TLN) is a simple WDS that was introduced by Alperovits and Shamir [9] in 1977. A schematic of the TLN pipe layout appears in Figure 6.2, with the pipes numbered for identification. The objective of optimisation is to design pipe sizes for the system in order to minimize cost, although the maximisation of reliability is also considered here. A set of 14 possible pipe diameters with associated unit costs appears in Table 6.3. Since there are eight pipes, the size of the decision space is $14^8 \approx 10^9$. A single demand loading condition must be satisfied, which appears in Table 6.4. Furthermore, the TLN WDS is required to satisfy a minimum pressure head of 30 m at all the nodes, and a minimum and maximum head loss gradient of 0.0005 and 0.05 respectively in each pipe, which has the effect of making pipe elimination impossible. It is assumed that all pipe Hazen-Williams coefficients are 130.

The TLN WDS design may be optimized using the multi-objective formulation (5.1), with pipe diameters as discrete variables, and with the penalty function adapted by replacing the velocity limit with lower and upper limits on head loss gradient, and excluding a maximum head constraint, but normalizing minimum head violations by a theoretical maximum of 90 m.

6.2.3 Benchmark 3 — The New York Tunnel System

The *New York Tunnel Problem* (NYTUN) was proposed by Schaake and Lai [212] in 1969. The pipe layout of the NYTUN benchmark is depicted in Figure 6.3, with pipes numbered for identification. The objective of the NYTUN problem is to determine the most economically effective design for rehabilitation of the existing system, which involves decisions on the duplication (parallelling of pipes) of the primary WDS for the city of New York. Each of twenty-one pipes that comprise the system may be considered for duplication. A single demand loading condition is available as shown in Table 6.5. Fifteen discrete diameters are available for duplicate pipes, which appear in Table 6.6, as well as the zero option of not having a pipe, yielding a decision space of size $16^{21} \approx 10^{25.28}$. The minimum head requirement at all nodes is 255 ft, apart from nodes 16, 17 and 1, for which the minimum head requirements are 260, 272.8 and 300 ft respectively. All pipes have a Hazen-Williams coefficient of 100.

NYTUN may be implemented using the standard multi-objective formulation (5.1) with twenty-one discrete variables representing the diameters of duplicate pipes (which may be assigned one of sixteen values 0–15, indexing the different pipe options, and including the zero option for no

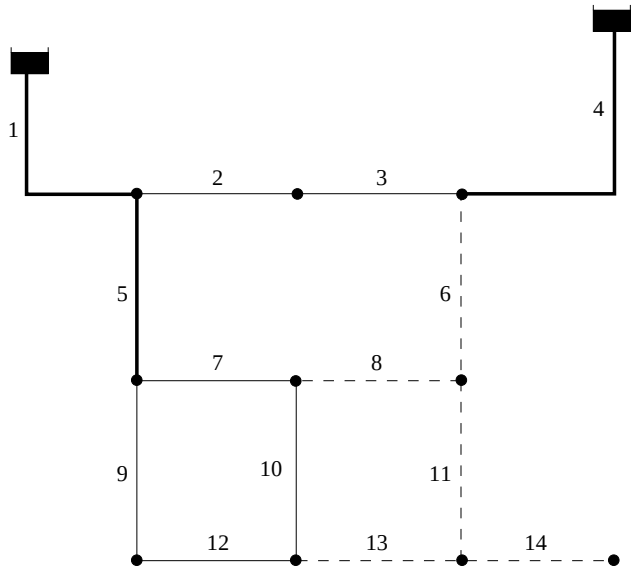


Figure 6.1: *Pipe layout for the TRP WDS benchmark [218].*

	Demand Pattern 1		Demand Pattern 2		Demand Pattern 3	
Node	Demand (l/s)	Minimum head (m)	Demand (l/s)	Minimum head (m)	Demand (l/s)	Minimum head (m)
2	12.62	28.18	12.62	14.09	12.62	14.09
3	12.62	17.61	12.62	14.09	12.62	14.09
4	0	17.61	0	14.09	0	14.09
6	18.93	35.22	18.93	14.09	18.93	14.09
7	18.93	35.22	82.03	10.57	18.93	14.09
8	18.93	35.22	18.93	14.09	18.93	14.09
9	12.62	35.22	12.62	14.09	12.62	14.09
10	18.93	35.22	18.93	14.09	18.93	14.09
11	18.93	35.22	18.93	14.09	18.93	14.09
12	12.62	35.22	12.62	14.09	50.48	10.57

Table 6.1: *Demand loading conditions for the TRP benchmark.*

Diameter (mm)	Cost of new pipe ($/m)	Cost of cleaning existing pipe ($/m)
152	49.54	47.57
203	63.32	51.51
254	94.82	55.12
305	132.87	58.07
356	170.93	60.70
407	194.88	63.00
458	232.94	–
509	264.10	–

Table 6.2: *Pipe sizing and rehabilitation options for the TRP benchmark.*

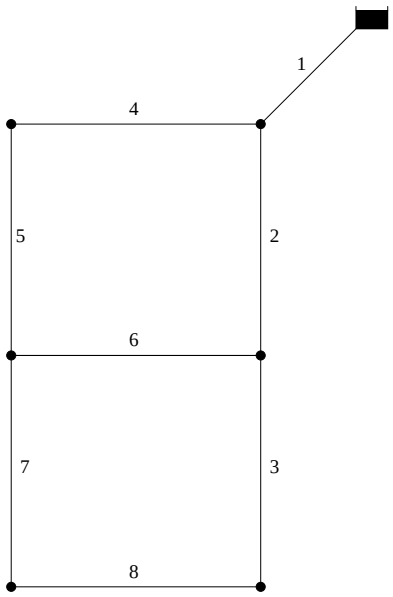


Figure 6.2*: Pipe layout for the TLN WDS benchmark [9].*

Diameter (mm)	Cost ($/m)	Diameter (mm)	Cost ($/m)
25.4	2	304.8	50.00
50.8	5	355.6	60.00
76.2	8	406.4	90.00
101.6	11	457.2	130.00
152.4	16	508.0	170.00
203.2	23	558.8	300.00
254.0	32	609.6	550.00

Table 6.3*: Pipe costs for the TLN benchmark.*

pipe). Only minimum head constraints are incorporated in the penalty function, and an upper limit of 300 ft is used to normalize the minimum head violation.

6.2.4 Benchmark 4 — The Hanoi Network

The Hanoi Network (HANOI) was first presented by Fujiwara and Khang [94] in 1990. HANOI consists of 32 nodes and 34 pipes arranged in three loops. A schematic of the HANOI pipe layout is shown in Figure 6.4. The objective of optimisation is to specify an assignment of pipe diameters in order to minimize costs, although reliability will also be considered here. Only a single fixed head source at an elevation of 100 m is available. The minimum head requirement at all nodes is 30 m. A single demand loading condition is enforced, which appears in Table 6.7. The set of commercially available diameters and associated costs appears in Table 6.8. All pipe Hazen-Williams coefficients are set at 130.

HANOI may be implemented using the standard multi-objective formulation (5.1) with thirty-four discrete variables representing pipe diameters (which may be assigned one of six values 1–6, indexing the various pipe options, or zero, representing an eliminated pipe). This yields a decision space of size $7^{34} \approx 10^{28.73}$, and there are only minimum head limits in the penalty

Node	Demand (m^3/h)
1	100
2	120
3	330
4	100
5	270
6	200

Table 6.4: *Demand loading condition for the TLN benchmark.*

function, normalized by a theoretical maximum head of 100 m.

6.2.5 Benchmark 5 — The Blacksburg Network

The *Blacksburg Network* (BLACK) was presented by Bragalli *et al.* [21] in 2008, adapted from the work of Sherali *et al.* [216]. It consists of 30 demand nodes, a single reservoir supplying a constant head of 715.56 m and 35 pipes, twelve of which have fixed diameters, leaving a total of 23 sizable pipes. A schematic of the BLACK pipe layout is shown in Figure 6.5. The objective of optimisation is to specify an assignment of pipe diameters in order to minimize costs, although reliability will also be considered here.

The minimum head requirement at all nodes is 30 m. Maximum heads are also enforced along with a single demand loading condition, which both appear in Table 6.9. The set of commercially available diameters and the associated costs appear in Table 6.10. Since 14 pipe size options are available, as well as the option of pipe elimination, the size of the decision space is $15^{23} \approx 10^{27.05}$. All pipe Hazen-Williams coefficients are set at 120. Maximum flow velocity constraints of 2 m/s are also enforced for all pipes.

BLACK may be implemented using the standard multi-objective formulation (5.1) with 23 discrete variables representing pipe diameters, and there are both minimum and maximum head limits in the penalty function, as well as maximum velocity limits.

6.2.6 Benchmark 6 — The Fossolo Network

The *Fossolo Network* (FOSS) was also presented by Bragalli *et al.* [21] in 2008, and represents a single neighbourhood of Bologna in Northern Italy. Bragalli *et al.* considered three different instances of FOSS (*foss_poly_0*, *foss_iron*, and *foss_poly_1*) which differ in available design options. Only *foss_poly_1* is considered in this dissertation, since it is the largest instance with 22 polyethylene pipe size options. A schematic of the FOSS pipe layout is shown in Figure 6.6. The objective of optimisation is to specify an assignment of pipe diameters in order to minimize costs, although reliability will also be considered here. FOSS comprises 36 demand nodes with one reservoir supplying a constant head of 121 m, and 58 pipes to be sized.

The set of commercially available diameters and the associated costs appear in Table 6.12. The minimum head requirement at all nodes is 40 m. Maximum heads are also enforced along with a single demand loading condition, which both appear in Table 6.11. The size of the decision space is $23^{58} \approx 10^{78.97}$. All pipe Hazen-Williams coefficients are set at 150. Finally, maximum flow velocity constraints of 1 m/s are enforced for all pipes.

FOSS may be implemented using the standard multi-objective formulation (5.1) with 58 discrete

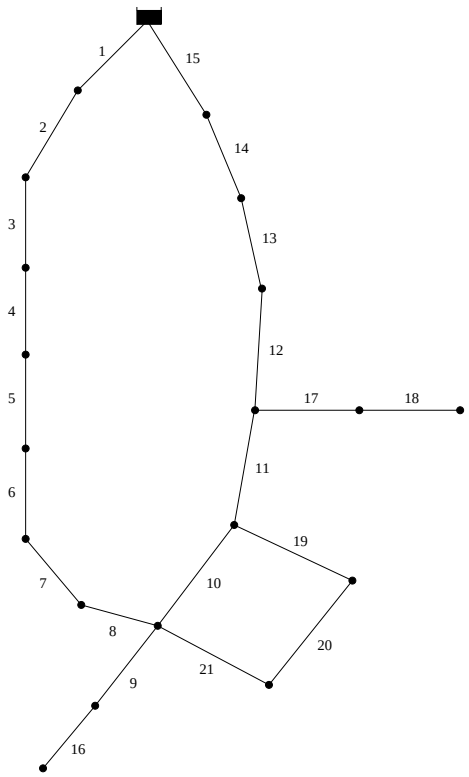


Figure 6.3*: Pipe layout for the NYTUN WDS benchmark [212].*

Node	Demand (cfs)	Node	Demand (cfs)
2	92.4	12	117.1
3	92.4	13	117.1
4	88.2	14	92.4
5	88.2	15	92.4
6	88.2	16	170.0
7	88.2	17	57.5
8	88.2	18	117.1
9	170.0	19	117.1
10	1.0	20	170.0
11	170.0	–	–

Table 6.5*: Demand loading condition for the NYTUN benchmark.*

Diameter (in)	Cost ($)	Diameter (in)	Cost ($)
36	93.5	132	469.00
48	134.0	144	522.00
60	176.0	156	577.00
72	221.0	168	632.00
84	267.0	180	689.00
96	316.0	192	746.00
108	365.0	204	804.00
120	417.0	–	–

Table 6.6*: New pipe costs for the NYTUN benchmark.*

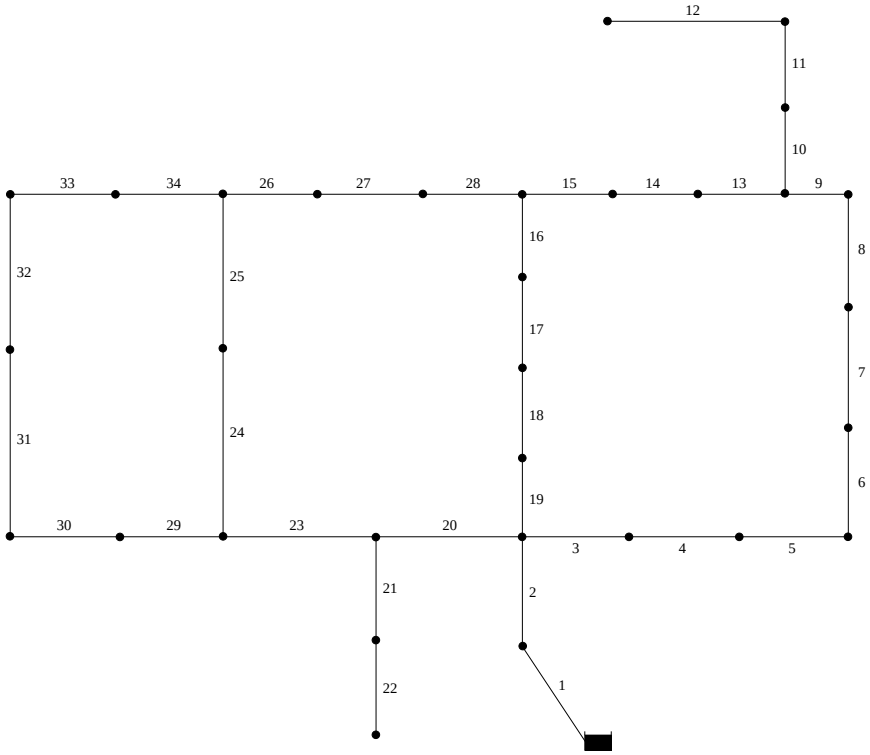


Figure 6.4*: Pipe layout for the HANOI WDS benchmark [94].*

Node	Demand (m^3/h)	Node	Demand (m^3/h)
2	890	18	1345
3	850	19	60
4	130	20	1275
5	725	21	930
6	1005	22	485
7	1350	23	1045
8	550	24	820
9	525	25	170
10	525	26	900
11	500	27	370
12	560	28	290
13	940	29	360
14	615	30	360
15	280	31	105
16	310	32	805
17	865	–	–

Table 6.7*: Demand loading condition for the HANOI benchmark.*

Diameter (mm)	Cost ($/m)
304.8	45.73
406.4	70.400
508.0	98.39
609.6	129.33
762.0	180.75
1016.0	278.28

Table 6.8*: New pipe costs for the HANOI benchmark.*

variables representing pipe diameters, and there are both minimum and maximum head limits and maximum velocity limits in the penalty function.

6.2.7 Benchmark 7 — The Pescara Network

The *Pescara Network* (PESCA) was presented by Bragalli *et al.* [21] in 2008, and represents the simplified WDS of a real Italian city. It comprises 68 demand nodes, three reservoirs supplying heads of 57.00, 53.08, and 55 m, and 99 pipes with 13 pipe size options each. A schematic of the PESCA pipe layout is shown in Figure 6.7. The objective of optimisation is to specify an assignment of pipe diameters in order to minimize cost, although reliability will also be considered here.

The minimum head requirement at all nodes is 20 m. Maximum heads are also enforced along with a single demand loading condition, which both appear in Table 6.13. The set of commercially available diameters and their associated costs appear in Table 6.14. The size of the decision space is $14^{99} \approx 10^{110}$. All pipe Hazen-Williams coefficients are set at 130. Finally, maximum flow velocity constraints of 2 m/s are enforced for all pipes.

PESCA may be implemented using the standard multi-objective formulation (5.1) with 58 discrete variables representing pipe diameters, and there are both minimum and maximum head limits and maximum velocity limits in the penalty function.

6.2.8 Benchmark 8 — The Modena Network

The *Modena Network* (MOD) is another WDS benchmark presented by Bragalli *et al.* [21] in 2008. MOD consists of 268 demand nodes and 317 pipes. A schematic of the MOD pipe layout is shown in Figure 6.8. The objective of optimisation is to specify an assignment of pipe diameters in order to minimize costs, although reliability will also be considered here. Four fixed head sources are available at elevations of 72, 73.8, 73, and 74.5 m.

The minimum head requirement at all nodes is 20 m. Maximum heads are also enforced along with a single demand loading condition, which both appear in Table D.1 (see Appendix D). The set of commercially available diameters and associated costs appears in Table 6.14. Since 13 pipe size options are available for each pipe, the size of the decision space is $13^{317} \approx 10^{354}$, which is much larger than the previous seven benchmarks. All pipe Hazen-Williams coefficients are set at 130. Velocity limits of 2 m/s are enforced for all pipes.

MOD may be implemented using the standard multi-objective formulation (5.1) with 317 discrete variables representing pipe diameters, and there are both minimum and maximum head limits as well as maximum velocity limits in the penalty function.

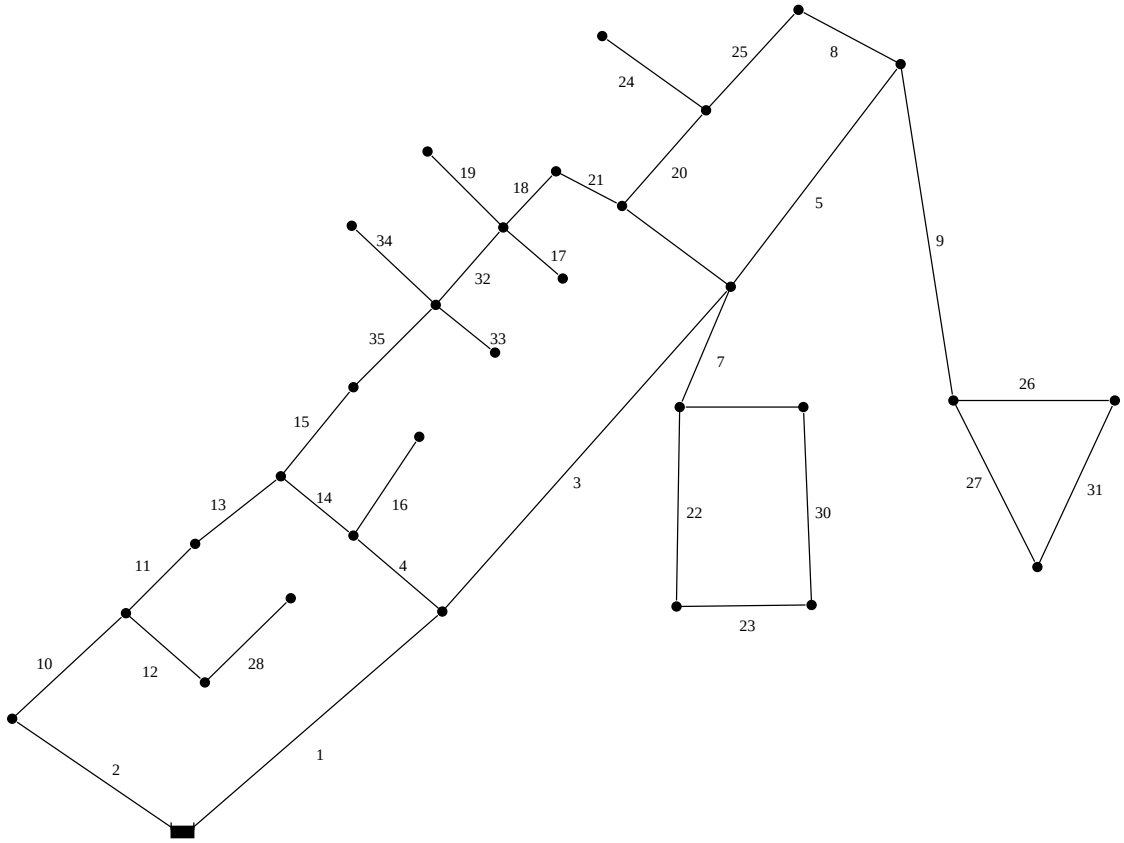


Figure 6.5*: Pipe layout for the BLACK WDS benchmark [21].*

Node	Demand (ℓps)	MaxPres m	Node	Demand (ℓps)	MaxPres m
1	0.003 287 62	62.99	16	0.003 287 62	60.55
2	0.003 191 09	65.73	17	0.000 654 87	72.74
3	0.001 625 83	69.09	18	0.006 539 28	62.08
4	0.000 873 17	59.18	19	0.003 287 62	60.40
5	0.003 384 78	62.84	20	0.000 691 47	63.29
6	0.003 263 65	66.65	21	0.003 239 67	62.84
7	0.012 654 59	67.26	22	0.000 740 05	62.08
8	0.000 716 07	67.26	23	0.003 251 66	58.27
9	0.000 679 48	72.59	24	0.006 466 73	51.71
10	0.003 287 62	69.09	25	0.003 231 66	70.00
11	0.006 357 58	63.60	26	0.003 312 23	75.64
12	0.001 710 37	72.44	27	0.003 215 07	74.88
13	0.006 357 58	64.36	28	0.001 613 84	75.94
14	0.001 625 83	62.38	29	0.002 644 73	69.39
15	0.003 251 66	61.92	30	0.003 239 67	68.48

Table 6.9*: Demand loading condition for the BLACK benchmark [21].*

Id	Diameter (mm)	Cost ($/m)	Id	Diameter (mm)	Cost ($/m)
1	0.025 4001	0.52	8	0.304 8012	75.60
2	0.050 8002	2.10	9	0.355 6014	102.90
3	0.076 2003	4.72	10	0.406 4016	134.39
4	0.101 6004	8.40	11	0.457 2018	170.09
5	0.152 4006	18.90	12	0.508 0020	209.98
6	0.203 2008	33.60	13	0.558 8022	254.08
7	0.254 0010	52.50	14	0.609 6024	302.37

Table 6.10*: New pipe costs for the BLACK benchmark.*

6.2.9 Benchmark 9 — The Exeter System

The *Exeter Network* (EXNET) benchmark is based on the real WDS of Exeter in the UK, which consists of 1 989 pipes and serves 400 000 people. EXNET has been developed by the Centre for Water Systems of Exeter University as a difficult, realistic problem. The pipe layout of EXNET is depicted in Figure 6.9.

The aim is to minimize design costs for the rehabilitation of the existing network such that projected demands are satisfied within acceptable pressure limits until the year 2020. EXNET consists of relatively small pipes and few transmission mains, such that a large head-loss range occurs at the system extremities. This makes EXNET highly sensitive to demand increases. The expansion plan is based on a calibrated model of the WDS with a minimum head requirement of 20 m at demand nodes.

The available pipe diameters are provided in Table 6.15 with associated Colebrook-White friction factors. The unit costs for pipes are a function of both pipe diameter and the type of roads. For major roads, excavations are more difficult and hence more expensive.

The full version of the EXNET problem includes tank design and extended period analysis, but the version analyzed in this dissertation has been simplified, as per *Farmani et al.* [88] by replacing two tanks with fixed head reservoirs and running simulations for a single snapshot of maximum system demand over the 24 hour period. The specification of this demand loading condition is very long, and is therefore only included on the CD accompanying this dissertation.

Rehabilitation includes possible duplication of a subset of the existing pipes (consisting of 567 pipes), with a range of ten possible pipe diameters, or a zero diameter. The size of the solution space is therefore 11^{567}, which makes EXNET a very large problem. Maximum system demand is 3245.81 ℓ/s. The two reservoirs are indicated as 3001 and 3002 in Figure 6.9, and supply water at fixed heads of 58.4 m and and 62.4 m, respectively. Nodes 3003, 3004, 3005, 3006 and 3007 supply water to EXNET from adjacent WDSs at fixed rates of 63, 1388, 10.78, 926, 26.1 ℓ/s, respectively. There are five valves in the system, three non-return valves (NRV), one pressure reducing valve (PRV) and one throttle control valve (TCV).

The EXNET benchmark may be solved via the standard multi-objective WDSDO formulation (5.1) with variables for duplicate pipes whose values may be assigned one of 11 diameter options, including the option of no pipe. Only minimum head violations are included in the penalty function, and are normalized using a theoretical maximum head of 90 m.

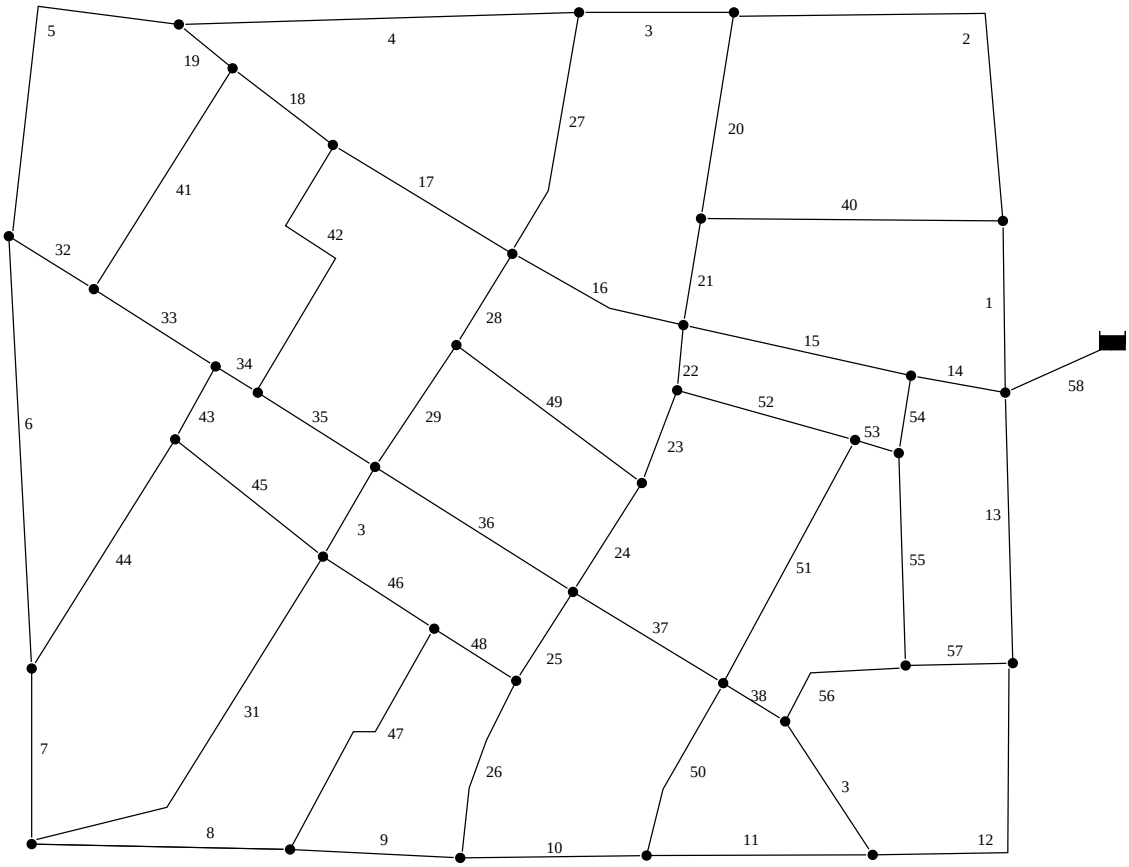


Figure 6.6*: Pipe layout for the FOSS WDS benchmark [21].*

Node	Demand (ℓps)	MaxPres m	Node	Demand (ℓps)	MaxPres m
1	0.000 49	55.85	19	0.001 88	58.1
2	0.001 04	56.6	20	0.000 93	58.17
3	0.001 02	57.65	21	0.000 96	58.2
4	0.000 81	58.5	22	0.000 97	57.1
5	0.000 63	59.76	23	0.000 86	56.8
6	0.000 79	55.6	24	0.000 67	53.5
7	0.000 26	53.1	25	0.000 77	56.6
8	0.000 58	54.5	26	0.001 69	57.6
9	0.000 54	55	27	0.001 42	57.1
10	0.001 11	56.83	28	0.000 30	55.35
11	0.001 75	57.3	29	0.000 62	56.5
12	0.000 91	58.36	30	0.000 54	56.9
13	0.001 16	59.1	31	0.000 90	56.6
14	0.000 54	58.4	32	0.001 03	56.8
15	0.001 10	57.5	33	0.000 77	56.4
16	0.001 21	56.7	34	0.000 74	56.3
17	0.001 27	55.5	35	0.001 16	55.57
18	0.002 02	56.9	36	0.000 47	55.1

Table 6.11*: Demand loading condition for the FOSS benchmark.*

Id	Diameter (mm)	Cost (/m)	Id	Diameter (mm)	Cost (/m)
1	0.016	0.38	12	0.1308	19.61
2	0.0204	0.56	13	0.1472	24.78
3	0.026	0.88	14	0.1636	30.55
4	0.0326	1.35	15	0.184	38.71
5	0.0408	2.02	16	0.2046	47.63
6	0.0514	3.21	17	0.2292	59.70
7	0.0614	4.44	18	0.2578	75.61
8	0.0736	6.45	19	0.2906	99.58
9	0.09	9.59	20	0.3274	126.48
10	0.1022	11.98	21	0.3682	160.29
11	0.1146	14.93	22	0.4092	197.71

Table 6.12*: New pipe costs for the FOSS benchmark.*

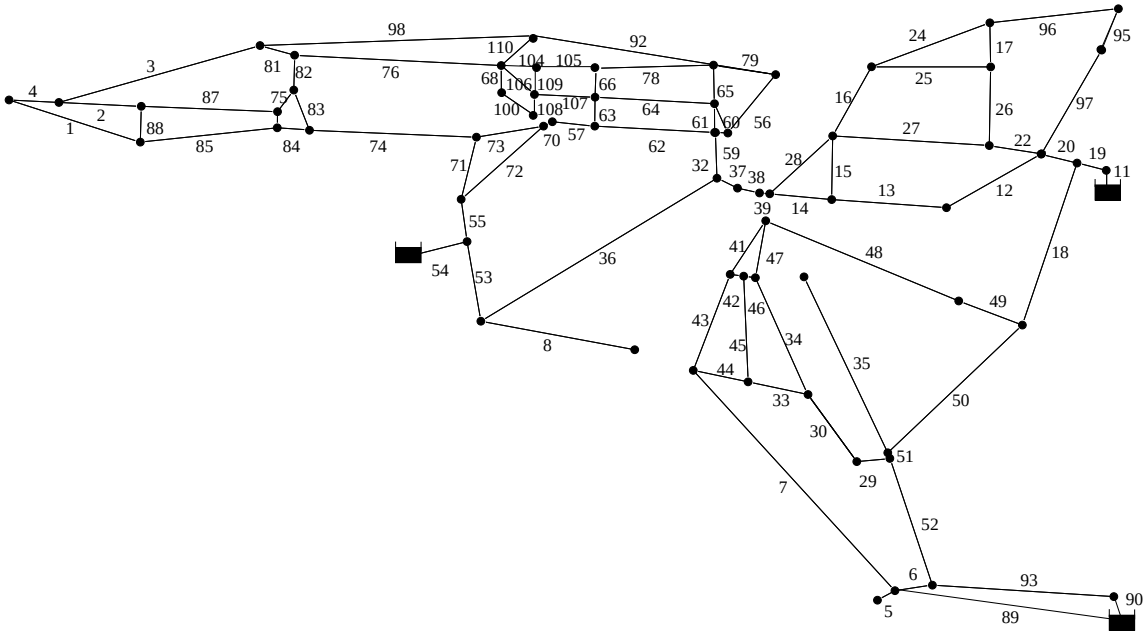


Figure 6.7*: Pipe layout for the PESCA WDS benchmark [21].*

6.3 Algorithmic Implementation

Implementation details are given with respect to the multi-objective algorithms employed for WDSDO in this dissertation. Global optimisation considerations are presented first, followed by individual meta-heuristic parameter settings.

6.3.1 Global Optimisation Considerations

All algorithms operate on the same set of initial populations, which are generated stochastically by means of LHS sampling, a quarter of which are further improved by means of the heuristic CANDA of Keedwell and Khu [142] (see Algorithm 19). This heuristic improvement was done in order to seed the initial population with some healthy alleles, but not an excessive amount (which

Node	Demand (ℓps)	MaxPres m	Node	Demand (ℓps)	MaxPres m	Node	Demand (ℓps)	MaxPres m
1	0.010 01	54.1	26	0.002 78	55.4	50	0.013 21	55.2
2	0.012 43	52	27	0.005 32	53.2	51	0.003 87	55.4
3	0.008 50	53.5	28	0.003 63	52.7	52	0.006 43	55.2
4	0.008 50	53.2	29	0.033 00	53.4	53	0.002 55	55.4
5	0.012 12	54.8	30	0.006 00	53.8	54	0	53.7
6	0.001 63	50	31	0.006 40	54.2	55	0.011 30	54.5
7	0	50.5	32	0.002 96	55.2	56	0.003 99	54.2
8	0.033 60	51.8	33	0.003 09	53.2	57	0.007 75	53.8
9	0.004 38	52.7	34	0.003 09	54.1	58	0.001 91	53.4
10	0.016 40	51.8	35	0.008 79	53.3	59	0.000 88	53.2
11	0.029 77	29	36	0.001 68	54.9	60	0.005 14	53.9
12	0.000 49	53.8	37	0.012 64	50.3	61	0.005 14	54
13	0.007 49	53.8	38	0.012 64	50.3	62	0.007 44	52.8
14	0.006 06	37.8	39	0.012 64	52.8	76	0	52.8
17	0.002 77	53	40	0.005 56	54.1	82	0.000 005	54.9
18	0.005 03	53.5	41	0.002 60	54.1	83	0	54.9
19	0.004 81	54.2	42	0.004 57	28.5	84	0.000 54	53.7
20	0.021 04	54.3	44	0.007 56	29.7	85	0.000 37	54.9
21	0.016 94	55.2	45	0.006 21	55.9	86	0.006 46	55.5
22	0.010 41	54.9	46	0.003 09	54.9	87	0.025 00	37.8
23	0.002 17	55	47	0.000 59	54.2	88	0.002 15	54.9
24	0.006 55	38.3	48	0.012 63	53.7	89	0.001 75	55.5
25	0.010 41	53.8	49	0.003 41	55.2	-	-	-

Table 6.13*: Demand loading condition for the PESCA benchmark.*

Id	Diameter (mm)	Cost (/m)
1	0.1	27.70
2	0.125	38.00
3	0.15	40.50
4	0.2	55.40
5	0.25	75.00
6	0.3	92.40
7	0.35	123.10
8	0.4	141.90
9	0.45	169.30
10	0.5	191.50
11	0.6	246.00
12	0.7	319.60
13	0.8	391.10

Table 6.14*: New pipe costs for the PESCA and MOD benchmarks.*

could otherwise lead to premature convergence). This increases the chances of finding feasible solutions for all algorithms. Thirty such populations were used for testing purposes (yielding thirty optimisation runs for each algorithm), selected as a reasonable value considering the trade-off between statistical significance and execution time. All algorithms with static populations (all algorithms excluding ADMOEA) employ a population size calculated per WDS benchmark in accordance with Goldberg's method, as explained in §5.4.6. ADMOEA begins with a similarly sized initial population.

The *replacement of duplicate solutions* in a population with randomly generated solutions has been found to improve algorithm performance. This convention of duplicate solution replace-

Diameter (mm)	Colebrook-White Friction Factor (mm)	Unit cost (£/m)	
		Minor Road / Footpath	Major Road
110	0.03	85	100
159	0.065	95	120
200	0.1	115	140
250	0.13	150	190
300	0.17	200	240
400	0.23	250	290
500	0.3	310	340
600	0.35	370	410
750	0.43	450	500
900	0.5	580	625

Table 6.15*: Pipe rehabilitation costs for the EXNET benchmark.*

ment was followed *for all algorithms in this dissertation.*

In the computation of fair trial running times, the 90^{th} percentile of longest average time to convergence amongst all algorithms was used as limit in the full time trials.

In this dissertation a WDS network is encoded as follows: pipes, valves and pumps are represented by *arrays of integer genes*, each element of which represents a sizing option for a particular component, and these values are used directly to index such options in separate tables; tanks use an *integer array of node indices* defining tank locations, and various *arrays of real-coded genes* (encoded using floating-point variables) represent tank dimensions, tank elevations and valve settings. Although the software permits pump, tank and valve design, the benchmarks analyzed only require pipe design.

Where multiple demand loadings were present, the minimum reliability surrogate value obtained for all the demand scenarios is used both as the objective to be maximized and to calculate hypervolume.

All solution quality metrics are calculated using only the feasible solutions in the final approximation sets, and were calculated in cost-reliability space.

6.3.2 Constraint Handling Schemes

The penalty factors for the penalty term constraint method were calculated as $p_{\text{f}} = C_{\max}/0.01$, where $C_{\max}$ denotes the maximum cost of a WDS design. This formula implies that a 1% or greater total normalized constraint violation will yield a penalty which is greater than the cost of the most expensive network. Such high penalty factors place substantial pressure on finding feasible solutions, which was achieved successfully for all the benchmarks in this dissertation. Since during multi-objective optimisation one is more interested in locating a trade-off set of solutions rather than solutions at the limit of feasibility, the argument that more lenient penalty factors should be employed is not particularly convincing. Penalty factors for the various benchmarks are presented in Table 6.16.

Constraints were divided into four classes: pressure head constraints, maximum velocity constraints, head loss gradient constraints and tank operational constraints. Only the first three constraint types were used in this dissertation, each normalized over their different feasible ranges. These may be sorted into their ranks in multi-dimensional constraint space for the purpose of constrained dominance testing (§5.4.4).

WDS benchmark	Penalty factor
TLN	440 000 000
TRP	286 123 962
NYTUN	29 410 320 000
HANOI	1 096 979 760
BLACK	187 010 000
FOSS	166 192 258
PESC	1 900 444 000
MOD	2 808 336 962
EXNET	9 751 805 500

Table 6.16*: Penalty factors for WDS benchmark systems.*

6.3.3 Epsilon Archiving Scheme

The $\epsilon-$archive size metric requires $\epsilon-$precision values for the different benchmarks. These values were generated by considering the represented ranges of cost and reliability in the achieved approximation sets. These values appear in Table 6.17.

WDS benchmark	ϵ_{C}	ϵ_{R}	$\epsilon_{\mathcal{E}}$
TLN	50 000	0.0005	0.01
TRP	15 000	0.002	0.01
NYTUN	1000 000	0.004	0.01
HANOI	100 000	0.002	0.005
BLACK	10 000	0.001 5	0.1
FOSS	10 000	0.001 5	0.1
PESC	75 000	0.001 5	0.1
MOD	75 000	0.002	0.1
EXNET	500 000	0.004	0.1

Table 6.17*: Epsilon precision values for cost (ϵ_{C}), surrogate reliability (ϵ_{R}), and entropy ($\epsilon_{\mathcal{E}}$), as used in the various WDS benchmarks.*

6.3.4 Hypervolume Reference Points

Reference points are required to compute the hypervolume in cost-reliability space, which is used for convergence analysis. Values of maximum cost and minimum reliability were generated for each benchmark by considering the represented ranges of cost and reliability in the achieved approximation sets. These values appear in Table 6.18.

6.3.5 Individual Algorithm Considerations

Several variations on the NSGA-II algorithm were investigated, including variants with single-point and double-point crossover and binary mutation, a variant with an adaptive SBX operator, and a jumping gene variant. However, the best performance was attained using the NSGA-II variant including the standard SBX operator (with $n_c = 2$ and a crossover probability of 0.5 for each corresponding gene pair) and TD mutation (using a component-wise mutation probability of 0.005).

Differential evolution was employed within the NSGA-II environmental selection framework (*i.e.* DE offspring do not replace their parents (target vectors), but are rejected if they are dominated

WDS benchmark	Max Cost	Min Reliability
TLN	5 000 000	0
TRP	100 000 000	0
NYTUN	300 000 000	0
HANOI	12 000 000	0
BLACK	1 500 000	0.4
FOSS	2 000 000	0.4
PESC	15 000 000	0.4
MOD	15 000 000	0
EXNET	5 000 000 000	0

Table 6.18*: Hypervolume reference points for convergence analysis, as used in the various WDS benchmarks.*

by the target vector). In a preliminary sensitivity analysis, it was found that a difference factor F of 0.7 produced the best results overall for these WDSDO studies.

The UMDA algorithm was implemented within the NSGA-II environmental selection framework. It requires no user-specified parameters. The PUMDA works similarly and recalculates the standard deviation of partition size as one third of the current reliability range.

The ANIMA and ADMOEA algorithms were implemented with the parameter settings specified in §5.7.5 and §5.7.4, respectively.

The greedy algorithm presented in Chapter 5 was implemented. The required offspring pool was filled in turn by the various heuristic steps, on condition that, in order to be accepted, a child solution may not be dominated by its parent. Finally, any duplicate solutions were replaced by stochastically generated CANDA solutions.

Multi-objective PSO was implemented as described in Chapter 5, using an inertial weight of $w = 0.75$ and learning factors $c_1 = c_2 = 2$.

The full version of SPEA-II was adapted to improve computational efficiency. This was achieved by calculating at most the first ten nearest distances to every solution, and setting solution density equal to the inverse of the distance to its 10-th nearest neighbour. In comparing solutions for truncation, only the first five nearest distances were considered, and a solution was chosen at random for deletion amongst any solutions still tied at σ^5. It is expected that this should occur relatively infrequently. An archive size equal to the population size was employed. Furthermore, distances between solutions in non-normalized objective space tend to be very large, so that their inverses are very small, which may cause rounding errors. The strategy employed in this dissertation was to take the 3-rd root of the distance. The same variational operators as used for NSGA-II were employed for SPEA-II.

It was decided to test AMALGAM with three different sets of sub-algorithms. In the first version, AMALGAM$_{\text{ndu}}$, three sub-algorithms (NSGA-II, DE & UMDA) were included, and in the second version, AMALGAM$_{\text{ndp}}$, the UMDA was replaced with the superior PUMDA algorithm. In AMALGAM$_{\text{ndug}}$, the GREEDY algorithm was additionally included. The reason for this decision is that the GREEDY algorithm functions as a powerful local search, which is capable of having a dramatic effect on search direction. A minimum of five offspring per sub-algorithm was enforced. Note that all the sub-algorithms used within the AMALGAM framework were able to conduct successful WDSDO in their own capacity. Furthermore, these algorithm combinations may also all be classed as regenerative methods (selected for ease of implementation). The TAMALGAM variants were also implemented to test the effect of incorporating the efficiency of the sub-algorithms, and the AMALGAMI and TAMALGAMJ variants were implemented to

determine whether alternative methods for quantifying offspring success could be beneficial.

6.3.6 Programming Considerations

The optimisation software was programmed in C++, using the EPANET v2.0.12 hydraulic engine (within the OOTEN Toolkit) and the *GNU Scientific Library* v1.13 for matrix operations.

When dealing with large data-structures and many iterations, as is the case with WDSDO using evolutionary optimisation, it was found to be especially important to exercise caution with regards to the computer memory environment. For example, when running optimisation trials it is not advisable to accumulate algorithmic performance data, but rather to output it after every optimisation run and compile statistics at the end. The memory environment of a computer can have a substantial influence on algorithmic performances, and should be refreshed before every new run. For similar reasons, when running comparative analysis, instead of executing all optimisation runs of each algorithm in sequence and then proceeding to the next algorithm, execute a sequence of single optimisation runs for each algorithm, which is then repeated the required number of times (*e.g.* 30). This will ensure that no algorithm is unfairly biased by natural variations in the computer memory or operating system services state.

6.4 Chapter Summary

The purpose of this chapter is to set the stage for WDSDO benchmark testing with a view to algorithmic comparison (the test results appear in Chapter 7), in partial fulfilment of dissertation Objectives 5, 6, 7 and 8 in §1.3, and to conduct probabilistic and failure reliability analysis (the subject matter of Chapter 8), in partial fulfilment of Objective 9.

A discussion on the overall benchmark testing scheme was included in §6.1, incorporating the phases of algorithmic comparison, constraint handling technique comparison, reliability surrogate measure comparison, and the illustration of design using redundant layouts.

Technical specifications of the nine WDS benchmarks from the literature used in this dissertation were presented in §6.2, and their implications with respect to the WDSDO model were highlighted. Finally, some issues pertaining to algorithmic implementation were discussed in general in §6.3, as well as for each algorithm in particular.

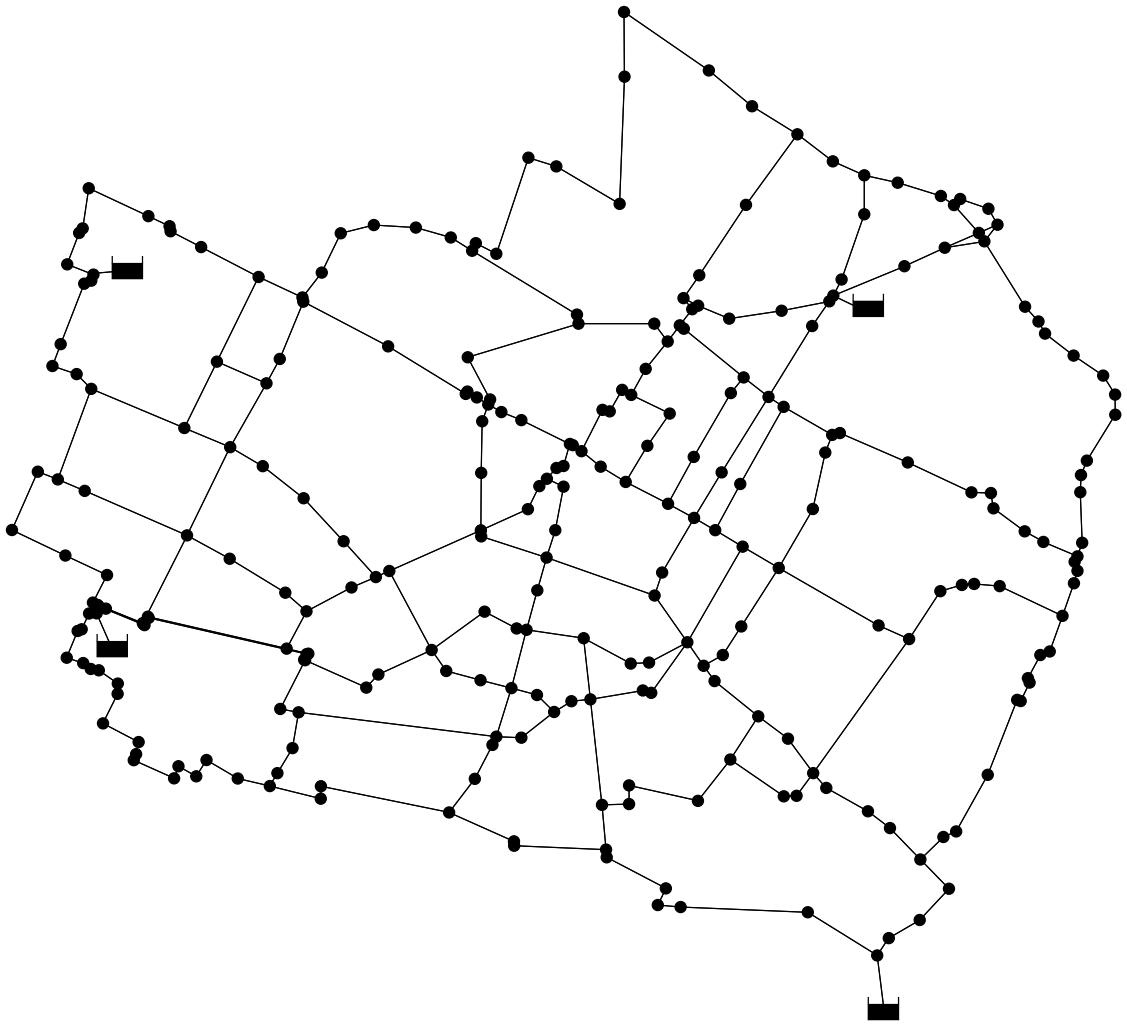


Figure 6.8: *Pipe layout for the MOD WDS benchmark [21].*

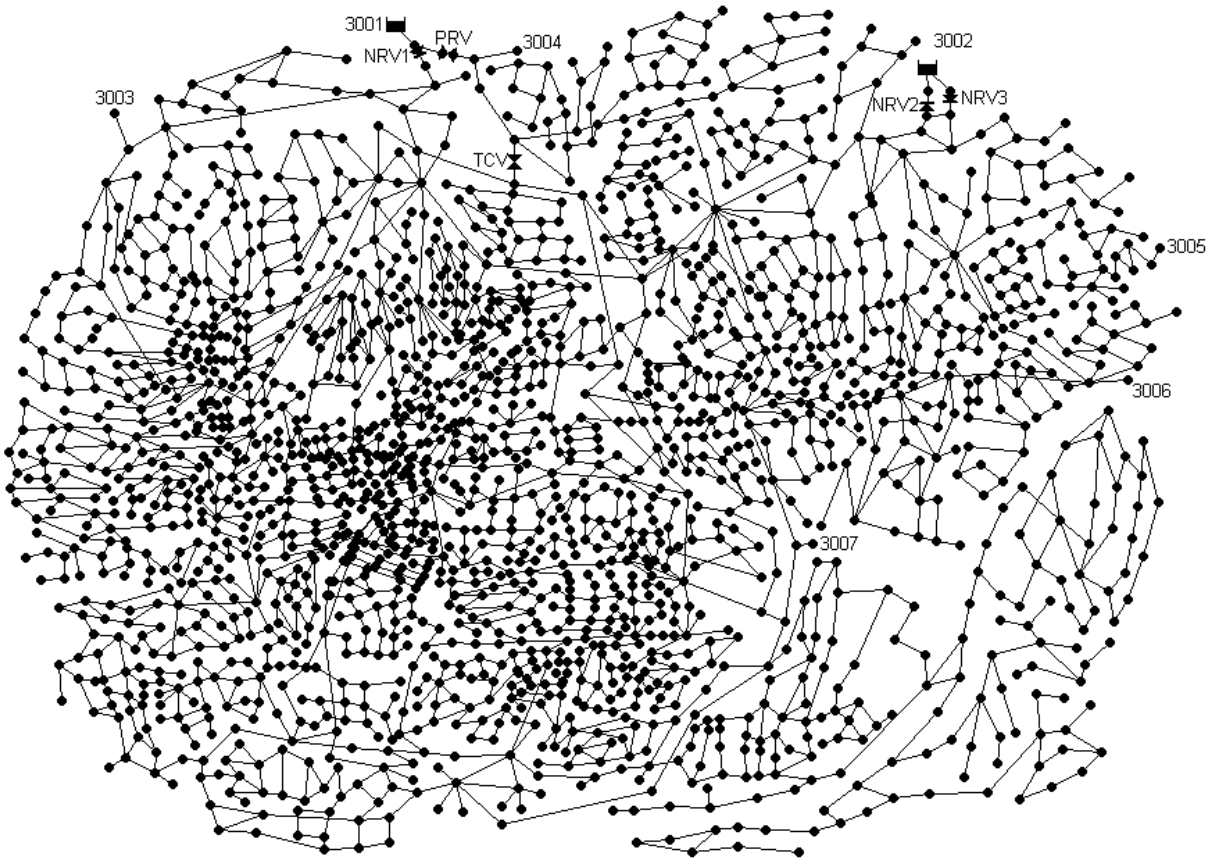


Figure 6.9*: Pipe layout for the EXNET WDS benchmark [246].*

Chapter 7

WDSDO Benchmark Tests and Results

In this chapter, the results of the WDSDO benchmark tests (Phases 1 and 2) are presented for the nine benchmarks described in Chapter 6, in order to compare algorithmic performance. All numerical computations were performed on an Intel Core 2 Duo processor (E6850 3.00GHz) with 3.25 GB of RAM, running in a Windows XP SP3 operating system. All hydraulic simulations were conducted using the OOTEN Toolkit for EPANET2, employing conventional DDA. The Network Resilience surrogate measure is employed throughout this chapter, and all optimisation is formulated as the bi-objective problem of cost minimisation and reliability maximisation, using the penalty factor method unless stated otherwise. It is important to consider both the convergence trials and the full time trials in this analysis, since in practise an algorithm's performance is reviewed independently under the condition of convergence to a stable population, but the condition of fairness in comparison additionally requires the use of equal processing times.

7.1 Phase 1 Results: WDSDO Algorithm Comparison

The first stage in this phase of analysis is population sizing, which must be conducted for each WDS benchmark. In this analysis, UMDA and PUMDA were treated separately from the other algorithms, since they typically required a population size at least double that of the other metaheuristics. Goldberg's sizing methodology (§5.4.6) was applied to these two algorithm groups, and the largest 'optimal' population size within each group was used for both convergence and time trials.

For each benchmark in this phase of testing, results from the convergence time trial are reported first, using the stopping criterion of less than 0.05% change in hypervolume for 200 consecutive generations, and conducting thirty convergence trials in order to compute average times. The 90^{th} percentile of average times to convergence amongst all algorithms is rounded to the nearest second and used as the time limit in the full length trials consisting of thirty optimisation runs. The results from these experiments are presented in tabular form for each benchmark, including averages and standard deviations of hypervolume, dominance rank, and epsilon archive size. Hypervolume is normalized by the hypervolume of the best attainment set produced after combining all algorithms' attainment sets from a particular analysis (*e.g.* a convergence analysis on a particular benchmark). This enables one to express hypervolume performance

as a percentage of best known hypervolume. Algorithms are ranked firstly on the basis of average dominance rank, and secondly using average hypervolume (in the case of identical dominance rank), then by the standard deviation of hypervolume, and finally by epsilon archive size. In the tables of results that follow, the *best unique values* of average dominance rank (Avg d_{R}), standard deviation of dominance rank (SD d_{R}), average hypervolume (Avg HV), standard deviation of hypervolume (SD HV), and average and standard deviation of time to convergence (T) in seconds are all highlighted in bold, while the *worst unique values* appear in italics. Finally, the attainment fronts (best Pareto solutions from all an algorithm's runs combined) of feasible solutions uncovered by the algorithms are presented graphically, showing the attainment fronts of the three best performing algorithms, and those of the three worst performing algorithms, superimposed on the same graph.

7.1.1 TRP Benchmark Time Trials

The optimal population size for TRP was found to be 64 for the first group of algorithms, and 128 for UMDA and PUMDA.

The convergence times of the various algorithms for the TRP benchmark are documented in Table 7.1. The 90^{th} percentile of average times to convergence is 37.67 seconds. The fastest average time to convergence of 6.07 seconds was attained by ADMOEA, and the longest average convergence time of 50.23 seconds was attained by UMDA. The group average time to convergence was 23.10 seconds, indicating that TRP is a relatively small problem. ADMOEA also demonstrated the smallest standard deviation for convergence time of 1.57 seconds, compared to the average standard deviation of 6.21 seconds. The algorithm demonstrating the best hypervolume attainment was PUMDA, which achieves an average hypervolume of 0.9939 in an average time of 38.43 seconds. The algorithm with the worst performance was GREEDY with an average dominance rank of 15.5.

In the full time trial, each algorithm was executed for thirty optimisation runs of length 38 seconds. Algorithmic performance statistics for TRP are presented in Table 7.2. The majority of the algorithms (19 out of 23) managed to obtain an average dominance rank of 1, indicating that TRP is a relatively easy problem. The best performing algorithm was PUMDA with an average hypervolume of 0.9941, followed in second place by ADMOEA with a hypervolume of 0.9919, compared to the group average hypervolume of 0.9876. The most consistent algorithm in terms of performance was SPEA-II with a hypervolume standard deviation of 0.0011. The worst performing algorithm was GREEDY with an average dominance rank of 15.5. ADMOEA produced an average ϵ-archive size of 74.43, while PUMDA and UMDA produced average ϵ-archive sizes of 121.83 and 100.47 respectively. The highest average ϵ-archive size amongst the other algorithms was 63.97, achieved by $\mathrm{AMALGAM}_{\mathrm{ndp}}$.

Plots of the attainment fronts of the three best and three worst algorithms appear in Figure 7.1, showing scarcely any difference between the attainment results of the various algorithms. It is clearly apparent that there is an exponential growth in cost as Network Resilience increases. GREEDY was the only algorithm unable to reach the low Network Resilience region of the global Pareto-front. TRP is evidently the easiest of the WDS benchmark problems.

7.1.2 TLN Benchmark Time Trials

The optimal population size for TLN was found to be 64 for the first group of algorithms, and 128 for UMDA and PUMDA.

Algorithm	**Avg d_{R}**	**SD d_{R}**	**Avg HV**	**SD HV**	**Avg T (s)**	**SD T (s)**
ADMOEA	1	0	0.9885	0.0038	**6.07**	**1.57**
$\mathrm{AMALGAM}_{\mathrm{ndp}}$	1	0	0.9896	0.0027	17.70	3.81
$\mathrm{AMALGAM}_{\mathrm{ndu}}$	1	0	0.9892	0.0020	17.00	3.34
$\mathrm{AMALGAM}_{\mathrm{ndug}}$	1	0	0.9898	0.0021	21.53	4.17
$\mathrm{AMALGAMI}_{\mathrm{ndp}}$	1	0	0.9894	0.0026	18.87	5.07
$\mathrm{AMALGAMI}_{\mathrm{ndu}}$	1	0	0.9891	0.0019	17.30	3.69
$\mathrm{AMALGAMI}_{\mathrm{ndug}}$	1	0	0.9896	0.0025	19.83	2.48
$\mathrm{AMALGAMS}_{\mathrm{ndp}}$	1	0	0.9896	0.0022	27.23	5.06
$\mathrm{AMALGAMS}_{\mathrm{ndu}}$	1	0	0.9878	0.0022	34.60	14.05
ANIMA	1	0	0.9896	0.0015	19.07	5.55
DE	1	0	0.9901	0.0017	15.47	1.98
GREEDY	*19.53*	*52.58*	*0.9574*	*0.0102*	25.00	3.04
NSGA-II	1	0	0.9895	0.0016	25.10	7.27
PSO	1.400	1.545	0.9787	0.0082	13.77	4.08
PUMDA	1	0	**0.9939**	0.0026	38.43	6.75
SPEA-II	1	0	0.9875	**0.0013**	44.10	14.97
$\mathrm{TAMALGAM}_{\mathrm{ndp}}$	1	0	0.9900	0.0024	18.57	4.27
$\mathrm{TAMALGAM}_{\mathrm{ndu}}$	1	0	0.9901	0.0021	19.40	5.08
$\mathrm{TAMALGAM}_{\mathrm{ndug}}$	1	0	0.9904	0.0020	20.53	5.28
$\mathrm{TAMALGAMJ}_{\mathrm{ndp}}$	1	0	0.9900	0.0014	20.33	6.23
$\mathrm{TAMALGAMJ}_{\mathrm{ndu}}$	1	0	0.9893	0.0019	18.77	3.82
$\mathrm{TAMALGAMJ}_{\mathrm{ndug}}$	1	0	0.9898	0.0022	22.50	6.45
UMDA	1	0	0.9695	0.0075	*50.23*	*24.88*

Table 7.1: *Time (T) to convergence (taking as stopping criterion less than 0.05% change in HV for 200 generations) for the TRP benchmark, computed over thirty optimisation runs for reference point* $(I_{\mathrm{n}}, C) = (0, 100\,000\,000)$.

Algorithm	**Rank**	**Avg d_{R}**	**SD d_{R}**	**Avg HV**	**SD HV**	**Avg A_{S}**	**SD A_{S}**
ADMOEA	3	1	0	0.9919	0.0030	74.43	3.92
$\mathrm{AMALGAM}_{\mathrm{ndp}}$	13	1	0	0.9897	0.0024	63.97	0.18
$\mathrm{AMALGAM}_{\mathrm{ndu}}$	16	1	0	0.9892	0.0019	63.73	0.52
$\mathrm{AMALGAM}_{\mathrm{ndug}}$	5	1	0	0.9902	0.0017	63.83	0.38
$\mathrm{AMALGAMI}_{\mathrm{ndp}}$	11	1	0	0.9900	0.0023	63.60	0.56
$\mathrm{AMALGAMI}_{\mathrm{ndu}}$	8	1	0	0.9901	0.0021	63.80	0.41
$\mathrm{AMALGAMI}_{\mathrm{ndug}}$	4	1	0	0.9903	0.0018	63.90	0.31
$\mathrm{AMALGAMS}_{\mathrm{ndp}}$	15	1	0	0.9896	0.0019	63.67	0.48
$\mathrm{AMALGAMS}_{\mathrm{ndu}}$	18	1	0	0.9881	0.0016	63.63	0.49
ANIMA	12	1	0	0.9898	0.0015	63.80	0.41
DE	7	1	0	0.9901	0.0017	63.87	0.43
GREEDY	*23*	*15.5*	*48.76*	*0.9612*	0.0076	63.77	0.43
NSGA-II	14	1	0	0.9896	0.0014	63.80	0.41
PSO	19	1	0	0.9833	0.0047	63.83	0.38
PUMDA	**1**	1	0	**0.9941**	0.0025	121.83	1.95
SPEA-II	17	1	0	0.9881	**0.0011**	63.80	0.48
$\mathrm{TAMALGAM}_{\mathrm{ndp}}$	22	1.10	0.31	0.9901	0.0017	63.83	0.46
$\mathrm{TAMALGAM}_{\mathrm{ndu}}$	9	1	0	0.9900	0.0018	63.83	0.38
$\mathrm{TAMALGAM}_{\mathrm{ndug}}$	2	1	0	0.9906	0.0015	63.77	0.43
$\mathrm{TAMALGAMJ}_{\mathrm{ndp}}$	21	1.10	0.31	0.9897	0.0020	63.73	0.52
$\mathrm{TAMALGAMJ}_{\mathrm{ndu}}$	10	1	0	0.9900	0.0019	63.83	0.38
$\mathrm{TAMALGAMJ}_{\mathrm{ndug}}$	6	1	0	0.9901	0.0014	63.80	0.48
UMDA	20	1.07	0.37	0.9682	*0.0077*	100.47	5.89

Table 7.2: *Mean and Standard Deviation of performance metrics for the full time trial analysis on the TRP benchmark, computed over thirty optimisation runs for reference point* $(I_{\mathrm{n}}, C) = (0, 10\,000\,000)$.

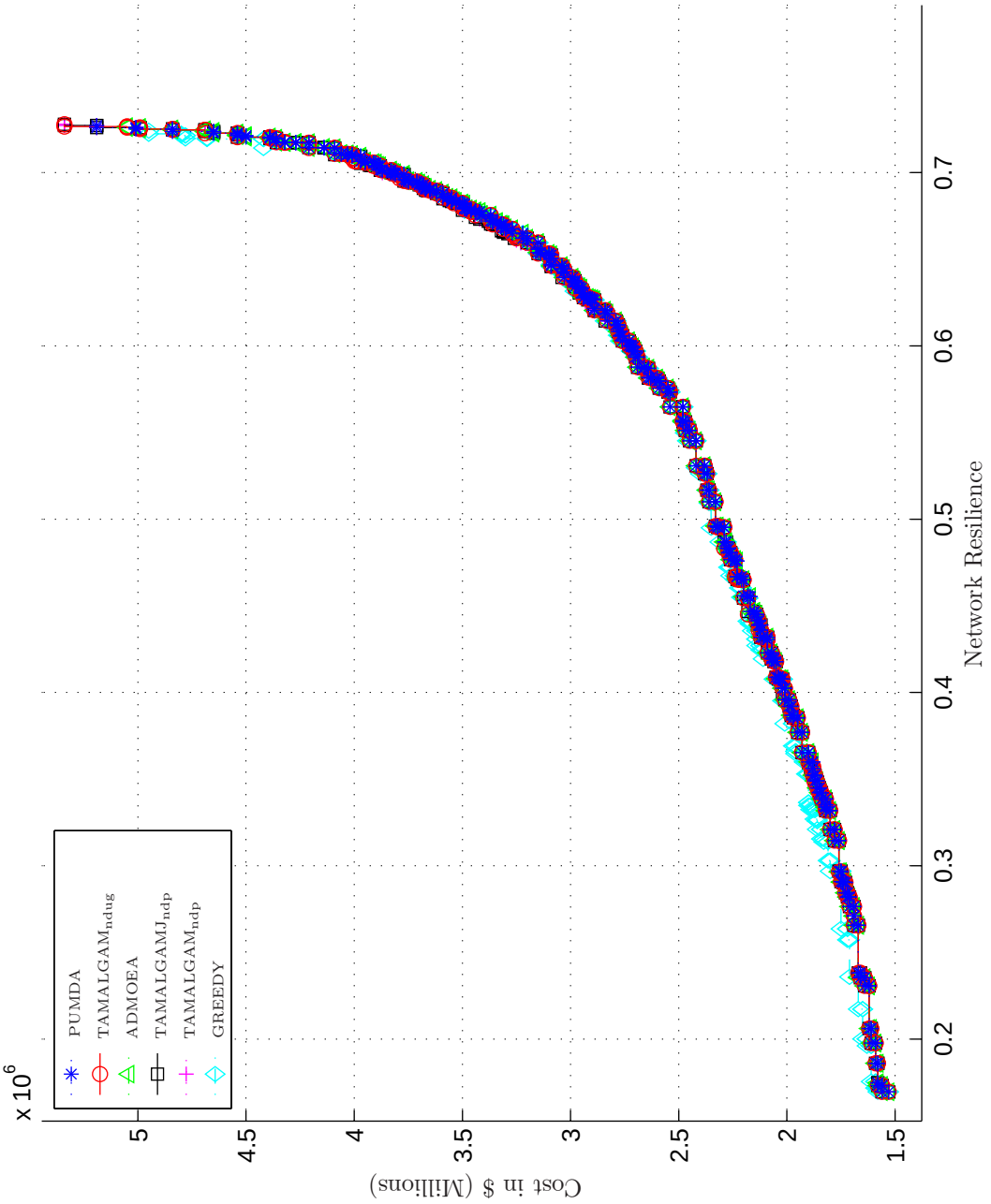


Figure 7.1: *Attainment fronts of the three best and three worst algorithms for the TRP benchmark.*

The convergence times of the various algorithms for the TLN benchmark are documented in Table 7.3. The 90^{th} percentile of average times to convergence is 22.53 seconds. The fastest average time to convergence of 3.8 seconds was attained by PUMDA, and the longest average convergence time of 37.1 seconds was attained by SPEA-II. The group average time to convergence was 13.27 seconds, also indicating that TLN is a relatively small problem. DE demonstrated the smallest standard deviation for convergence time of 1.2 seconds, compared to the average standard deviation of 4.60 seconds. The two algorithms demonstrating the best hypervolume attainment were $\text{AMALGAMS}_{\text{ndp}}$ and $\text{AMALGAMS}_{\text{ndu}}$ both with 0.9960, where the former achieved the lower standard deviation of 0.0007.

In the full time trial, each algorithm was executed for thirty optimisation runs of length 23 seconds. Algorithmic performance statistics for TLN are presented in Table 7.4. The majority of the algorithms (18 out of 23) managed to obtain an average dominance rank of 1, indicating that TLN is also a relatively easy problem. The best performing algorithm in terms of hypervolume was $\text{AMALGAMS}_{\text{ndp}}$ with an average hypervolume of 0.9965. In second place was $\text{AMALGAMS}_{\text{ndu}}$ with an average hypervolume of 0.9961. The group average hypervolume was 0.9787. These two algorithms also shared the least hypervolume standard deviation of 0.0007. The worst performing algorithm was PUMDA with a dominance rank of 271.80. ADMOEA produced an average ϵ-archive size of 37.93, while PUMDA and UMDA produced average ϵ-archive sizes of 31.83 and 31.40, respectively. The highest average ϵ-archive size amongst the other algorithms was 63.83, achieved by all three of $\text{AMALGAMI}_{\text{ndu}}$, $\text{AMALGAMS}_{\text{ndu}}$ and DE.

Plots of the attainment fronts of the three best and three worst algorithms appear in Figure 7.2, showing larger variety than for the TRP. Once again the exponential relationship between cost and reliability is revealed. Although most algorithms were able to attain the global Pareto-front, the worse performing algorithms are lagging in several regions of the objective space. Although PSO produced results that are widely spread, it forms a distinctive sub-front. The fronts produced by UMDA and PUMDA are less widely spread and break away from the Pareto-front at several places, with the former achieving greater representation in the lower Network Resilience region and the latter in the upper region.

7.1.3 NYTUN Benchmark Time Trials

The optimal population size for NYTUN was found to be 64 for all algorithms.

The convergence times of the various algorithms for the NYTUN benchmark are documented in Table 7.5. The 90^{th} percentile of average times to convergence is 60.02 seconds. The fastest average time to convergence of 10.83 seconds was attained by ADMOEA, and the longest average convergence time of 78.30 seconds was attained by $\text{AMALGAMS}_{\text{ndu}}$. The group average time to convergence was 39.76 seconds. PUMDA demonstrated the lowest standard deviation for convergence time of 2.29 seconds, compared to the average standard deviation of 12.32 seconds. The algorithm demonstrating the best hypervolume attainment was $\text{TAMALGAMJ}_{\text{ndp}}$ with a value of 0.9647. The worst performing algorithm was GREEDY with an average dominance rank of 431.47.

In the full time trial, each algorithm was executed for thirty optimisation runs of length 60 seconds. Algorithmic performance statistics for NYTUN are presented in Table 7.6. The majority of the algorithms (18 out of 23) managed to obtain an average dominance rank of 1, indicating that NYTUN is still a relatively easy problem for most of the algorithms. The best performing algorithm in terms of hypervolume was ADMOEA with an average hypervolume of 0.9732. In second place was $\text{TAMALGAMJ}_{\text{ndp}}$ with an average hypervolume of 0.9698, which

Algorithm	**Avg d_R**	**SD d_R**	**Avg HV**	**SD HV**	**Avg T (s)**	**SD T (s)**
ADMOEA	8.90	41.03	0.9808	0.0125	23.2	*32.57*
AMALGAM$_{ndp}$	1	0	0.9915	0.0047	9.5	1.2
AMALGAM$_{ndu}$	1	0	0.9953	0.0012	9.0	1.4
AMALGAM$_{ndug}$	1	0	0.9955	0.0015	11.0	2.1
AMALGAMI$_{ndp}$	1	0	0.9926	0.0027	10.0	2.2
AMALGAMI$_{ndu}$	1	0	0.9951	0.0015	9.4	1.6
AMALGAMI$_{ndug}$	1	0	0.9954	0.0010	11.4	2.4
AMALGAMS$_{ndp}$	1	0	0.9960	**0.0007**	18.6	3.9
AMALGAMS$_{ndu}$	1	0	0.9960	0.0013	20.0	3.9
ANIMA	1	0	0.9942	0.0018	10.6	2.2
DE	1	0	0.9946	0.0012	8.0	**1.2**
GREEDY	1	0	0.9165	0.0193	13.7	3.5
NSGA-II	1	0	0.9762	0.0151	14.1	5.4
PSO	59.03	13.97	0.9301	0.0087	26.4	7.2
PUMDA	314.90	*210.59*	0.9166	0.0246	**3.8**	1.8
SPEA-II	1	0	0.9833	0.0127	*37.1*	17.9
TAMALGAM$_{ndp}$	1	0	0.9911	0.0058	11.3	3.2
TAMALGAM$_{ndu}$	1	0	0.9953	0.0013	9.8	1.4
TAMALGAM$_{ndug}$	1	0	0.9956	0.0009	11.4	2.0
TAMALGAMJ$_{ndp}$	1	0	0.9894	0.0060	10.8	2.4
TAMALGAMJ$_{ndu}$	1	0	0.9951	0.0014	10.1	1.7
TAMALGAMJ$_{ndug}$	1	0	0.9955	0.0011	11.5	3.0
UMDA	*429.77*	202.60	*0.8642*	*0.0303*	4.6	1.6

Table 7.3*: Time to convergence (taking as stopping criterion less than 0.05% change in HV for 200 generations) for the TLN benchmark, computed over thirty optimisation runs for reference point* $(I_n, C) = (0, 5\,000\,000)$.

Algorithm	**Rank**	**Avg d_R**	**SD d_R**	**Avg HV**	**SD HV**	**Avg A_S**	**SD A_S**
ADMOEA	20	9.57	44.86	0.9823	0.0117	37.93	4.01
AMALGAM$_{ndp}$	13	1	0	0.9929	0.0032	63.73	0.58
AMALGAM$_{ndu}$	7	1	0	0.9954	0.0014	63.70	0.47
AMALGAM$_{ndug}$	4	1	0	0.9959	0.0010	63.50	0.68
AMALGAMI$_{ndp}$	15	1	0	0.9917	0.0039	63.73	0.45
AMALGAMI$_{ndu}$	9	1	0	0.9953	0.0012	63.83	0.38
AMALGAMI$_{ndug}$	5	1	0	0.9959	0.0012	63.53	0.82
AMALGAMS$_{ndp}$	**1**	1	0	**0.9965**	0.0007	62.50	0.78
AMALGAMS$_{ndu}$	2	1	0	0.9961	0.0007	63.83	0.46
ANIMA	12	1	0	0.9944	0.0018	63.77	0.50
DE	11	1	0	0.9948	0.0011	63.83	0.38
GREEDY	19	1.03	0.18	0.9190	0.0179	62.37	1.87
NSGA-II	17	1	0	0.9796	0.0128	63.47	0.63
PSO	21	44.30	8.36	0.9271	0.0057	40.07	4.85
PUMDA	*23*	*271.8*	178.86	0.9248	0.0189	31.83	0.38
SPEA-II	18	1	0	0.9772	0.0129	61.27	1.39
TAMALGAM$_{ndp}$	14	1	0	0.9922	0.0057	63.63	0.49
TAMALGAM$_{ndu}$	6	1	0	0.9955	0.0010	63.70	0.47
TAMALGAM$_{ndug}$	3	1	0	0.9959	0.0010	63.73	0.45
TAMALGAMJ$_{ndp}$	16	1	0	0.9888	0.0085	63.47	0.68
TAMALGAMJ$_{ndu}$	10	1	0	0.9952	0.0018	63.70	0.60
TAMALGAMJ$_{ndug}$	8	1	0	0.9954	0.0015	63.70	0.53
UMDA	22	269.70	*213.328*	*0.8884*	*0.0217*	31.40	0.89

Table 7.4*: Mean and Standard Deviation of performance metrics for the full time trial analysis on the TLN benchmark, computed over thirty optimisation runs for reference point* $(I_n, C) = (0, 5\,000\,000)$.

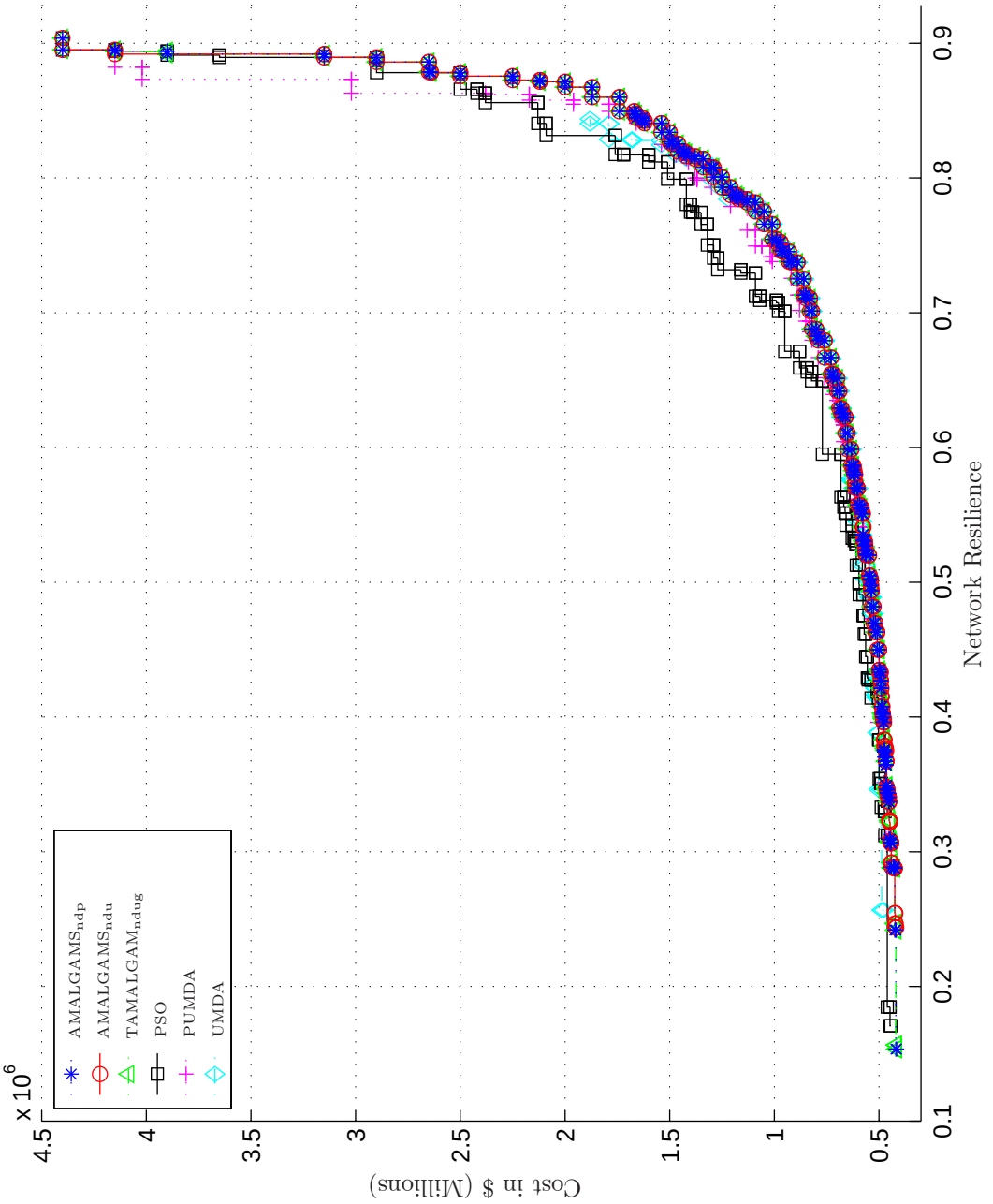


Figure 7.2: *Attainment fronts of the three best and three worst algorithms for the TLN benchmark.*

also demonstrated the smallest standard deviation for hypervolume of 0.0044. The group average hypervolume was 0.9435. The worst performing algorithm was GREEDY with a dominance rank of 482.5. ADMOEA produced an average ϵ-archive size of 126.467, and the highest average ϵ-archive size for the other algorithms was 63.43, achieved by TAMALGAMJ$_{\text{ndp}}$. UMDA and PUMDA produced ϵ-archive sizes of 57.43 and 62.33, respectively.

Plots of the attainment fronts of the three best and three worst algorithms for NYTUN appear in Figure 7.3. The ADMOEA attainment set is along the global Pareto-front, but it is less widely spread than the TAMALGAM variants which achieved additional solutions of low and high Network Resilience. The results of GREEDY and PSO form distinctive sub-fronts which meet with the global front in the low Network Resilience region of the objective space. UMDA has representation in the mid region of the objective space, but failed to locate solutions of Network Resilience greater than 0.675.

7.1.4 HANOI Benchmark Time Trials

The optimal population size for HANOI was found to be 128 for the first group of algorithms, and 64 for UMDA and PUMDA. This was the only case where the latter group assumed a smaller population size.

The convergence times of the various algorithms for the HANOI benchmark are documented in Table 7.7. The 90$^{\text{th}}$ percentile of average times to convergence is 172.51 seconds. The fastest average time to convergence of 14.5 seconds was attained by ADMOEA, and the longest average convergence time of 256.13 seconds was attained by AMALGAMS$_{\text{ndu}}$. The group average time to convergence was 107.65 seconds. PUMDA demonstrated the lowest standard deviation for convergence time of 1.21 seconds, compared to the average standard deviation of 33.56 seconds. The algorithm demonstrating the best hypervolume attainment was AMALGAM$_{\text{ndp}}$ with a value of 0.9817. This was also the algorithm with the most consistent performance, yielding a standard deviation for hypervolume of 0.0058. The worst performing algorithm was UMDA with an average dominance rank of 519.833.

In the full time trial, each algorithm was executed for thirty optimisation runs of length 173 seconds. Algorithmic performance statistics for HANOI are presented in Table 7.8. Over half of the algorithms (15 out of 23) managed to obtain an average dominance rank of 1, indicating along with convergence time that HANOI is a more difficult problem than the preceding benchmarks, but there is still not significant differentiation between the leading algorithms. The best performing algorithm in terms of hypervolume was AMALGAMS$_{\text{ndp}}$ with an average hypervolume of 0.9828, followed by TAMALGAM$_{\text{ndp}}$ in second place with a hypervolume of 0.9824. The group average hypervolume is 0.9297.

The algorithm achieving the smallest standard deviation for hypervolume of 0.0063 was AMALGAM$_{\text{ndp}}$. The worst performing algorithm was UMDA with an average dominance rank of 512.93. ADMOEA produced an average ϵ-archive size of 52.23, while the highest average ϵ-archive size was 82.80, achieved both by TAMALGAM$_{\text{ndp}}$ and TAMALGAMJ$_{\text{ndp}}$. A much larger variation of archive size was apparent for HANOI than for the previous three benchmarks.

Plots of the attainment fronts of the three best and three worst algorithms appear in Figure 7.4. The best performing algorithms were consistent at locating the global Pareto-front, showing little differentiation. The worst algorithms produced distinctive sub-fronts, revealing that they struggled to find solutions in the regions of lower to mid cost and Network Resilience, but all approached the global front in the higher Network Resilience region.

Algorithm	**Avg d_{R}**	**SD d_{R}**	**Avg HV**	**SD HV**	**Avg T (s)**	**SD T (s)**
ADMOEA	1	0	0.9636	0.0075	**10.83**	3.30
$\mathrm{AMALGAM}_{\mathrm{ndp}}$	1	0	0.9628	0.0055	33.27	12.95
$\mathrm{AMALGAM}_{\mathrm{ndu}}$	1	0	0.9545	0.0062	32.30	9.77
$\mathrm{AMALGAM}_{\mathrm{ndug}}$	1	0	0.9429	0.0064	45.03	15.14
$\mathrm{AMALGAMI}_{\mathrm{ndp}}$	1	0	0.9622	0.0065	35.13	11.72
$\mathrm{AMALGAMI}_{\mathrm{ndu}}$	1	0	0.9528	0.0058	32.33	8.86
$\mathrm{AMALGAMI}_{\mathrm{ndug}}$	1.10	0.31	0.9456	0.0058	41.97	17.09
$\mathrm{AMALGAMS}_{\mathrm{ndp}}$	1	0	0.9597	0.0060	42.53	10.31
$\mathrm{AMALGAMS}_{\mathrm{ndu}}$	1	0	0.9530	0.0059	*78.30*	*33.10*
ANIMA	1	0	0.9541	0.0069	40.07	9.22
DE	1	0	0.9388	**0.0041**	32.43	10.66
GREEDY	*431.47*	*141.70*	*0.7999*	0.0186	63.77	20.08
NSGA-II	1	0	0.9541	0.0079	40.70	12.07
PSO	24.70	25.16	0.8843	0.0135	39.20	11.94
PUMDA	8.40	14.73	0.9133	0.0166	17.17	**2.29**
SPEA-II	1	0	0.9538	0.0051	74.37	20.57
$\mathrm{TAMALGAM}_{\mathrm{ndp}}$	1	0	0.9623	0.0061	34.73	8.96
$\mathrm{TAMALGAM}_{\mathrm{ndu}}$	1	0	0.9531	0.0061	34.53	8.92
$\mathrm{TAMALGAM}_{\mathrm{ndug}}$	1	0	0.9487	0.0070	41.33	10.19
$\mathrm{TAMALGAMJ}_{\mathrm{ndp}}$	1	0	**0.9647**	0.0074	41.40	15.54
$\mathrm{TAMALGAMJ}_{\mathrm{ndu}}$	1	0	0.9536	0.0089	38.67	10.30
$\mathrm{TAMALGAMJ}_{\mathrm{ndug}}$	1.03	0.18	0.9510	0.0053	37.60	9.39
UMDA	31.57	40.10	0.8021	*0.0238*	26.90	11.10

Table 7.5*: Time (T) to convergence (taking as stopping criterion less than 0.05% change in HV for 200 generations) for the NYTUN benchmark, computed over thirty optimisation runs for reference point* $(I_{\mathrm{n}}, C) = (0, 300\,000\,000)$.

Algorithm	**Rank**	**Avg d_{R}**	**SD d_{R}**	**Avg HV**	**SD HV**	**Avg A_{S}**	**SD A_{S}**
ADMOEA	**1**	1	0	**0.9732**	0.0060	126.47	6.22
$\mathrm{AMALGAM}_{\mathrm{ndp}}$	4	1	0	0.9667	0.0060	62.77	1.07
$\mathrm{AMALGAM}_{\mathrm{ndu}}$	7	1	0	0.9658	0.0065	62.80	1.06
$\mathrm{AMALGAM}_{\mathrm{ndug}}$	14	1	0	0.9607	0.0064	62.63	1.35
$\mathrm{AMALGAMI}_{\mathrm{ndp}}$	8	1	0	0.9652	0.0056	62.73	1.34
$\mathrm{AMALGAMI}_{\mathrm{ndu}}$	5	1	0	0.9661	0.0049	62.73	1.14
$\mathrm{AMALGAMI}_{\mathrm{ndug}}$	13	1	0	0.9612	0.0063	62.87	1.17
$\mathrm{AMALGAMS}_{\mathrm{ndp}}$	19	1.03	0.18	0.9595	0.0070	63.07	0.78
$\mathrm{AMALGAMS}_{\mathrm{ndu}}$	12	1	0	0.9614	0.0070	61.33	1.52
ANIMA	16	1	0	0.9557	0.0059	62.63	1.10
DE	18	1	0	0.9383	0.0059	63.37	0.76
GREEDY	*23*	*482.5*	*118.19*	*0.7980*	*0.0158*	58.77	2.14
NSGA-II	15	1	0	0.9569	0.0052	62.37	1.16
PSO	21	40.63	55.12	0.8881	0.0124	63.03	1.22
PUMDA	20	12.17	16.41	0.9180	0.0135	62.33	1.30
SPEA-II	17	1	0	0.9522	0.0065	60.23	1.36
$\mathrm{TAMALGAM}_{\mathrm{ndp}}$	3	1	0	0.9691	0.0054	63.33	0.84
$\mathrm{TAMALGAM}_{\mathrm{ndu}}$	6	1	0	0.9658	0.0052	62.97	0.76
$\mathrm{TAMALGAM}_{\mathrm{ndug}}$	11	1	0	0.9631	0.0057	62.93	1.23
$\mathrm{TAMALGAMJ}_{\mathrm{ndp}}$	2	1	0	0.9698	**0.0044**	63.43	0.77
$\mathrm{TAMALGAMJ}_{\mathrm{ndu}}$	9	1	0	0.9648	0.0062	62.53	0.90
$\mathrm{TAMALGAMJ}_{\mathrm{ndug}}$	10	1	0	0.9641	0.0064	62.30	1.73
UMDA	22	43.63	44.09	0.8176	0.0151	57.43	2.43

Table 7.6*: Mean and Standard Deviation of performance metrics for the full time trial analysis on the NYTUN benchmark, computed over thirty optimisation runs for reference point* $(I_{\mathrm{n}}, C) = (0, 300\,000\,000)$.

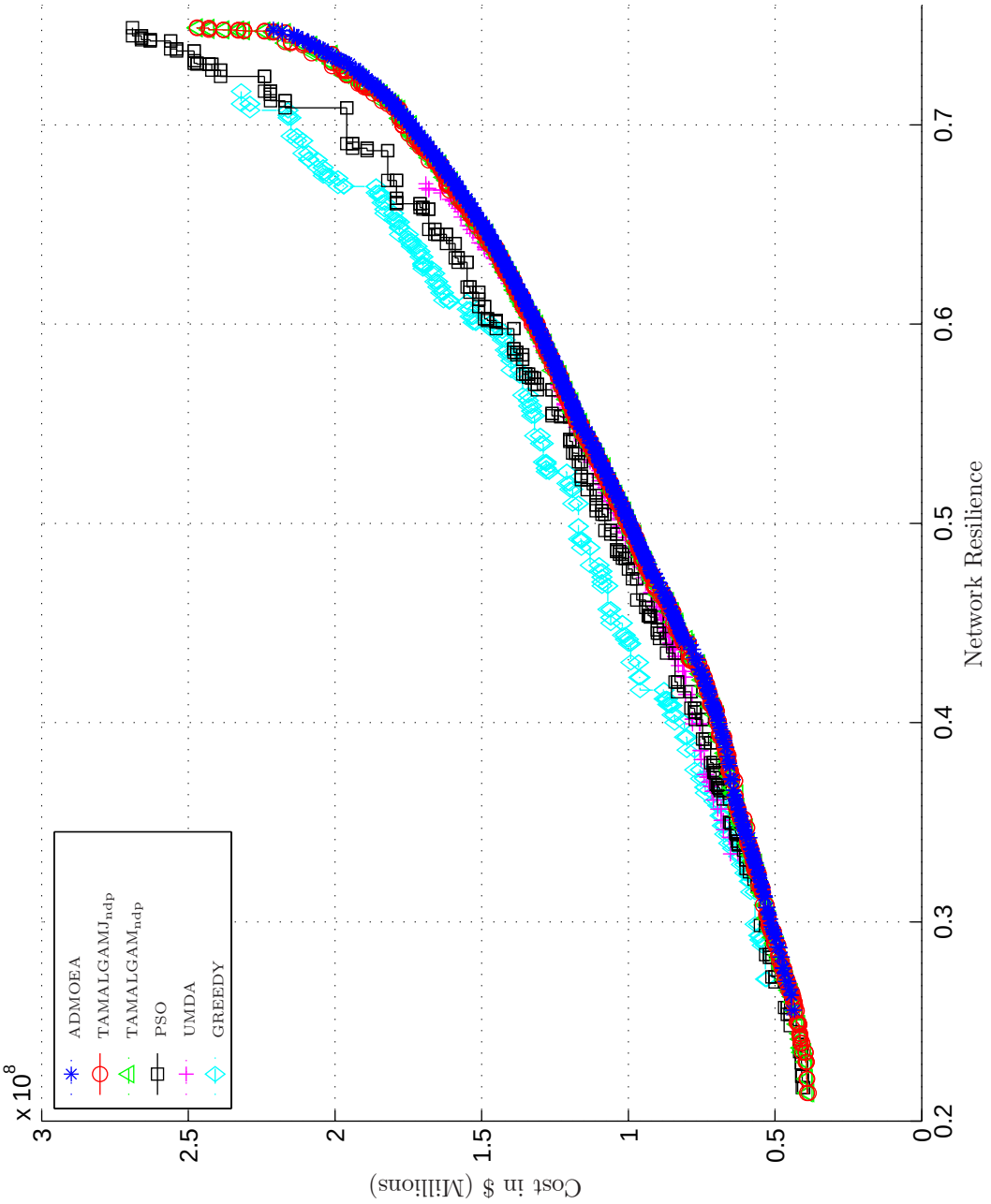


Figure 7.3: *Attainment fronts of the three best and three worst algorithms for the NYTUN benchmark.*

Algorithm	**Avg d_R**	**SD d_R**	**Avg HV**	**SD HV**	**Avg T (s)**	**SD T (s)**
ADMOEA	49.00	102.86	0.9165	0.0323	**14.5**	8.47
AMALGAM$_{ndp}$	1	0	**0.9817**	**0.0058**	86.10	21.64
AMALGAM$_{ndu}$	1	0	0.9670	0.0125	107.87	31.56
AMALGAM$_{ndug}$	1	0	0.9599	0.0162	119.67	37.05
AMALGAMI$_{ndp}$	1	0	0.9776	0.0099	77.67	25.62
AMALGAMI$_{ndu}$	1	0	0.9649	0.0126	102.13	29.06
AMALGAMI$_{ndug}$	1	0	0.9617	0.0112	120.73	50.26
AMALGAMS$_{ndp}$	1	0	0.9699	0.0117	184.07	94.53
AMALGAMS$_{ndu}$	1	0	0.9658	0.0149	*256.13*	77.50
ANIMA	1	0	0.9683	0.0143	126.27	40.48
DE	10.67	28.90	0.9409	0.0247	104.47	35.13
GREEDY	424.13	98.46	0.8728	0.0230	97.37	20.06
NSGA-II	1	0	0.9735	0.0102	109.10	27.34
PSO	381.53	59.40	0.7066	0.0328	117.53	33.41
PUMDA	387.37	*131.581*	0.7282	0.0531	14.83	**1.21**
SPEA-II	1	0	0.9668	0.0111	233.83	68.83
TAMALGAM$_{ndp}$	1	0	0.9800	0.0081	84.53	23.57
TAMALGAM$_{ndu}$	1	0	0.9688	0.0115	109.10	25.79
TAMALGAM$_{ndug}$	1	0	0.9664	0.0104	104.03	34.03
TAMALGAMJ$_{ndp}$	1	0	0.9802	0.0086	89.67	26.49
TAMALGAMJ$_{ndu}$	1	0	0.9725	0.0103	96.23	27.02
TAMALGAMJ$_{ndug}$	1	0	0.9696	0.0107	101.37	26.75
UMDA	*519.833*	88.40	*0.6166*	*0.0620*	18.80	6.12

Table 7.7*: Time (T) to convergence (taking as stopping criterion less than 0.05% change in HV for 200 generations) for the HANOI benchmark, computed over thirty optimisation runs for reference point* $(I_n, C) = (0, 12\,000\,000)$.

Algorithm	**Rank**	**Avg d_R**	**SD d_R**	**Avg HV**	**SD HV**	**Avg A_S**	**SD A_S**
ADMOEA	19	3.40	7.37	0.9174	0.0289	52.23	6.17
AMALGAM$_{ndp}$	2	1	0	0.9820	**0.0063**	78.43	4.81
AMALGAM$_{ndu}$	10	1	0	0.9708	0.0101	74.93	3.93
AMALGAM$_{ndug}$	13	1	0	0.9659	0.0131	74.37	4.73
AMALGAMI$_{ndp}$	3	1	0	0.9818	0.0076	77.20	5.25
AMALGAMI$_{ndu}$	11	1	0	0.9691	0.0107	75.03	4.54
AMALGAMI$_{ndug}$	15	1	0	0.9641	0.0095	72.20	3.13
AMALGAMS$_{ndp}$	**1**	1	0	**0.9828**	0.0077	78.83	4.28
AMALGAMS$_{ndu}$	12	1	0	0.9681	0.0122	74.00	4.56
ANIMA	8	1	0	0.9722	0.0107	75.10	4.78
DE	20	7.23	20.98	0.9453	0.0248	74.40	4.66
GREEDY	22	436.17	100.36	0.8744	0.0232	67.07	4.27
NSGA-II	7	1	0	0.9754	0.0093	75.83	3.71
PSO	17	376	68.10	0.7186	0.0301	49.30	4.80
PUMDA	21	326.33	*197.70*	0.7382	0.0560	50.20	3.92
SPEA-II	14	1	0	0.9649	0.0098	75.80	4.73
TAMALGAM$_{ndp}$	18	1.07	0.25	0.9824	0.0080	82.80	3.25
TAMALGAM$_{ndu}$	6	1	0	0.9755	0.0119	77.37	5.56
TAMALGAM$_{ndug}$	9	1	0	0.9715	0.0118	76.07	3.64
TAMALGAMJ$_{ndp}$	16	1.03	0.18	0.9825	0.0080	82.80	2.95
TAMALGAMJ$_{ndu}$	4	1	0	0.9783	0.0085	77.37	3.90
TAMALGAMJ$_{ndug}$	5	1	0	0.9756	0.0077	76.10	4.02
UMDA	*23*	*512.93*	119.62	*0.6268*	*0.0639*	34.50	4.24

Table 7.8*: Mean and Standard Deviation of performance metrics for the full time trial analysis on the HANOI benchmark, computed over thirty optimisation runs for reference point* $(I_n, C) = (0, 12\,000\,000)$.

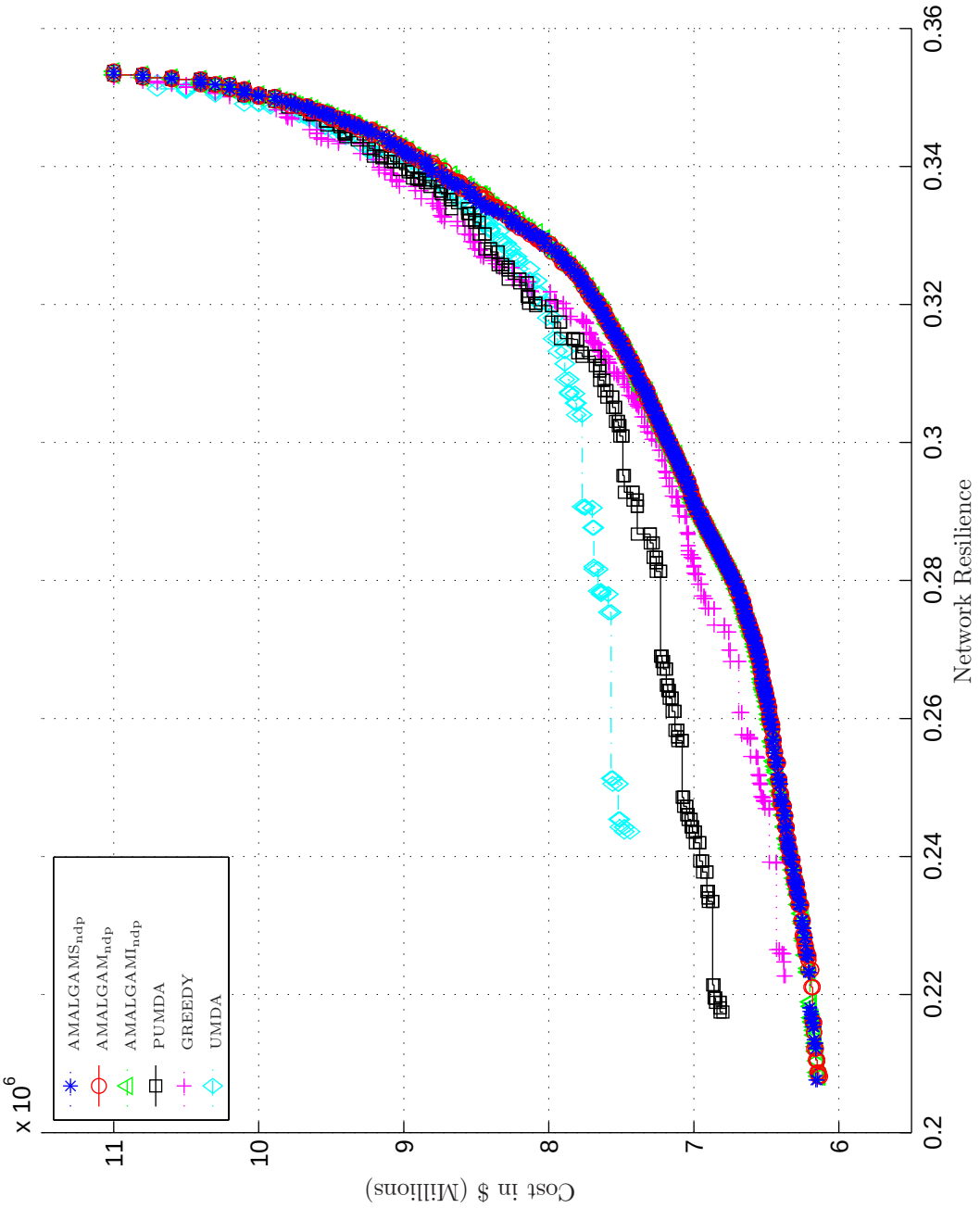


Figure 7.4: *Attainment fronts of the three best and three worst algorithms for the HANOI benchmark.*

7.1.5 BLACK Benchmark Time Trials

The optimal population size for BLACK was found to be 64 for the first group of algorithms, and 256 for UMDA and PUMDA.

The convergence times of the various algorithms for the BLACK benchmark are documented in Table 7.9. The 90^{th} percentile of average times to convergence is 79.55 seconds. The fastest average time to convergence of 16.67 seconds was attained by ADMOEA, and the longest average convergence time of 125.97 seconds was attained by UMDA. The group average time to convergence was 46.84 seconds. ADMOEA also demonstrated the lowest standard deviation for convergence time of 3.74 seconds, compared to the average standard deviation of 13.64 seconds. The algorithm demonstrating the best hypervolume attainment was SPEA-II with a value of 0.9783. The worst performing algorithm was PSO with an average dominance rank of 620.4.

In the full time trial, each algorithm was executed for thirty optimisation runs of length 80 seconds. Algorithmic performance statistics for BLACK are presented in Table 7.10. Over half of the algorithms (14 out of 23) managed to obtain an average dominance rank of 1. The best performing algorithm in terms of average dominance rank and hypervolume was $\text{TAMALGAMJ}_{\text{ndu}}$ with a dominance rank of 1 and a hypervolume of 0.9798. Although $\text{AMALGAMS}_{\text{ndp}}$ achieved the highest average hypervolume of 0.9805, it trailed with a dominance rank of 1.03, while second place goes to $\text{AMALGAMI}_{\text{ndu}}$ with an average hypervolume of 0.9793. The group average hypervolume was 0.9631. The algorithm with the smallest standard deviation in hypervolume was $\text{AMALGAMS}_{\text{ndu}}$ with an average value of 0.0043. The worst performing algorithm was PSO with a dominance rank of 616.8. ADMOEA produced an average ϵ-archive size of 76.6. PUMDA and UMDA produced ϵ-archive sizes of 213.83 and 130.63 respectively, despite having identical population sizes. The largest ϵ-archive size amongst the other algorithms was obtained by $\text{AMALGAMI}_{\text{ndp}}$ and is 63.80.

Plots of the attainment fronts of the three best and three worst algorithms appear in Figure 7.5. The best algorithms again had little trouble locating the global Pareto-front, and show little differentiation. The results of PSO and GREEDY form distinctive sub-fronts which merge with the global front towards the lower Network Resilience region of the objective space. UMDA produced results that approach the front quite closely, but are localised to the central region.

7.1.6 FOSS Benchmark Time Trials

The optimal population size for FOSS was found to be 64 for the first group of algorithms, and 256 for UMDA and PUMDA.

The convergence times of the various algorithms for the FOSS benchmark are documented in Table 7.11. The 90^{th} percentile of average times to convergence is 119.24 seconds. The fastest average time to convergence of 52.5 seconds was attained by NSGA-II, and the longest average convergence time of 141.97 seconds was attained by UMDA. The group average time to convergence was 86.50 seconds. ADMOEA also demonstrated the lowest standard deviation for convergence time of 16.08 seconds, compared to the average standard deviation of 29.53 seconds. The algorithm demonstrating the best average dominance rank and hypervolume attainment was DE with values of 1.23 and 0.9457, respectively. The worst performing algorithm was GREEDY with an average dominance rank of 628.4.

In the full time trial, each algorithm was executed for thirty optimisation runs of length 119 seconds. Algorithmic performance statistics for FOSS are presented in Table 7.12. FOSS presents a greater challenge than the previous benchmarks, with only three of the algorithms

Algorithm	**Avg d_R**	**SD d_R**	**Avg HV**	**SD HV**	**Avg T (s)**	**SD T (s)**
ADMOEA	9.77	26.88	0.9607	0.0105	**16.67**	**3.74**
AMALGAM$_{ndp}$	1.07	0.37	0.9710	0.0092	29.87	9.75
AMALGAM$_{ndu}$	1	0	0.9721	0.0079	34.07	11.40
AMALGAM$_{ndug}$	1.07	0.37	0.9697	0.0058	35.87	11.94
AMALGAMI$_{ndp}$	1.17	0.75	0.9708	0.0068	30.03	10.92
AMALGAMI$_{ndu}$	1	0	0.9726	0.0081	32.83	11.13
AMALGAMI$_{ndug}$	1.07	0.25	0.9685	0.0078	37.63	10.54
AMALGAMS$_{ndp}$	1	0	0.9780	0.0061	64.67	17.29
AMALGAMS$_{ndu}$	1.13	0.35	0.9780	**0.0053**	83.27	19.26
ANIMA	3.80	9.74	0.9646	0.0088	34.57	13.84
DE	1.03	0.18	0.9686	0.0072	41.77	10.60
GREEDY	407.27	*131.61*	0.8800	*0.0244*	45.47	9.34
NSGA-II	1.23	0.94	0.9688	0.0112	30.27	12.00
PSO	*620.4*	16.07	*0.8342*	0.0156	61.63	22.75
PUMDA	4.83	10.00	0.9733	0.0089	111.73	15.91
SPEA-II	3.07	6.52	**0.9783**	0.0065	63.47	13.43
TAMALGAM$_{ndp}$	1.10	0.40	0.9699	0.0071	25.13	7.00
TAMALGAM$_{ndu}$	1	0	0.9744	0.0066	38.97	11.89
TAMALGAM$_{ndug}$	1	0	0.9737	0.0075	36.00	12.54
TAMALGAMJ$_{ndp}$	4.17	14.81	0.9679	0.0073	30.00	11.98
TAMALGAMJ$_{ndu}$	1.07	0.37	0.9715	0.0078	34.00	11.08
TAMALGAMJ$_{ndug}$	1	0	0.9706	0.0095	33.53	10.06
UMDA	155.87	109.18	0.9027	0.0091	*125.97*	*45.32*

Table 7.9*: Time (T) to convergence (taking as stopping criterion less than 0.05% change in HV for 200 generations) for the BLACK benchmark, computed over thirty optimisation runs for reference point* $(I_n, C) = (0.4, 1\,500\,000)$.

Algorithm	**Rank**	**Avg d_R**	**SD d_R**	**Avg HV**	**SD HV**	**Avg A_S**	**SD A_S**
ADMOEA	19	4.77	12.20	0.9716	0.0106	76.60	6.04
AMALGAM$_{ndp}$	11	1	0	0.9750	0.0105	63.60	0.67
AMALGAM$_{ndu}$	8	1	0	0.9776	0.0058	63.63	0.56
AMALGAM$_{ndug}$	14	1	0	0.9719	0.0067	63.43	0.77
AMALGAMI$_{ndp}$	7	1	0	0.9778	0.0057	63.80	0.41
AMALGAMI$_{ndu}$	2	1	0	0.9793	0.0061	63.67	0.61
AMALGAMI$_{ndug}$	13	1	0	0.9719	0.0063	63.63	0.72
AMALGAMS$_{ndp}$	15	1.03	0.18	**0.9805**	0.0050	63.00	1.23
AMALGAMS$_{ndu}$	17	1.63	1.43	0.9782	**0.0043**	62.77	1.10
ANIMA	16	1.33	1.65	0.9730	0.0084	63.70	0.47
DE	12	1	0	0.9743	0.0046	63.70	0.47
GREEDY	22	448.97	81.22	0.8806	*0.0228*	63.77	0.50
NSGA-II	5	1	0	0.9782	0.0072	63.57	0.68
PSO	*23*	*616.8*	12.21	*0.8404*	0.0136	26.40	6.13
PUMDA	20	7.67	13.46	0.9729	0.0090	213.83	6.94
SPEA-II	18	2.93	5.05	0.9796	0.0057	63.43	0.77
TAMALGAM$_{ndp}$	10	1	0	0.9754	0.0053	63.47	0.57
TAMALGAM$_{ndu}$	3	1	0	0.9788	0.0059	63.67	0.55
TAMALGAM$_{ndug}$	6	1	0	0.9781	0.0052	63.37	0.67
TAMALGAMJ$_{ndp}$	4	1	0	0.9785	0.0048	63.47	0.82
TAMALGAMJ$_{ndu}$	**1**	1	0	0.9798	0.0058	63.43	0.82
TAMALGAMJ$_{ndug}$	9	1	0	0.9773	0.0067	63.37	0.72
UMDA	21	157.37	*104.82*	0.9017	0.0093	130.63	17.19

Table 7.10*: Mean and Standard Deviation of performance metrics for the full time trial analysis on the BLACK benchmark, computed over thirty optimisation runs for reference point* $(I_n, C) = (0.4, 1\,500\,000)$.

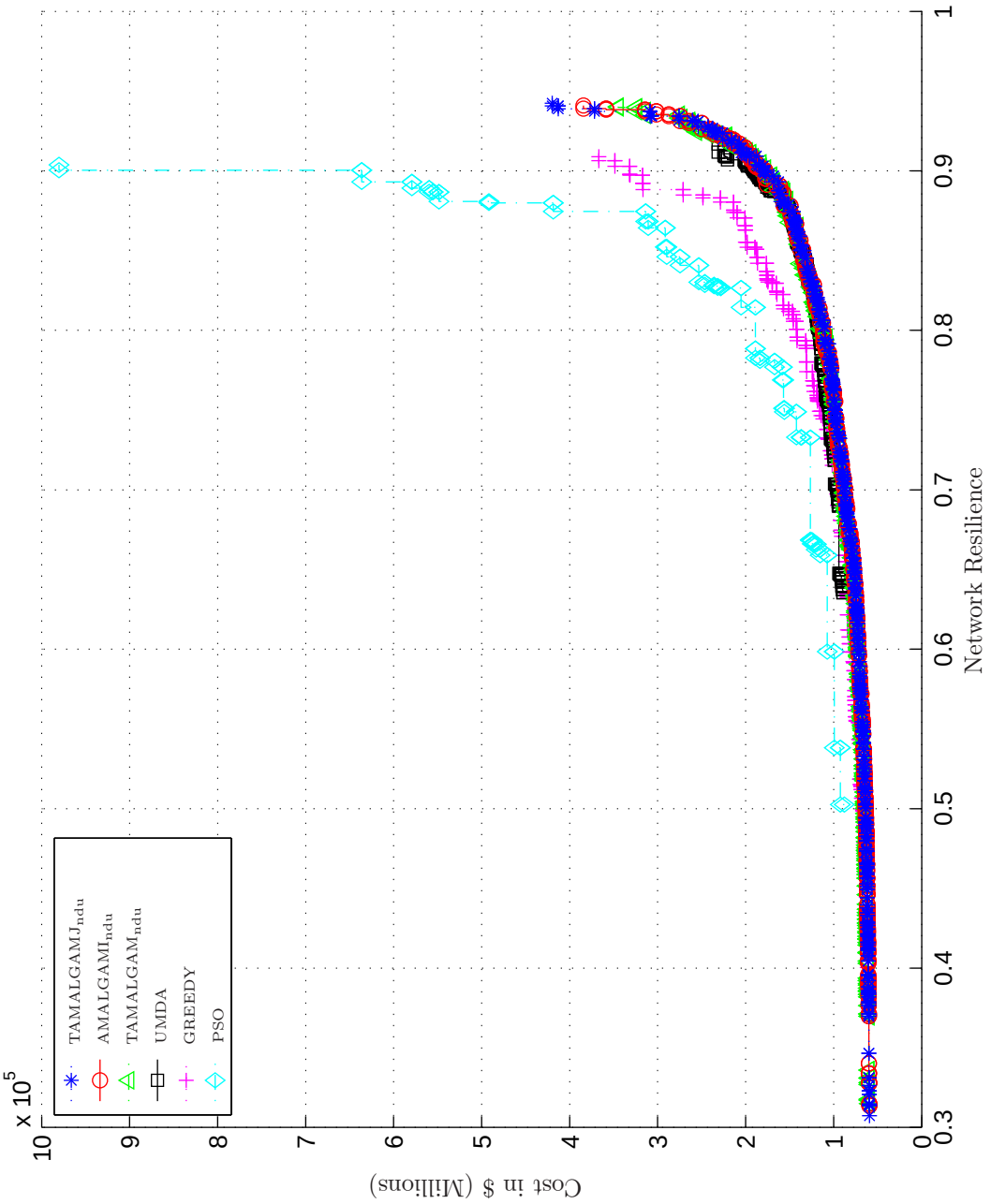


Figure 7.5: *Attainment fronts of the three best and three worst algorithms for the BLACK benchmark.*

achieving an average dominance rank of 1. A further ten algorithms managed to obtain a dominance rank between 1 and 2. The best performing algorithm by a long margin was DE with an average hypervolume of 0.9459, trailed in second place by $\text{AMALGAMI}_{\text{ndu}}$ which, although it has a dominance rank of 1, only managed an average hypervolume of 0.8437. The group average hypervolume is 0.8141. The algorithm with the smallest standard deviation in hypervolume was PUMDA with a value of 0.0137. The worst performing algorithm was GREEDY with a dominance rank of 629.8 and an average hypervolume of 0.5415. ADMOEA produced an average ϵ-archive size of 65.06, while PUMDA and UMDA produced ϵ-archive sizes of 184.27 and 47.9, respectively. The largest ϵ-archive size amongst the other algorithms, of 63.07, was obtained by DE.

Plots of the attainment fronts of the three best and three worst algorithms appear in Figure 7.6, showing the most interesting results thus far. There is clear differentiation between the attainment results of the best algorithms, uncovering unique parts of the global Pareto-front. The AMALGAM variants were better at finding solutions in the lower Network Resilience region, up to approximately 0.915, whereafter DE ventured off on its own to uniquely locate some solutions of very high cost and Network Resilience. For the first time completely disjoint sub-fronts generated by PSO and GREEDY are clearly visible. The results of UMDA are once again located in the central region, where the Pareto-front curvature and solution cost-benefit is highest.

7.1.7 PESC Benchmark Time Trials

The optimal population size for PESC was found to be 64 for the first group of algorithms, and 256 for UMDA and PUMDA.

The convergence times of the various algorithms for the PESC benchmark are documented in Table 7.13. The 90^{th} percentile of average times to convergence was 241.97 seconds. The fastest average time to convergence of 67.33 seconds was attained by ADMOEA, and the longest average convergence time of 281.63 seconds was attained by $\text{AMALGAMS}_{\text{ndu}}$. The group average time to convergence was 161.66 seconds. PUMDA demonstrated the lowest standard deviation for convergence time of 4.50 seconds, compared to the average standard deviation of 47.02 seconds. The algorithm achieving the best average dominance rank was $\text{AMALGAMS}_{\text{ndu}}$ with a value of 1.5. The algorithm demonstrating the best hypervolume attainment was DE with a value of 0.9261. The worst performing algorithm was GREEDY with an average dominance rank of 632.67.

In the full time trial, each algorithm was executed for thirty optimisation runs of length 242 seconds. Algorithmic performance statistics for PESC are presented in Table 7.14. PESC seems to be an even more difficult a problem than FOSS, with only two of the algorithms achieving an average dominance rank of 1, and a further ten algorithms managing to obtain a dominance rank of between 1 and 2. The best performing algorithm in terms of dominance rank was NSGA-II, with an average hypervolume of 0.8974, followed in second place by $\text{TAMALGAMJ}_{\text{ndu}}$ which managed an average hypervolume of 0.8847. The algorithm achieving the highest average hypervolume was SPEA-II, with a value of 0.9194, although it only managed a dominance rank of 1.3. The group average hypervolume was 0.8539. The algorithm with the smallest standard deviation in hypervolume was $\text{AMALGAMS}_{\text{ndu}}$ with a value of 0.0080. The worst performing algorithm was GREEDY with a dominance rank of 633.07 and an average hypervolume of 0.5823. ADMOEA produced an average ϵ-archive size of 84.73. PUMDA and UMDA produced ϵ-archive sizes of 201.33 and 38.86, respectively.

Algorithm	**Avg d_{R}**	**SD d_{R}**	**Avg HV**	**SD HV**	**Avg T (s)**	**SD T (s)**
ADMOEA	35.33	72.00	0.8233	0.0436	55.30	**16.08**
AMALGAM$_{\mathrm{ndp}}$	1.37	1.22	0.8512	0.0255	67.03	20.17
AMALGAM$_{\mathrm{ndu}}$	2.23	6.57	0.8226	0.0373	87.37	34.97
AMALGAM$_{\mathrm{ndug}}$	47.00	68.40	0.8018	0.0220	76.77	25.62
AMALGAMI$_{\mathrm{ndp}}$	1.60	2.92	0.8548	0.0285	68.30	16.95
AMALGAMI$_{\mathrm{ndu}}$	3.13	8.46	0.8252	0.0267	100.93	33.13
AMALGAMI$_{\mathrm{ndug}}$	47.17	65.02	0.8032	0.0281	85.10	33.93
AMALGAMS$_{\mathrm{ndp}}$	3.07	7.62	0.8601	*0.0563*	94.63	36.98
AMALGAMS$_{\mathrm{ndu}}$	2.37	4.80	0.7878	0.0295	115.17	*53.51*
ANIMA	8.63	23.29	0.8031	0.0270	67.70	21.66
DE	**1.23**	**0.82**	**0.9457**	0.0392	85.50	24.35
GREEDY	*628.40*	11.08	0.5284	0.0387	120.43	34.09
NSGA-II	5.30	9.87	0.8087	0.0171	**52.5**	19.31
PSO	622.80	71.46	*0.4928*	0.0461	80.80	42.88
PUMDA	68.03	38.48	0.8543	**0.0141**	120.00	17.43
SPEA-II	1.77	1.70	0.8392	0.0189	116.20	46.25
TAMALGAM$_{\mathrm{ndp}}$	1.57	2.75	0.8472	0.0244	73.17	21.24
TAMALGAM$_{\mathrm{ndu}}$	1.43	2.01	0.8137	0.0276	80.73	30.65
TAMALGAM$_{\mathrm{ndug}}$	33.63	52.65	0.8068	0.0226	80.43	37.25
TAMALGAMJ$_{\mathrm{ndp}}$	4.73	12.67	0.8406	0.0286	63.13	19.74
TAMALGAMJ$_{\mathrm{ndu}}$	1.97	5.29	0.8048	0.0256	78.57	29.90
TAMALGAMJ$_{\mathrm{ndug}}$	30.93	47.22	0.7970	0.0227	77.73	32.03
UMDA	411.40	*109.36*	0.7575	0.0188	*141.97*	48.52

Table 7.11*: Time (T) to convergence (taking as stopping criterion less than 0.05% change in HV for 200 generations) for the FOSS benchmark, computed over thirty optimisation runs for reference point* $(I_{\mathrm{n}}, C) = (0.4, 2\,000\,000)$.

Algorithm	**Rank**	**Avg d_{R}**	**SD d_{R}**	**Avg HV**	**SD HV**	**Avg A_{S}**	**SD A_{S}**
ADMOEA	11	1.67	1.54	0.8722	0.0337	65.07	8.78
AMALGAM$_{\mathrm{ndp}}$	5	1.03	0.18	0.8594	0.0173	62.87	0.97
AMALGAM$_{\mathrm{ndu}}$	13	1.77	4.20	0.8414	0.0313	62.93	0.78
AMALGAM$_{\mathrm{ndug}}$	19	13.50	22.02	0.8188	0.0190	62.60	1.10
AMALGAMI$_{\mathrm{ndp}}$	9	1.43	1.38	0.8643	0.0253	62.77	1.30
AMALGAMI$_{\mathrm{ndu}}$	2	1	0	0.8437	0.0197	62.90	1.09
AMALGAMI$_{\mathrm{ndug}}$	18	8.77	14.98	0.8234	0.0216	62.77	0.90
AMALGAMS$_{\mathrm{ndp}}$	8	1.33	0.66	0.8643	*0.0503*	56.00	3.03
AMALGAMS$_{\mathrm{ndu}}$	14	2.30	5.90	0.7967	0.0265	55.23	2.49
ANIMA	15	2.93	7.85	0.8162	0.0245	62.17	1.53
DE	**1**	1	0	**0.9459**	0.0298	63.07	1.05
GREEDY	*23*	*629.8*	9.33	0.5415	0.0371	53.47	6.10
NSGA-II	12	1.67	3.65	0.8227	0.0163	62.57	1.14
PSO	22	624.70	44.18	*0.5183*	0.0397	33.57	14.52
PUMDA	20	82.63	50.93	0.8605	**0.0137**	184.27	10.25
SPEA-II	6	1.10	0.31	0.8469	0.0190	55.13	2.79
TAMALGAM$_{\mathrm{ndp}}$	10	1.47	2.19	0.8562	0.0275	62.90	0.99
TAMALGAM$_{\mathrm{ndu}}$	3	1	0	0.8380	0.0211	62.77	1.07
TAMALGAM$_{\mathrm{ndug}}$	16	3.67	5.73	0.8249	0.0231	62.37	1.13
TAMALGAMJ$_{\mathrm{ndp}}$	4	1.03	0.18	0.8614	0.0246	62.83	0.91
TAMALGAMJ$_{\mathrm{ndu}}$	7	1.10	0.55	0.8269	0.0212	62.57	1.01
TAMALGAMJ$_{\mathrm{ndug}}$	17	5.87	15.00	0.8143	0.0187	62.70	1.18
UMDA	21	438.17	*100.56*	0.7655	0.0179	47.90	11.98

Table 7.12*: Mean and Standard Deviation of performance metrics for the full time trial analysis on the FOSS benchmark, computed over thirty optimisation runs for reference point* $(I_{\mathrm{n}}, C) = (0.4, 2\,000\,000)$.

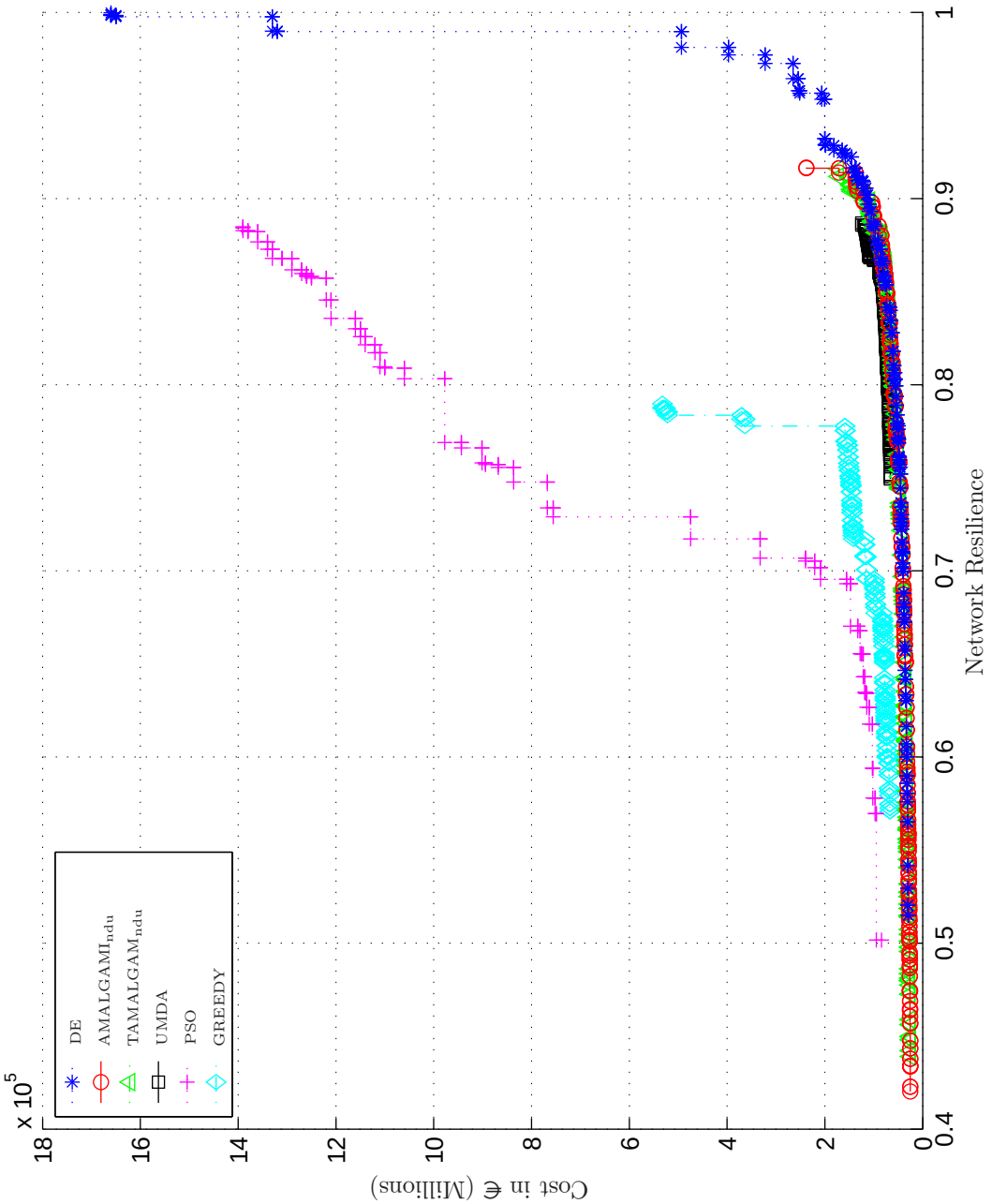


Figure 7.6: *Attainment fronts of the three best and three worst algorithms for the FOSS benchmark.*

Algorithm	**Avg d_{R}**	**SD d_{R}**	**Avg HV**	**SD HV**	**Avg T (s)**	**SD T (s)**
ADMOEA	59.43	86.08	0.832	*0.0382*	**67.33**	28.81
AMALGAM$_{\mathrm{ndp}}$	5.60	13.86	0.878	0.0226	112.60	34.51
AMALGAM$_{\mathrm{ndu}}$	1.60	**1.00**	0.872	0.0293	122.20	33.68
AMALGAM$_{\mathrm{ndug}}$	88.17	79.57	0.838	0.0211	195.00	60.71
AMALGAMI$_{\mathrm{ndp}}$	5.40	15.20	0.883	0.0144	123.27	43.43
AMALGAMI$_{\mathrm{ndu}}$	1.87	1.72	0.875	0.0260	128.77	35.24
AMALGAMI$_{\mathrm{ndug}}$	63.47	79.73	0.845	0.0180	223.03	69.60
AMALGAMS$_{\mathrm{ndp}}$	3.70	6.69	0.906	0.0230	219.37	83.81
AMALGAMS$_{\mathrm{ndu}}$	**1.50**	1.07	0.911	0.0123	*281.63*	74.72
ANIMA	8.00	19.16	0.884	0.0248	141.57	29.35
DE	2.17	2.91	**0.9261**	0.0248	145.87	47.77
GREEDY	*632.67*	5.35	*0.5808*	0.0246	232.67	78.77
NSGA-II	3.00	8.82	0.886	0.0153	112.67	25.68
PSO	540.77	*155.07*	0.627	0.0308	157.73	69.72
PUMDA	77.50	55.18	0.865	0.0203	244.53	**4.50**
SPEA-II	2.73	4.27	0.919	0.0115	207.07	45.63
TAMALGAM$_{\mathrm{ndp}}$	3.13	4.49	0.879	0.0163	104.90	29.44
TAMALGAM$_{\mathrm{ndu}}$	2.60	3.77	0.870	0.0201	128.83	45.54
TAMALGAM$_{\mathrm{ndug}}$	27.27	34.00	0.841	0.0172	143.47	38.27
TAMALGAMJ$_{\mathrm{ndp}}$	6.43	10.86	0.874	0.0192	107.07	37.67
TAMALGAMJ$_{\mathrm{ndu}}$	1.67	1.60	0.872	0.0160	116.77	28.01
TAMALGAMJ$_{\mathrm{ndug}}$	12.37	18.99	0.858	0.0162	157.47	50.24
UMDA	226.73	74.40	0.779	**0.0112**	244.30	*86.32*

Table 7.13*: Time (T) to convergence (taking as stopping criterion less than 0.05% change in HV for 200 generations) for the PESC benchmark, computed over thirty optimisation runs for reference point* $(I_{\mathrm{n}}, C) = (0.4, 15\,000\,000)$.

Algorithm	**Rank**	**Avg d_{R}**	**SD d_{R}**	**Avg HV**	**SD HV**	**Avg A_{S}**	**SD A_{S}**
ADMOEA	13	2.27	2.86	0.8607	*0.0352*	84.73	19.37
AMALGAM$_{\mathrm{ndp}}$	7	1.30	1.21	0.8872	0.0211	63.37	0.72
AMALGAM$_{\mathrm{ndu}}$	5	1.17	0.46	0.8824	0.0170	63.60	0.56
AMALGAM$_{\mathrm{ndug}}$	19	92.13	58.43	0.8416	0.0174	63.27	0.74
AMALGAMI$_{\mathrm{ndp}}$	15	7.53	25.40	0.8871	0.0147	63.20	0.81
AMALGAMI$_{\mathrm{ndu}}$	4	1.07	0.25	0.8849	0.0169	62.60	3.79
AMALGAMI$_{\mathrm{ndug}}$	18	71.20	44.78	0.8364	0.0188	62.87	1.38
AMALGAMS$_{\mathrm{ndp}}$	12	1.83	2.76	0.9062	0.0179	58.37	2.40
AMALGAMS$_{\mathrm{ndu}}$	9	1.60	1.57	0.9054	**0.0080**	56.23	1.91
ANIMA	14	3.37	8.44	0.8954	0.0236	62.60	1.89
DE	10	1.77	1.83	0.9187	0.0274	62.83	2.34
GREEDY	*23*	*633.07*	5.13	*0.5823*	0.0217	48.27	7.46
NSGA-II	**1**	1	0	0.8974	0.0112	63.33	0.71
PSO	22	546.43	*144.41*	0.6366	0.0275	41.93	10.79
PUMDA	20	92.97	76.89	0.8634	0.0200	201.33	8.40
SPEA-II	6	1.30	0.84	**0.9194**	0.0089	56.93	2.02
TAMALGAM$_{\mathrm{ndp}}$	8	1.53	1.55	0.8896	0.0166	63.37	0.76
TAMALGAM$_{\mathrm{ndu}}$	3	1.03	0.18	0.8869	0.0168	63.27	0.74
TAMALGAM$_{\mathrm{ndug}}$	17	15.47	19.38	0.8501	0.0174	63.03	1.52
TAMALGAMJ$_{\mathrm{ndp}}$	11	1.83	2.07	0.8852	0.0163	63.23	0.73
TAMALGAMJ$_{\mathrm{ndu}}$	2	1	0	0.8847	0.0134	63.37	0.67
TAMALGAMJ$_{\mathrm{ndug}}$	16	11.10	21.61	0.8596	0.0176	63.20	1.10
UMDA	21	246.93	69.47	0.7777	0.0110	38.87	6.45

Table 7.14*: Mean and Standard Deviation of performance metrics for the full time trial analysis on the PESC benchmark, computed over thirty optimisation runs for reference point* $(I_{\mathrm{n}}, C) = (0.4, 15\,000\,000)$.

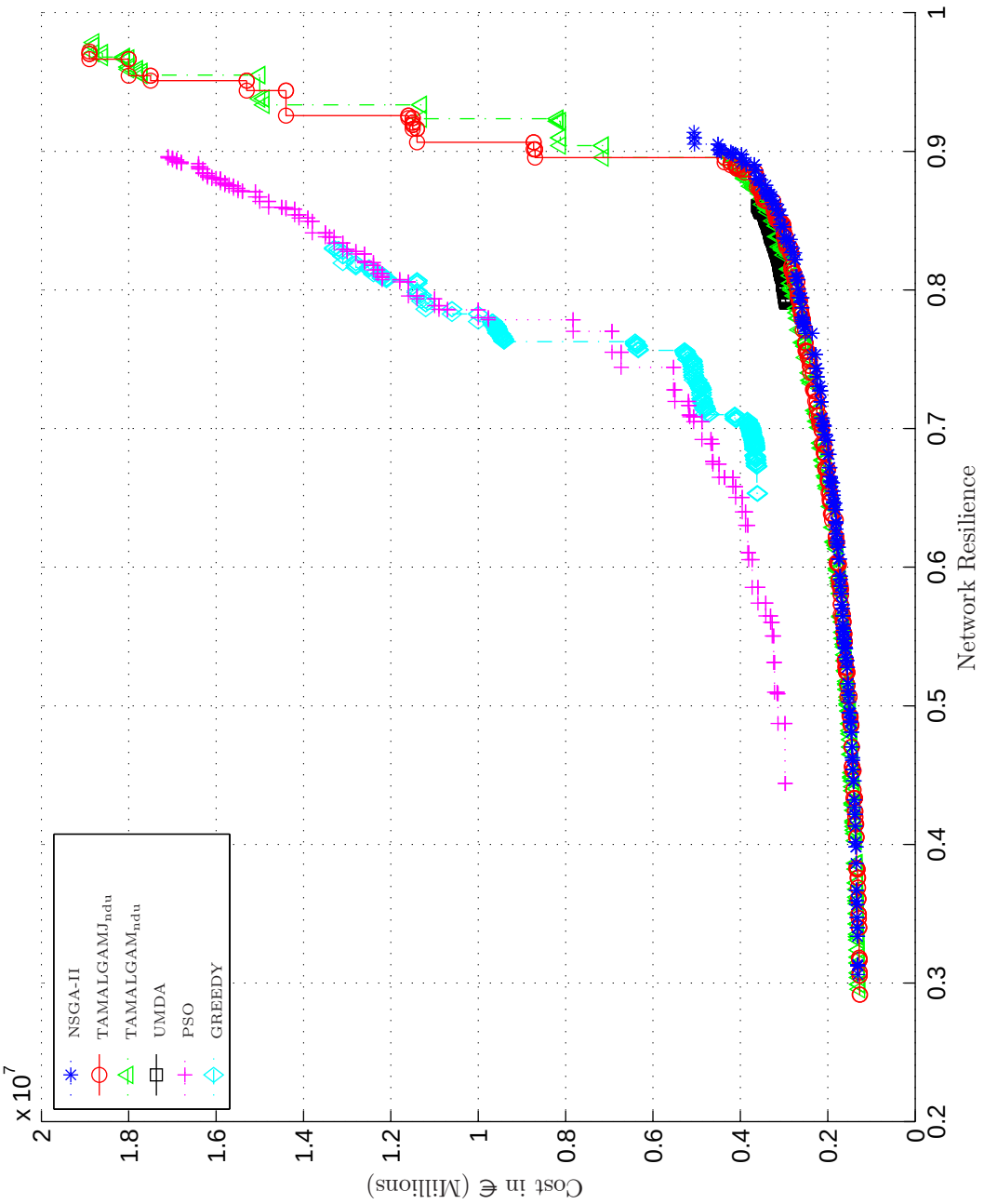


Figure 7.7: *Attainment fronts of the three best and three worst algorithms for the PESC benchmark.*

Plots of the attainment fronts of the three best and three worst algorithms appear in Figure 7.7. The graphical results are quite similar to those of FOSS, except that it is now the AMALGAM variants that find the upper reaches of the global Pareto-front, while the lower reaches are located by the majority of the best algorithms. The results of PSO and GREEDY form distinctive sub-fronts and those of UMDA approach a localized region of the global front near the region of maximal curvature.

7.1.8 MOD Benchmark Time Trials

The optimal population size for MOD was found to be 64 for the first group of algorithms, and 256 for UMDA and PUMDA.

The convergence times of the various algorithms for the MOD benchmark are documented in Table 7.15. The 90^{th} percentile of average times to convergence is 1310.21 seconds, significantly longer than for the previous benchmarks. The fastest average time to convergence of 288.00 seconds was attained by PSO, and the longest average convergence time of 1604.10 seconds was attained by $\text{AMALGAMI}_{\text{ndug}}$. The group average time to convergence was 892.14 seconds. UMDA demonstrated the lowest standard deviation for convergence time of 91.7 seconds, compared to the average standard deviation of 208.07 seconds. The algorithm demonstrating the best hypervolume attainment was SPEA-II with a value of 0.9453. The worst performing algorithm was once again GREEDY with an average dominance rank of 637.3.

In the full time trial, each algorithm was executed for thirty optimisation runs of length 1310 seconds. Algorithmic performance statistics for MOD are presented in Table 7.16. MOD is obviously the most challenging of the problems thus far, as no algorithm was able to obtain an average dominance rank of 1, with only three managing a value between 1 and 2. The best performing algorithm was NSGA-II with an average dominance rank of 1.20 and an average hypervolume of 0.9368, followed in second place by $\text{AMALGAMS}_{\text{ndp}}$ with an average dominance rank of 1.43 and an average hypervolume of 0.9301. The algorithm with the highest hypervolume was SPEA-II at 0.9523. The group average hypervolume was 0.8710. The algorithm with the smallest standard deviation in hypervolume was $\text{AMALGAMI}_{\text{ndu}}$ with a value of 0.0070. The worst performing algorithm was GREEDY, with an average dominance rank of 638.10 and an average hypervolume of 0.5337. ADMOEA produced an average ϵ-archive size of 69.80. PUMDA and UMDA produced ϵ-archive sizes of 166.83 and 15.57, respectively. The largest ϵ-archive size amongst the other algorithms was 62.27, obtained by both $\text{AMALGAM}_{\text{ndp}}$ and $\text{AMALGAMI}_{\text{ndp}}$.

Plots of the attainment fronts of the three best and three worst algorithms appear in Figure 7.8. These results demonstrate the principle that there is a trade-off between the performance of the various algorithms, with MOD having a non-trivial Pareto-front. SPEA-II failed to reach the Pareto-front in the low Network Resilience region of the objective space, but was the best at finding the high Network Resilience solutions. NSGA-II was the best mid-range performer and $\text{AMALGAMS}_{\text{ndp}}$ located unique solutions of low cost. GREEDY and PSO performed extremely poorly in comparison to the best algorithms, locating solutions of more than double the cost for similar Network Resilience.

7.1.9 Summary and Analysis of First Eight Benchmarks in Phase 1

The results for the first eight benchmark tests (TRP, TLN, HANOI, NYTUN, BLACK, FOSS, PESC, and MOD) are summarized in Tables 7.17 and 7.18, showing results for the convergence

Algorithm	**Avg d_{R}**	**SD d_{R}**	**Avg HV**	**SD HV**	**Avg T (s)**	**SD T (s)**
ADMOEA	181.43	*143.31*	0.7855	*0.0352*	377.20	111.6
AMALGAM$_{\mathrm{ndp}}$	39.40	38.39	0.9145	0.0142	592.23	113.1
AMALGAM$_{\mathrm{ndu}}$	11.30	18.81	0.9061	0.0116	816.50	146.7
AMALGAM$_{\mathrm{ndug}}$	304.57	88.88	0.8535	0.0113	1473.30	*406.843*
AMALGAMI$_{\mathrm{ndp}}$	54.73	68.36	0.9128	0.0175	560.00	106.0
AMALGAMI$_{\mathrm{ndu}}$	9.97	20.34	0.9069	0.0098	867.77	160.5
AMALGAMI$_{\mathrm{ndug}}$	288.93	103.13	0.8501	0.0155	*1604.10*	387.7
AMALGAMS$_{\mathrm{ndp}}$	**3.50**	4.39	0.9346	0.0119	1229.33	246.9
AMALGAMS$_{\mathrm{ndu}}$	6.30	8.56	0.9362	0.0113	1114.00	272.8
ANIMA	15.67	33.50	0.9294	0.0165	867.60	195.7
DE	194.40	131.73	0.8690	0.0269	1124.93	379.9
GREEDY	*637.3*	5.68	*0.5210*	0.0219	1013.80	381.6
NSGA-II	6.63	11.51	0.9277	0.0094	736.20	116.3
PSO	634.57	**3.95**	0.5403	0.0225	**288.00**	137.0
PUMDA	123.63	86.11	0.8750	0.0183	1330.43	311.1
SPEA-II	8.00	8.55	**0.9453**	0.0092	862.27	181.6
TAMALGAM$_{\mathrm{ndp}}$	25.00	31.00	0.9151	0.0094	606.27	127.1
TAMALGAM$_{\mathrm{ndu}}$	5.20	8.79	0.9105	0.0102	909.37	157.9
TAMALGAM$_{\mathrm{ndug}}$	135.73	55.30	0.8777	0.0125	1147.70	201.6
TAMALGAMJ$_{\mathrm{ndp}}$	44.57	39.68	0.9142	0.0146	609.57	144.7
TAMALGAMJ$_{\mathrm{ndu}}$	9.50	20.61	0.9136	0.0117	866.70	211.4
TAMALGAMJ$_{\mathrm{ndug}}$	115.60	65.43	0.8901	0.0145	1000.40	195.9
UMDA	234.63	60.60	0.8269	**0.0066**	521.43	**91.7**

Table 7.15*: Time (T) to convergence (taking as stopping criterion less than 0.05% change in HV for 200 generations) for the MOD benchmark, computed over thirty optimisation runs for reference point* $(I_{\mathrm{n}}, C) = (0, 15\,000\,000)$.

Algorithm	**Rank**	**Avg d_{R}**	**SD d_{R}**	**Avg HV**	**SD HV**	**Avg A_{S}**	**SD A_{S}**
ADMOEA	11	6.93	18.95	0.8874	*0.0492*	69.80	8.89
AMALGAM$_{\mathrm{ndp}}$	9	6.50	7.10	0.9259	0.0127	62.27	1.20
AMALGAM$_{\mathrm{ndu}}$	8	3.03	3.15	0.9160	0.0096	61.07	1.76
AMALGAM$_{\mathrm{ndug}}$	21	393.07	39.93	0.8509	0.0092	60.10	2.07
AMALGAMI$_{\mathrm{ndp}}$	10	6.60	8.63	0.9287	0.0106	62.27	1.55
AMALGAMI$_{\mathrm{ndu}}$	7	2.90	3.75	0.9135	**0.0070**	61.87	1.36
AMALGAMI$_{\mathrm{ndug}}$	20	390.63	47.64	0.8491	0.0099	59.77	2.85
AMALGAMS$_{\mathrm{ndp}}$	2	1.43	1.10	0.9301	0.0120	61.97	1.45
AMALGAMS$_{\mathrm{ndu}}$	13	46.20	36.69	0.8944	0.0104	60.30	3.13
ANIMA	6	2.50	3.72	0.9359	0.0134	60.60	1.79
DE	19	208.50	*120.06*	0.8687	0.0223	60.80	2.30
GREEDY	*23*	*638.10*	6.37	*0.5337*	0.0182	32.40	7.57
NSGA-II	**1**	**1.20**	**0.48**	0.9368	0.0102	61.93	1.74
PSO	22	633.57	2.56	0.5673	0.0184	23.53	5.41
PUMDA	15	123.30	92.44	0.8746	0.0186	166.83	11.01
SPEA-II	3	1.53	0.94	**0.9523**	0.0074	56.30	2.26
TAMALGAM$_{\mathrm{ndp}}$	14	52.87	65.73	0.9133	0.0233	61.20	3.57
TAMALGAM$_{\mathrm{ndu}}$	5	2.47	1.91	0.9209	0.0100	61.63	1.79
TAMALGAM$_{\mathrm{ndug}}$	17	160.80	72.83	0.8833	0.0101	60.80	2.07
TAMALGAMJ$_{\mathrm{ndp}}$	12	42.97	57.92	0.9141	0.0220	59.97	9.33
TAMALGAMJ$_{\mathrm{ndu}}$	4	2.10	2.22	0.9191	0.0093	61.77	1.72
TAMALGAMJ$_{\mathrm{ndug}}$	16	145.03	66.62	0.8880	0.0123	60.90	2.11
UMDA	18	208.30	68.94	0.8285	0.0075	15.57	5.53

Table 7.16*: Mean and Standard Deviation of performance metrics for the full time trial analysis on the MOD benchmark, computed over thirty optimisation runs for reference point* $(I_{\mathrm{n}}, C) = (0, 15\,000\,000)$.

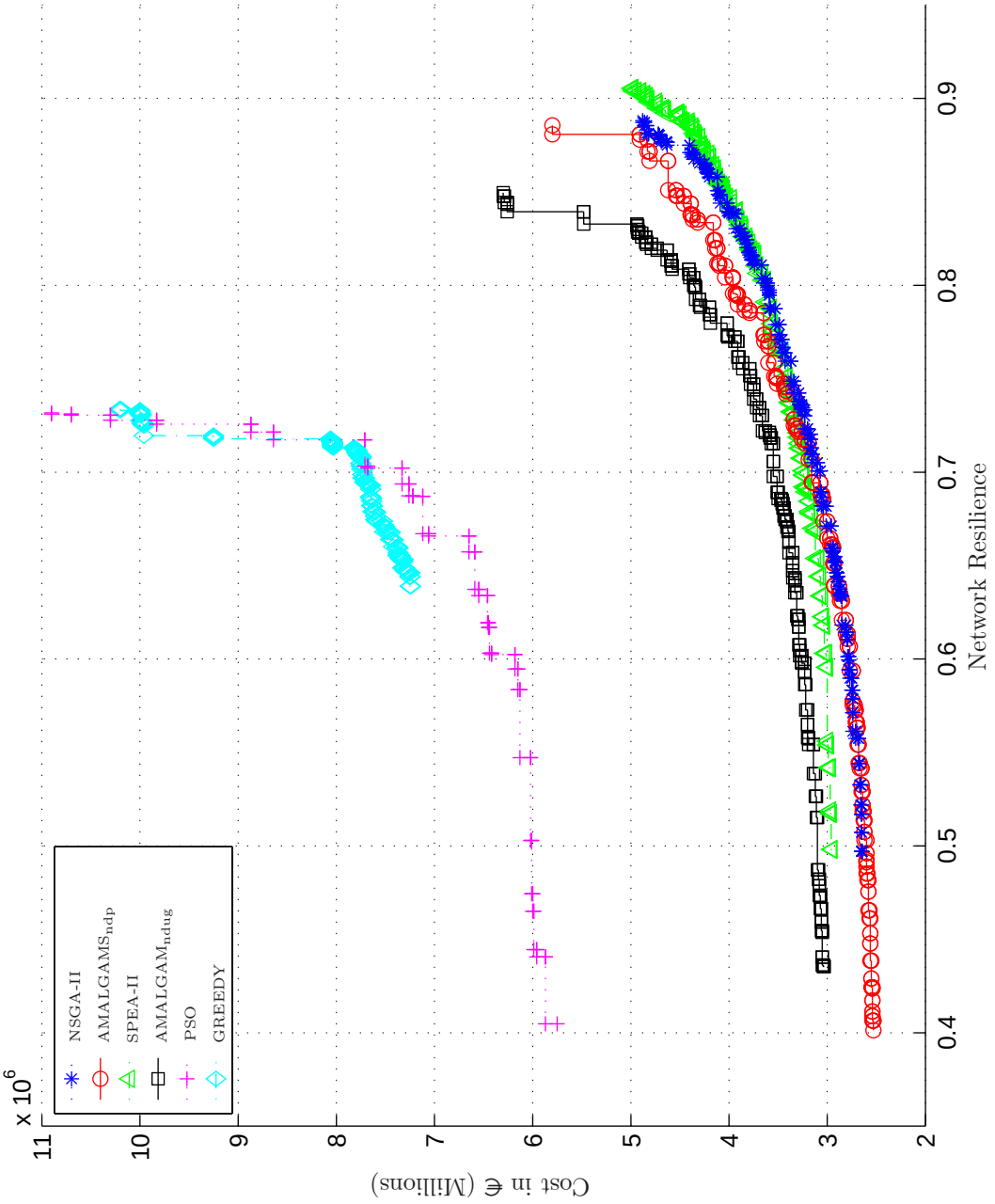


Figure 7.8*: Attainment fronts of the three best and three worst algorithms for the MOD benchmark.*

analysis and time trial analysis, respectively. These performance metrics have been averaged across all eight benchmarks. Table 7.17 reports the averages of average and standard deviation for d_{R}, the averages of average and standard deviation for NHV, and the averages of average and standard deviation for normalised time (NT) (where time has been normalised by the average convergence times for each benchmark). This table is sorted in terms of increasing average NT. Table 7.18 reports the average rank, the averages of average and standard deviation for d_{R}, the averages of average and standard deviation for NHV, and the averages of average ϵ-archive size A_{S}. This table is sorted in terms of increasing average dominance rank.

The results from the convergence analysis indicate that ADMOEA was the fastest algorithm, requiring on average 0.5279 of the average time to converge in order to achieve on average 0.9063 of the best known NHV. This can be attributed to its dynamic population sizing and offspring generation methodologies which typically process fewer solutions per evolutionary generation. However, an average dominance rank of 43.23 and the highest standard deviation of 0.0230 for NHV indicates that the ADMOEA speed enhancement comes at the cost of missed Pareto-optimal solutions. The closest competitor in terms of time is $\mathrm{AMALGAM}_{\mathrm{ndp}}$ with 0.7327 of the average time to convergence for an average NHV of 0.9426, and the best SD NT of 0.1945, revealing that it is the most consistent performer in terms of convergence time. With an average dominance rank of 6.43 $\mathrm{AMALGAM}_{\mathrm{ndp}}$ still lagged behind the leading algorithms, showing that the speed enhancement comes at the cost of reduced performance, but that it may be suitable for use in time critical applications.

The algorithm with the best average NHV is $\mathrm{AMALGAM}_{\mathrm{ndu}}$, the original formulation which achieves an average of NHV of 0.9502 in 0.8218 of the average convergence time. With an average dominance rank of 2.39, this algorithm would seem suitable for general use. The algorithm with the best average and standard deviation of dominance rank is $\mathrm{TAMALGAM}_{\mathrm{ndu}}$, with values of 1.78 and 1.82, respectively. It also attains a good average NHV of 0.9345 with a standard deviation of 0.0107. This is achieved at an average convergence time of 0.8754 of the mean. $\mathrm{TAMALGAMJ}_{\mathrm{ndu}}$ and NSGA-II also achieve good performance in below average time, and could be considered candidates for general use. The worst performing algorithm in terms of time is SPEA-II, which requires 1.7043 of the mean convergence time, but yields good results with a dominance rank of 2.45 and an average NHV of 0.9467. SPEA-II is also the best in terms of NHV standard deviation with less than 1%. The worst performing algorithm in the convergence trials is GREEDY with an average dominance rank of 397.72 and average NHV of 0.7571. It is also slower than the average convergence rate at 1.1876.

Figure 7.9 shows the performance trade-off between average NT and average NHV for each of the algorithms, where the algorithms forming the Pareto-front have been labelled. The standard deviations of average NHV are also indicated on the secondary axis. The Pareto-ranked algorithms in terms of average convergence time and average NHV are $\mathrm{AMALGAM}_{\mathrm{ndu}}$, $\mathrm{AMALGAMI}_{\mathrm{ndp}}$, $\mathrm{AMALGAM}_{\mathrm{ndp}}$ and ADMOEA. Not shown in the graph are the non-dominated algorithms with respect to average convergence time and average dominance rank: $\mathrm{TAMALGAMJ}_{\mathrm{ndu}}$, $\mathrm{TAMALGAM}_{\mathrm{ndu}}$, $\mathrm{TAMALGAM}_{\mathrm{ndp}}$, $\mathrm{AMALGAM}_{\mathrm{ndu}}$ and $\mathrm{AMALGAM}_{\mathrm{ndp}}$. Of these algorithms, only $\mathrm{AMALGAM}_{\mathrm{ndu}}$ and $\mathrm{AMALGAM}_{\mathrm{ndp}}$ are non-dominated in terms of all three performance criteria.

The results from the time trial analysis indicate that NSGA-II is the top performing algorithm in terms of average dominance rank with a value of 1.1. This seems surprising given its simplicity. However, its average rank is only 9, compared to $\mathrm{TAMALGAM}_{\mathrm{ndu}}$ which has an average rank value of 5.125. $\mathrm{TAMALGAM}_{\mathrm{ndu}}$ furthermore achieved the lowest standard deviation for NHV of 0.26, compared to 0.52 for NSGA-II. The best algorithm in terms of average NHV is $\mathrm{AMALGAMS}_{\mathrm{ndp}}$ with a value of 0.9512. The algorithm with the lowest

Convergence Analysis Summary						
Algorithm	**Avg d_R**	**SD d_R**	**Avg NHV**	**SD NHV**	**Avg NT**	**SD NT**
ADMOEA	43.23	59.02	0.9063	*0.0230*	**0.5279**	0.4057
$AMALGAM_{ndp}$	6.43	6.73	0.9426	0.0113	0.7327	**0.1945**
$AMALGAMI_{ndp}$	8.36	10.90	0.9429	0.0111	0.7451	0.2161
$TAMALGAM_{ndp}$	4.35	4.83	0.9418	0.0100	0.7490	0.1978
$TAMALGAMJ_{ndp}$	7.99	9.75	0.9402	0.0116	0.7817	0.2449
$AMALGAMI_{ndu}$	7.98	11.31	0.9323	0.0109	0.8175	0.2119
$AMALGAM_{ndu}$	2.39	2.58	**0.9502**	0.0138	0.8218	0.2091
$TAMALGAMJ_{ndu}$	2.28	3.48	0.9341	0.0105	0.8413	0.2224
DE	27.49	23.38	0.9289	0.0147	0.8576	0.2449
NSGA-II	2.52	3.89	0.9356	0.0110	0.8673	0.2548
$TAMALGAM_{ndu}$	**1.78**	**1.82**	0.9345	0.0107	0.8754	0.2299
$TAMALGAMJ_{ndug}$	20.49	16.48	0.9277	0.0103	0.9247	0.2615
$TAMALGAM_{ndug}$	25.20	17.74	0.9250	0.0100	0.9484	0.2621
ANIMA	4.23	8.40	0.9340	0.0130	0.9587	0.2875
PSO	360.65	43.33	*0.7492*	0.0223	1.0213	0.3592
$AMALGAM_{ndug}$	50.00	21.92	0.9215	0.0127	1.0743	0.3171
$AMALGAMI_{ndug}$	45.09	23.98	0.9227	0.0111	1.1240	0.3307
GREEDY	*397.72*	55.81	0.7571	0.0226	1.1876	0.3220
UMDA	251.35	*85.58*	0.8148	0.0211	1.2166	0.4595
PUMDA	123.21	68.33	0.8899	0.0198	1.2738	0.2254
$AMALGAMS_{ndp}$	1.91	2.34	0.9493	0.0147	1.3158	0.4031
$AMALGAMS_{ndu}$	7.51	9.38	0.9414	0.0101	1.6331	0.5011
SPEA-II	2.45	2.63	0.9467	**0.0096**	*1.7043*	*0.5546*
Average	61.07	21.46	0.9117	0.0137	1	0.3007

Table 7.17*: Summary statistics of Phase 1 convergence analysis, with average performance metrics computed over eight benchmarks (TRP, TLN, HANOI, NYTUN, BLACK, PESC, FOSS, MOD), listed in order of increasing normalised convergence time.*

standard deviation for NHV is $TAMALGAMJ_{ndu}$ with a value of 0.0085. All four of these algorithms are non-dominated with respect to the various performance criteria, and each of them may be considered a candidate for general usage.

The performance results for all algorithms with an average dominance rank lower than 4 are graphed in Figure 7.10, with the Pareto-front solutions labelled. SPEA-II has also been labelled due to its smaller standard deviation for NHV than $AMALGAMS_{ndp}$. The newly developed algorithm ANIMA performed reasonably well, but did not make it into the top five algorithms. It is thought that its adaptive mechanisms may potentially be refined in a more rational manner to improve performance, possibly through the use of machine learning techniques. ADMOEA just fell short of claiming a top-ten position, with its greatest weakness being high performance variability (it has the highest standard deviation for NHV of 0.0223). However, its fast convergence time may be reason enough to consider it for general usage. The worst performing algorithms are PUMDA, UMDA, PSO and GREEDY, where the latter is undeniably the worst algorithm for WDS design given this sample of benchmarks. The poor performance of UMDA is probably expected due to its lack of innovation, and that of GREEDY may be explained by the fact that it is a local search.

Due to their superior performance in the time trials, the four algorithms NSGA-II, $TAMALGAM_{ndu}$, $TAMALGAMJ_{ndu}$ and $AMALGAMS_{ndp}$ were selected to compete in solving the large EXNET WDS design problem, covered in the next section.

Time Trial Analysis Summary						
Algorithm	**Avg Rank**	**Avg** d_R	**SD** d_R	**Avg HV**	**SD HV**	**Avg** A_S
NSGA-II	9.000	**1.11**	0.52	0.9421	0.0092	64.61
$\text{TAMALGAMJ}_{\text{ndu}}$	5.875	1.15	0.35	0.9423	**0.0085**	64.82
$\text{TAMALGAM}_{\text{ndu}}$	**5.125**	1.19	**0.26**	0.9439	0.0092	64.90
$\text{AMALGAMS}_{\text{ndp}}$	9.125	1.21	0.61	**0.9512**	0.0128	63.43
$\text{AMALGAMI}_{\text{ndu}}$	6.000	1.25	0.50	0.9428	0.0086	64.55
SPEA-II	12.375	1.36	0.89	0.9476	0.0089	61.61
$\text{AMALGAM}_{\text{ndu}}$	9.250	1.37	0.98	0.9423	0.0105	64.55
$\text{AMALGAM}_{\text{ndp}}$	8.000	1.73	1.06	0.9474	0.0099	65.13
ANIMA	12.375	1.77	2.71	0.9416	0.0112	64.30
$\text{AMALGAMI}_{\text{ndp}}$	9.750	2.57	4.43	0.9483	0.0095	64.91
ADMOEA	12.125	3.83	10.97	0.9321	*0.0223*	73.41
$\text{TAMALGAMJ}_{\text{ndp}}$	10.750	6.37	7.58	0.9463	0.0113	65.37
$\text{AMALGAMS}_{\text{ndu}}$	12.125	6.97	5.70	0.9360	0.0088	62.17
$\text{TAMALGAM}_{\text{ndp}}$	12.375	7.63	8.75	0.9460	0.0117	65.57
$\text{TAMALGAMJ}_{\text{ndug}}$	10.875	20.87	12.90	0.9330	0.0090	64.51
$\text{TAMALGAM}_{\text{ndug}}$	10.125	23.12	12.24	0.9322	0.0095	64.51
DE	12.250	27.81	17.86	0.9470	0.0147	64.48
$\text{AMALGAMI}_{\text{ndug}}$	13.250	59.45	13.42	0.9240	0.0094	63.94
$\text{AMALGAM}_{\text{ndug}}$	13.625	62.96	15.05	0.9245	0.0093	64.22
PUMDA	17.500	114.73	78.33	0.8933	0.0190	129.06
UMDA	21.000	234.76	*90.15*	0.8218	0.0193	57.10
PSO	20.875	360.43	41.87	*0.7600*	0.0190	42.71
GREEDY	*22.25*	*410.64*	46.19	0.7614	0.0205	56.23
Average	12.000	58.88	16.23	0.9177	0.0123	65.92

Table 7.18*: Summary statistics of Phase 1 time trials, with average performance metrics computed over eight benchmarks (TRP, TLN, HANOI, NYTUN, BLACK, PESC, FOSS, MOD), listed in order of increasing average dominance rank.*

7.1.10 EXNET Benchmark Time Trials

EXNET is very large in comparison to the previous WDS benchmarks, and consequently requires a large increase in computational processing time in order to conduct a hydraulic simulation, as well as a correspondingly large increase in memory resources. If the full time trials were to be performed for all the algorithms, as was done for the previous benchmarks, it would take several months of computing time. Only the four top performing algorithms for the first eight WDS benchmarks in Phase 1 were therefore considered for the EXNET analysis, namely NSGA-II, $\text{TAMALGAMJ}_{\text{ndu}}$, $\text{TAMALGAM}_{\text{ndu}}$, and $\text{AMALGAMS}_{\text{ndp}}$. The combined convergence and time trial analysis with thirty optimisation runs required more than a month of computing time. EXNET does not admit any feasible solutions which completely satisfy the pressure head requirements, owing to demand node 1 107, where the minimum head is unattainable due to the node elevation. Therefore, this node was ignored, classifying solutions as feasible if the minimum head is satisfied at all other demand nodes. The point (0, 5 000 000 000) was used as the reference point for hypervolume calculation.

The results of the convergence analysis are shown in Table 7.19. $\text{AMALGAMS}_{\text{ndp}}$ demonstrated the best performance in terms of all metrics, at the cost of the longest average convergence time of 19 507 seconds (5.4 hours). In addition to the lowest average dominance rank of 1.4 and the highest average hypervolume of 0.9014, it also provided the most reliable performance with the lowest standard deviations of 0.84, 0.0444 and 2 638 seconds for dominance rank, hypervolume and convergence time, respectively. The algorithm with the fastest convergence time was $\text{TAMALGAM}_{\text{ndp}}$ with a time of 10 312 seconds (2.9 hours); however this comes at

Algorithm	Avg d_R	SD d_R	Avg HV	SD HV	Avg T (s)	SD T (s)
TAMALGAM$_{ndu}$	15.7	4.42	0.7086	0.0529	**10312**	4652
TAMALGAMJ$_{ndu}$	14.5	8.68	0.6526	0.1381	11344	3575
NSGA-II	15.6	6.02	0.7256	0.0715	10821	2942
AMALGAMS$_{ndp}$	**1.4**	**0.84**	**0.9014**	**0.0444**	19507	**2638**

Table 7.19*: Time (T) to convergence (taking as stopping criterion less than 0.05% change in HV for 200 generations) for the EXNET benchmark, computed over thirty optimisation runs for reference point* $(I_n, C) = (0, 5\,000\,000\,000)$.

Algorithm	Rank	Avg d_R	SD d_R	Avg HV	SD HV	Avg A_S	SD A_S
TAMALGAM$_{ndu}$	2	12.80	23.33	0.7143	0.0856	24.27	6.16
TAMALGAMJ$_{ndu}$	3	14.13	26.05	0.6090	0.1302	29.37	8.70
NSGA-II	4	49.03	20.62	0.5648	0.0432	16.70	**2.39**
AMALGAMS$_{ndp}$	1	**3.23**	**3.62**	**0.8632**	**0.0429**	**36.53**	6.46

Table 7.20*: Mean and Standard Deviation of performance metrics for the full time trial analysis on the EXNET benchmark, computed over thirty optimisation runs for reference point* $(I_n, C) = (0, 5\,000\,000\,000)$.

the cost of greatly reduced performance, with average dominance rank and hypervolume values of 15.7 and 0.7086, respectively. The results produced by TAMALGAMJ$_{ndu}$ and NSGA-II lie somewhere in-between, mutually non-dominated with respect to dominance rank, hypervolume and convergence time.

Full-length time trials were conducted using thirty optimisation runs and a time limit of 19 507 seconds for each algorithm. The results of these trials are shown in Table 7.20. AMALGAMS$_{ndp}$ maintained a convincing lead, with an average dominance rank of 3.23 and an average hypervolume of 0.8632. It once again demonstrated the lowest standard deviation values of 3.62 and 0.0429 for dominance rank and hypervolume, respectively, and obtained a significantly larger average ϵ-archive size of 36.53, indicating in combination with dominance rank that it located a much larger portion of the Pareto-front than did the other algorithms. The closest competitor was TAMALGAM$_{ndu}$ with an average dominance rank of 12.80 and an average hypervolume of 0.7143. TAMALGAMJ$_{ndu}$ takes third position with an average dominance rank of 14.13 and an average hypervolume of 0.6090. NSGA-II was the worst of the four algorithms by a substantial margin, with an average dominance rank of 49.03 and an average hypervolume of 0.5648.

The attainment sets achieved by the various algorithms are graphed in NR-Cost space in Figure 7.11. Here it is clear that AMALGAMS$_{ndp}$ located a much larger portion of the Pareto-front, exclusively providing the global front from NR-values of 0.469 to 0.539.

However, in the region of NR-values from approximately 0.54 to 0.561, TAMALGAM$_{ndu}$ found the majority of the non-dominated solutions, exclusively representing most of that section of the global Pareto-front. TAMALGAMJ$_{ndu}$ and NSGA-II located dominated sub-fronts, with NSGA-II exhibiting the most localised and dominated attainment front. The algorithms' attainment fronts converge at an NR-value of approximately 0.551 and diverge again thereafter. From an NR of approximately 0.555 onwards, AMALGAMS$_{ndp}$ found only dominated solutions, suggesting that it may not be the best method for discovering solutions of high reliability. The least expensive solution (also representing the lowest Network Resilience), found by AMALGAMS$_{ndp}$, has a cost of 23 168 000 and a Network Resilience of 0.469 758. The most resilient solution, found

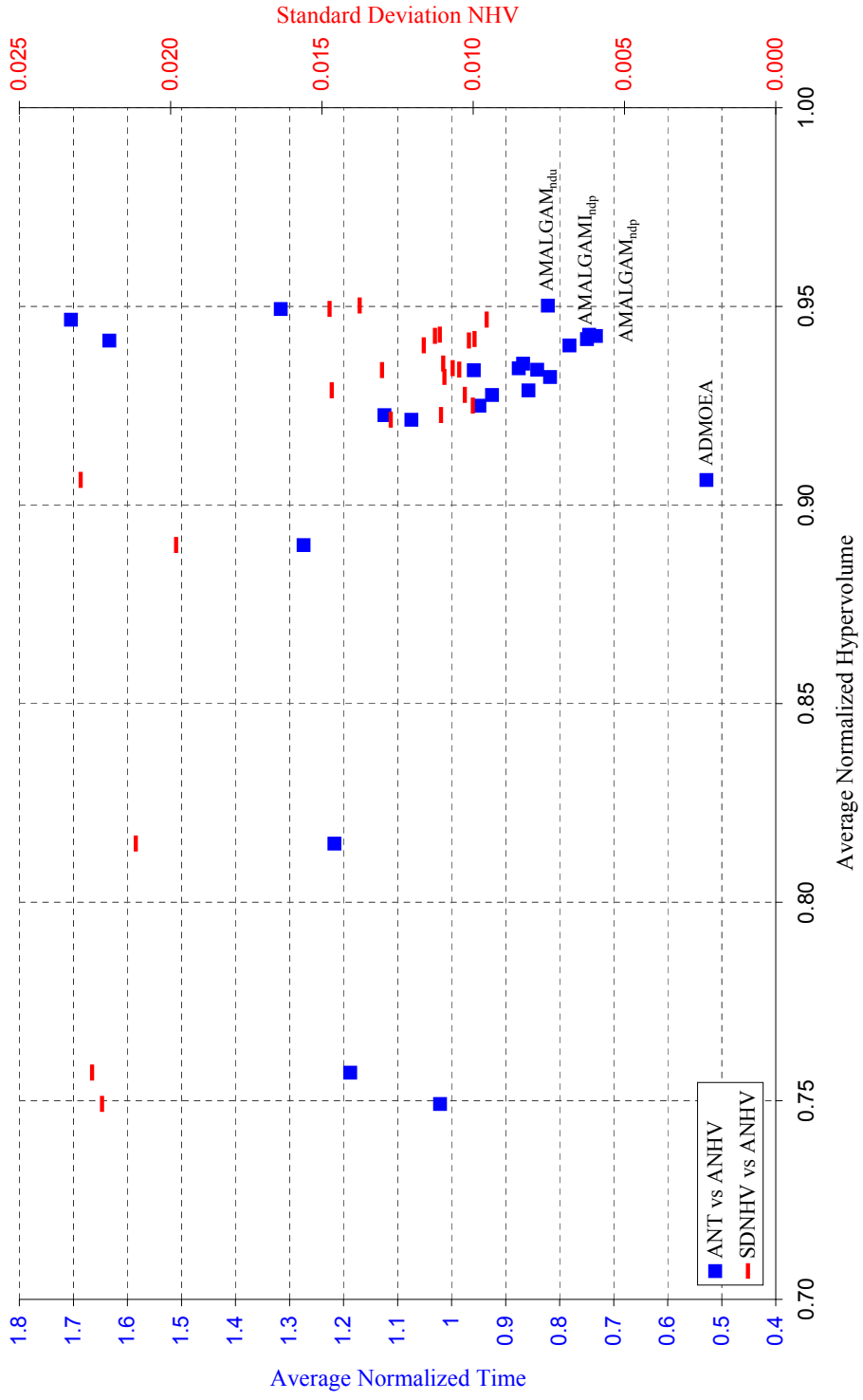


Figure 7.9: *Summary statistics for convergence analysis: Average normalised hypervolume vs average normalised convergence time.*

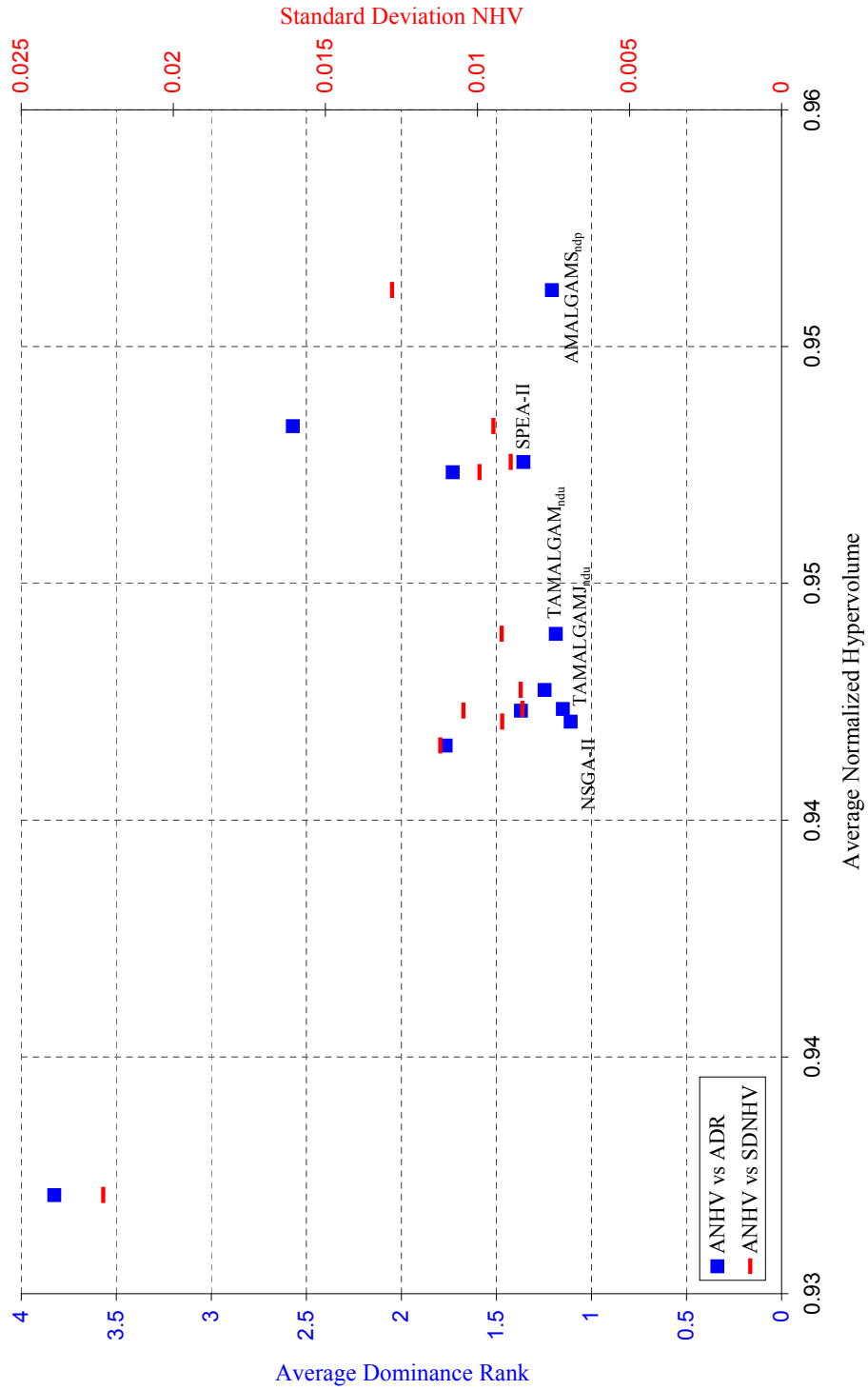


Figure 7.10: *Summary statistics for time trial analysis: Average normalised hypervolume vs average dominance rank.*

by TAMALGAMJ$_{\text{ndu}}$, has a cost of 38 757 400, and a Network Resilience of 0.561 725.

7.1.11 Performance Analysis of AMALGAM Sub-algorithms

The AMALGAM sub-algorithms demonstrated very similar behaviour over the different WDS benchmarks. Therefore, only the sub-algorithm analysis of AMALGAM$_{\text{ndu}}$ and AMALGAM$_{\text{ndug}}$ for the HANOI benchmark is presented here. The average number of successful offspring per generation (averaged over thirty optimisation runs) is shown for each sub-algorithm, as well as the sum total of successful offspring, for AMALGAM$_{\text{ndu}}$ in Figure 7.12 and for AMALGAM$_{\text{ndug}}$ in Figure 7.13.

Both implementations of AMALGAM experienced a brief initial peak of UMDA successes, attributable to the rapid recombination of existing healthy genetic building blocks in the initial population, which is UMDA's primary strength. Once the population becomes more diluted with building blocks — including new ones uncovered by the other sub-algorithms, individual building blocks have a smaller probability of being selected, and the usefulness of UMDA diminishes.

In AMALGAM$_{\text{ndu}}$ the dominant sub-algorithm is clearly NSGA-II, which produced the most successful offspring from generation 8 onwards. In second place is DE, which rises to an initial peak and experiences a steady decline until approximately generation 200. In AMALGAM$_{\text{ndug}}$, the relative ranking of the previous sub-algorithms is maintained; however, the GREEDY algorithm is now dominant. The scales of the two graphs are identical, in order to provide the reader with a visual comparison of the average quantity of solutions found by each algorithm. It is obvious that AMALGAM$_{\text{ndu}}$ found many more solutions than did AMALGAM$_{\text{ndug}}$, which experienced a sharp decline in solutions found from generation 50 onwards. This may be described as a premature convergence phenomenon, caused by the GREEDY algorithm which places an excess of pressure to locate local optima. The reason GREEDY is dominant is because it is able to find numerous neighbouring solutions in each generation, which typically provides it with more successful offspring than the competing sub-algorithms. However, these GREEDY offspring typically differ only slightly in terms of fitness from their parent solutions, due to their closeness in phenotype space. Therefore, despite the fact that the metaheuristic sub-algorithms may be vastly superior in terms of the magnitude of fitness improvement of successful offspring, because the number of such offspring generated is much smaller than the number of GREEDY offspring, they do not benefit from the existing AMALGAM reward scheme. This illustrates a fundamental problem with the original AMALGAM strategy, which is why alternative offspring partitioning / reward strategies were considered in this dissertation.

7.1.12 Performance Analysis of GREEDY Heuristic Steps

It is interesting to consider the performance of the GREEDY algorithm substeps, in terms of the proportion of successful offspring generated by each substep, as shown in Figures 7.14–7.17 for the TRP, TLN, HANOI and NYTUN benchmarks, respectively. It is clear that the Efficient Path is the most successful heuristic step, followed by the CANDA replacement method and the Cost-Power step. The Min/Max Headloss step is the next most successful; however, it frequently identifies the same pipe as the Headloss Gradient step (reducing the offspring count for this step, since they are executed in turn).

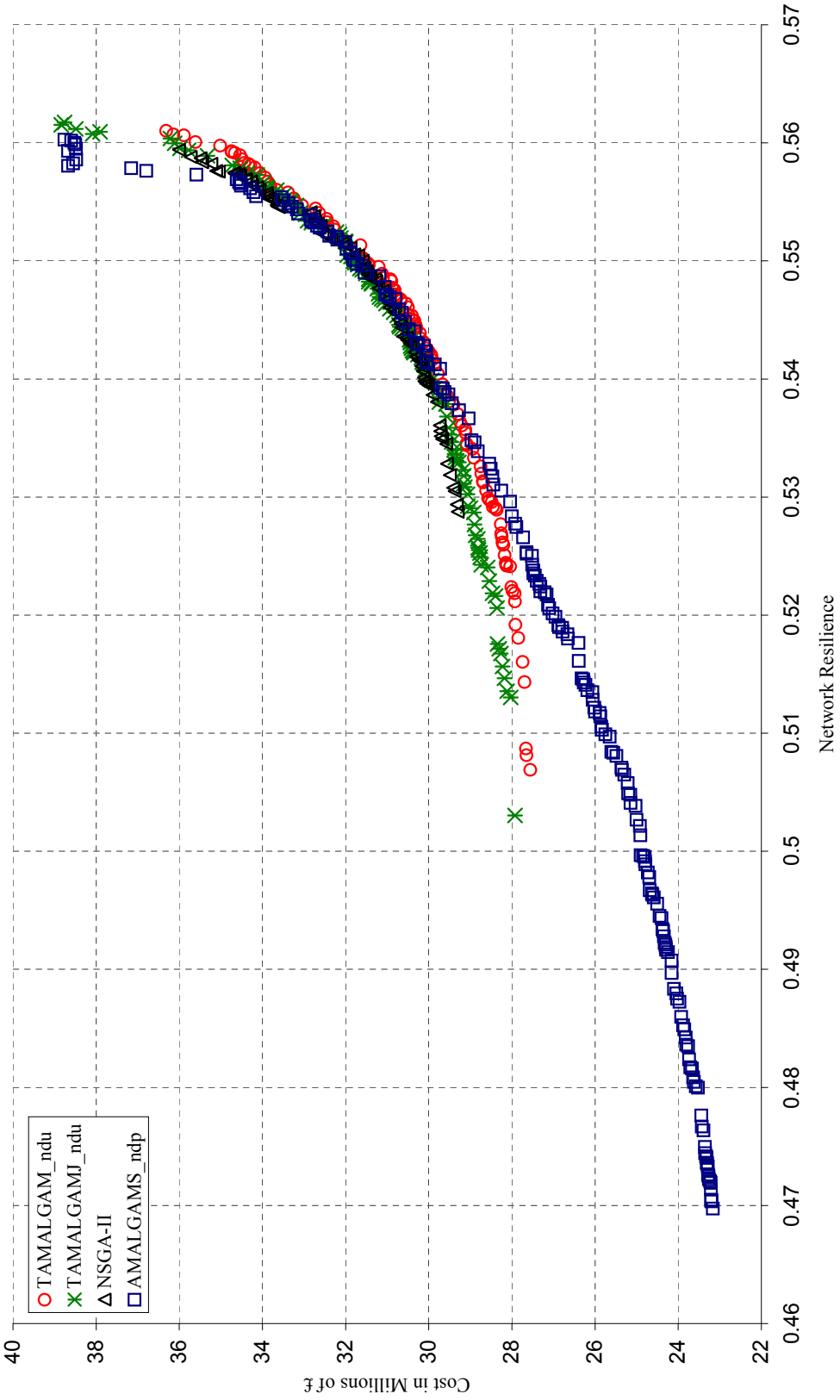


Figure 7.11: *Attainment sets found by TAMALGAM*$_{\text{ndu}}$*, TAMALGAMJ*$_{\text{ndu}}$*, NSGA-II and AMALGAMS*$_{\text{ndp}}$ *for EXNET.*

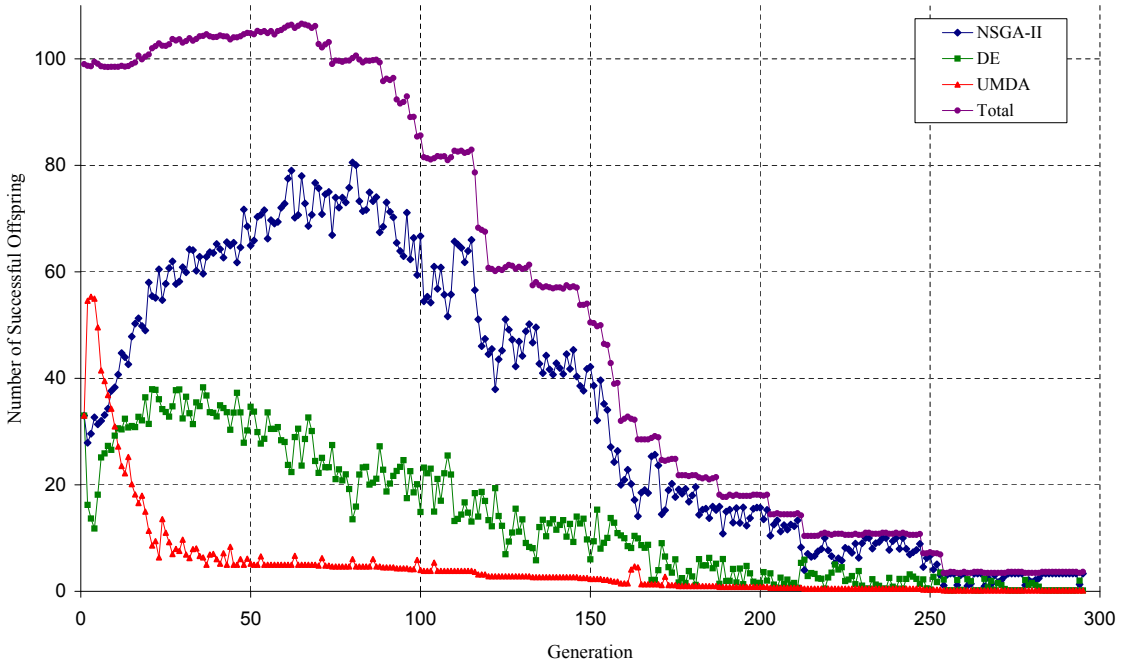


Figure 7.12: *Average offspring per sub-algorithm per generation in* $AMALGAM_{\text{ndu}}$ *for the HANOI benchmark.*

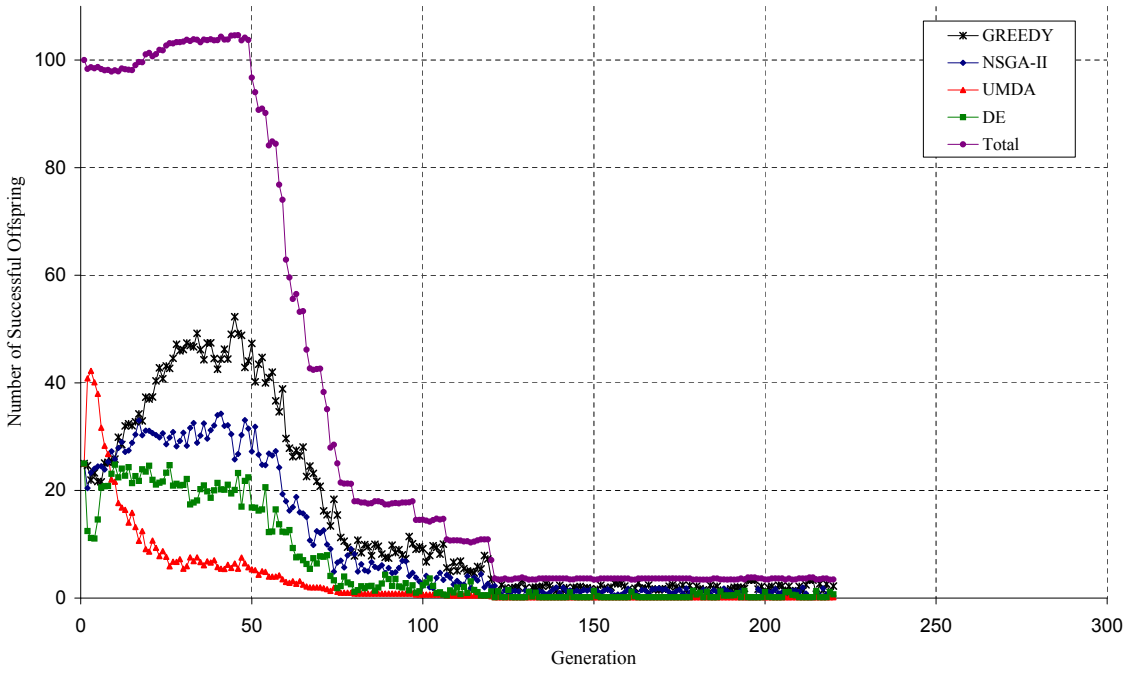


Figure 7.13: *Average number of successful offspring per sub-algorithm per generation in* $AMALGAM_{\text{ndug}}$ *for the HANOI benchmark.*

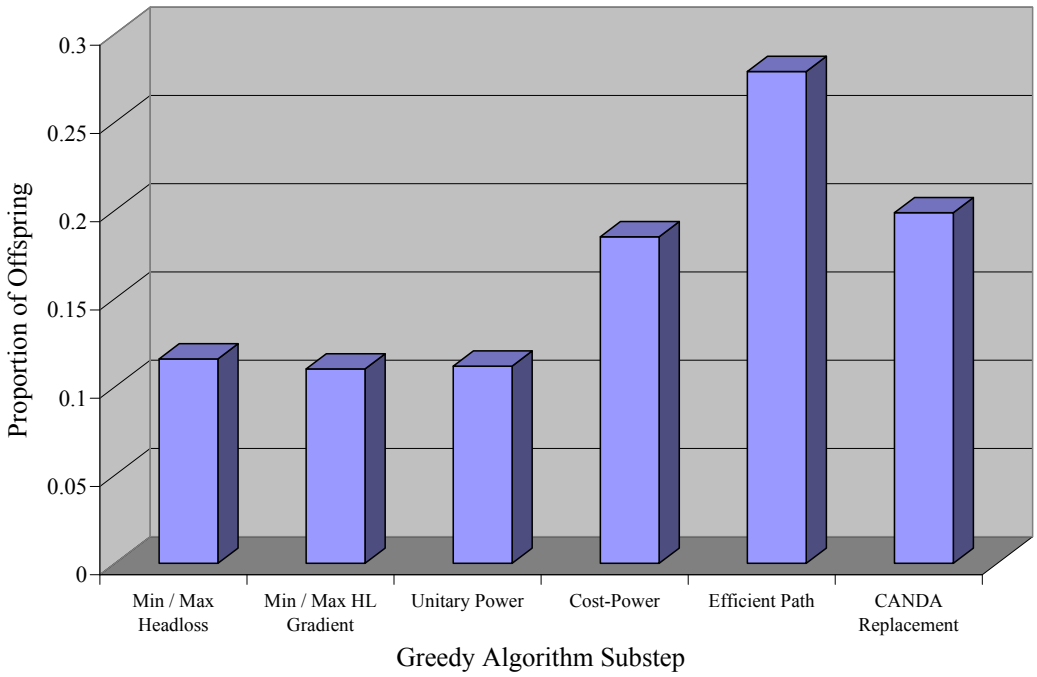


Figure 7.14: *Comparison of the proportions of accepted offspring generated by the heuristic substeps of the GREEDY algorithm for the TRP benchmark.*

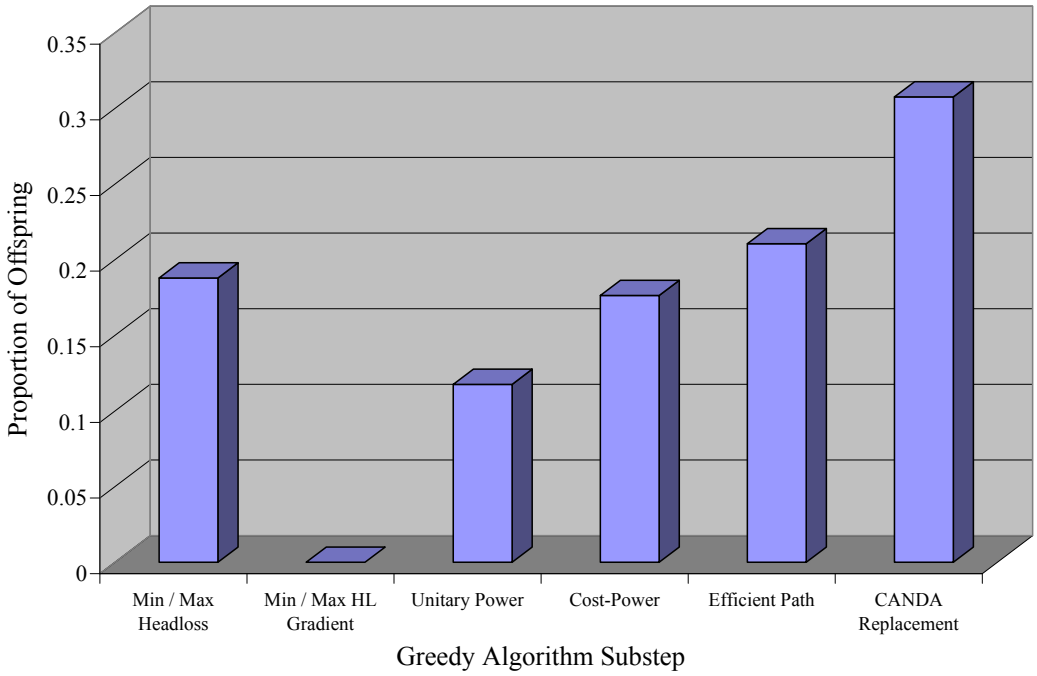


Figure 7.15: *Comparison of the proportions of accepted offspring generated by the heuristic substeps of the GREEDY algorithm for the TLN benchmark.*

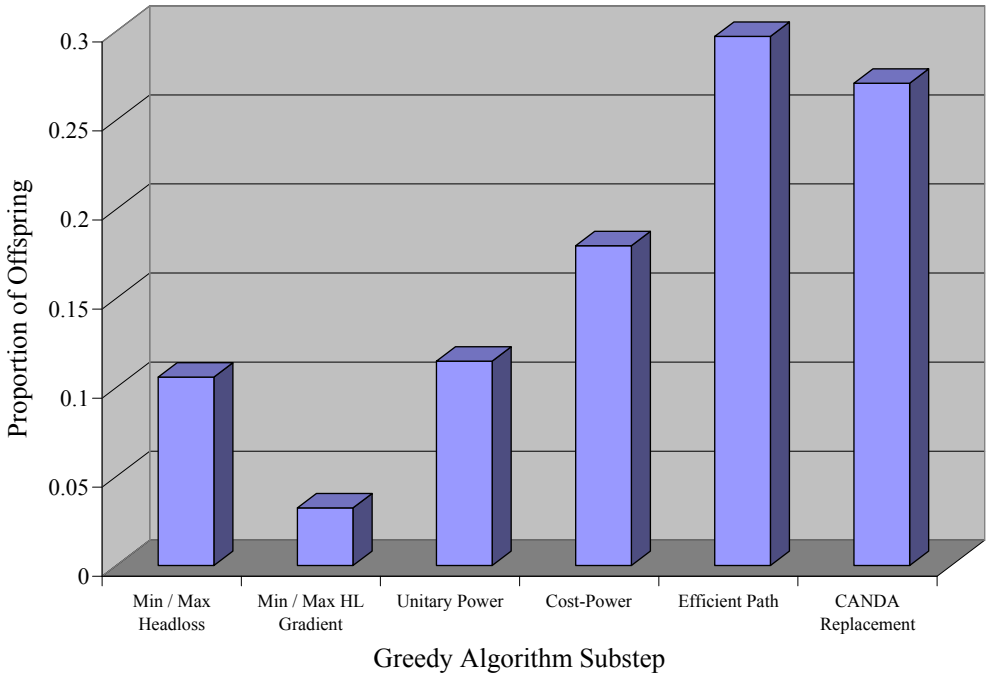


Figure 7.16: *Comparison of the proportions of accepted offspring generated by the heuristic substeps of the GREEDY algorithm for the HANOI benchmark.*

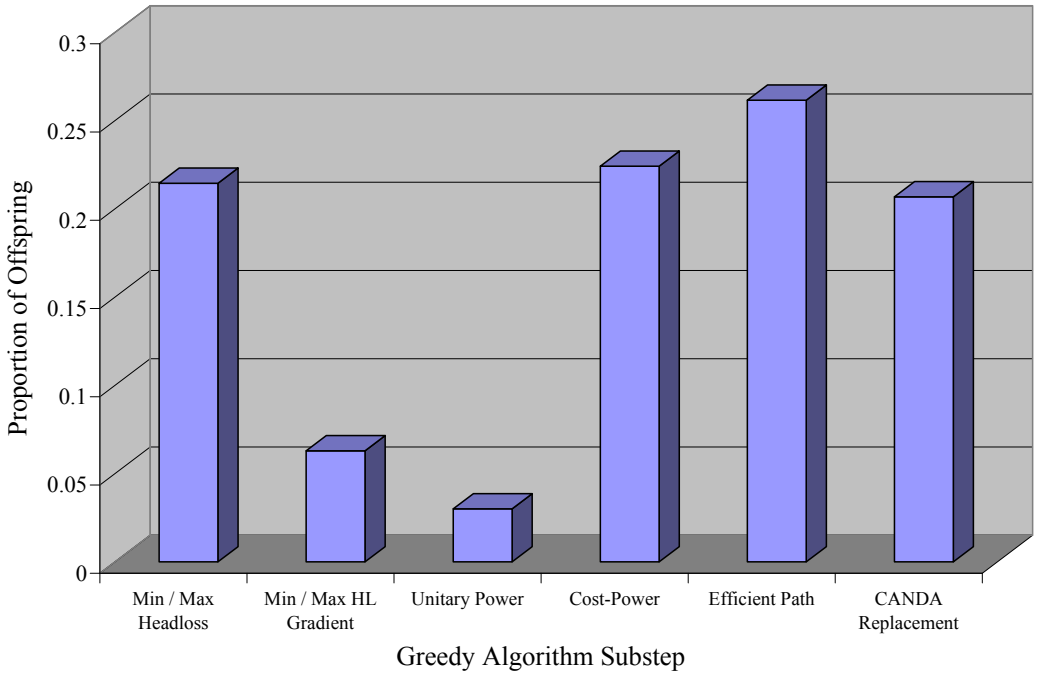


Figure 7.17: *Comparison of the proportions of accepted offspring generated by the heuristic substeps of the GREEDY algorithm for the NYTUN benchmark.*

7.2 Phase 2 Results: Constraint Handling Scheme Comparison

In order to analyze constraint handling schemes, NSGA-II was implemented and tested using the two different constraint handling techniques discussed previously, namely the penalty method (NSGA-II) and the constrained domination method (NSGA-II-CD). The same time limits and hypervolume reference points were used to compute average performance metrics over thirty optimisation runs for each algorithm. The summarized test results for the eight benchmarks TRP, TLN, NYTUN, HANOI, BLACK, FOSS, PESC, and MOD are shown in Table 7.21, including average and standard deviation of dominance rank, and average and standard deviation of hypervolume. For all but three WDS benchmarks, the algorithms achieved equivalent dominance ranks of 1, and NSGA-II-CD obtained slightly higher hypervolume values. However, for the HANOI benchmark, NSGA-II outclassed NSGA-II-CD with a dominance rank of 1.0667 compared to 1.6667, a dominance rank standard deviation of 0.2582 compared to 1.2910, and a large hypervolume difference of 0.9380 compared to 0.8619 for NSGA-II-CD. Then again, for the FOSS benchmark, NSGA-II-CD outperformed NSGA-II with a dominance rank of 1 compared to 1.3, and a slightly higher average hypervolume. The benchmark MOD, on the other hand, favoured NSGA-II in terms of both dominance rank standard deviation and hypervolume.

This stated, both algorithms were able to attain very similar Pareto-optimal fronts, particularly for the central portions of the Pareto-fronts, which may arguably be the most important region of the front, since it usually contains solutions with the best benefit-cost ratios. A sample attainment front comparison is provided for the HANOI benchmark in Figure 7.18, also showing the benefit-cost ratios (Network Resilience divided by cost) of the solutions multiplied by a scaling factor.

Although performance statistics were calculated only for feasible individuals in the final approximation sets, it was noticeable that all the individuals in the sets generated by NSGA-II-CD were feasible, whereas a large proportion (10–40%) of those generated by NSGA-II were infeasible. This illustrates a feature of the penalty method — improved flexibility comes at the price of having to process the results at the end in order to remove infeasible individuals. This, combined with the issue of having to specify a penalty factor in the first place, and the fact that NSGA-II-CD still performs reasonably well in comparison to NSGA-II, may well serve to recommend it as the method of choice for general use.

7.3 Chapter Summary

In this Chapter two phases of testing on WDS benchmarks from the literature were conducted in order (i) to compare twenty-three alternative metaheuristics for multi-objective WDSDO, in fulfilment of Dissertation Objectives 6 and 7 in §1.3, and (ii) to compare two different constraint handling techniques, namely the penalty term method and the constrained domination method, in fulfilment of Dissertation Objective 8. Finally, in order to conduct this analysis, the implementation of this optimisation model in a software library was completed in partial fulfilment of Dissertation Objective 10.

The metaheuristics compared in Phase 1 included ADMOEA, AMALGAM$_{\text{ndp}}$, AMALGAM$_{\text{ndu}}$, AMALGAM$_{\text{ndug}}$, AMALGAMI$_{\text{ndp}}$, AMALGAMI$_{\text{ndu}}$, AMALGAMI$_{\text{ndug}}$, AMALGAMS$_{\text{ndp}}$, AMALGAMS$_{\text{ndu}}$, ANIMA, DE, GREEDY, NSGA-II, PSO, PUMDA, SPEA-II, TAMALGAM$_{\text{ndp}}$, TAMALGAM$_{\text{ndu}}$, TAMALGAM$_{\text{ndug}}$, TAMALGAMJ$_{\text{ndp}}$, TAMALGAMJ$_{\text{ndu}}$, TAMALGAMJ$_{\text{ndug}}$, and UMDA. The WDS benchmarks analyzed were TRP, TLN, NYTUN, HANOI, BLACK, FOSS, PESCA, MOD and EXNET. Thirty hypervolume convergence time trials were

Benchmark	Algorithm	Avg d_R	SD d_R	Avg HV	SD HV
TRP	NSGA-II	1	0	0.9896	0.0014
	NSGA-II-CD	1	0	0.9901	0.0016
TLN	NSGA-II	1	0	0.9796	0.0129
	NSGA-II-CD	1	0	0.9816	0.0134
NYTUN	NSGA-II	1	0	0.9598	0.0054
	NSGA-II-CD	1	0	0.9624	0.0061
HANOI	NSGA-II	1.0667	0.2582	0.9380	0.0909
	NSGA-II-CD	1.6667	1.2910	0.8619	0.0382
BLACK	NSGA-II	1	0	0.9754	0.0053
	NSGA-II-CD	1	0	0.9857	0.0026
FOSS	NSGA-II	1.3	1.6432	0.9328	0.0134
	NSGA-II-CD	1	0	0.9477	0.0129
PESC	NSGA-II	1	0	0.9402	0.0117
	NSGA-II-CD	1	0	0.9406	0.0116
MOD	NSGA-II	1.1667	0.4611	0.9214	0.0180
	NSGA-II-CD	1.1667	0.7466	0.9189	0.0159

Table 7.21: *Performance metrics for NSGA-II and NSGA-II-CD computed for eight benchmarks (TRP, TLN, HANOI, NYTUN, BLACK, FOSS, PESC and MOD), averaged over thirty optimisation runs.*

conducted for each algorithm-benchmark pair in order to compare algorithmic efficiency. Thirty fair time trials were then used in the optimisation process (with time limits computed using the 90^{th} percentile of average convergence times) in order to compare the solution quality produced by the various algorithms.

The fastest algorithm in terms of convergence was ADMOEA, which was 0.5279 of the global average convergence time, yielding an average NHV of 0.9063, but it fared relatively poorly in terms of dominance rank and standard deviations of all metrics, indicating that it is one of the less stable algorithms. Other non-dominated algorithms in terms of convergence time and average hypervolume were $\text{AMALGAM}_{\text{ndp}}$, $\text{AMALGAMI}_{\text{ndp}}$, $\text{AMALGAM}_{\text{ndu}}$, with the latter producing the highest average normalized hypervolume of 0.9502 in a convergence time 0.8218 of the global average. The algorithm with the best performance in terms of dominance rank was $\text{TAMALGAM}_{\text{ndu}}$ with a value of 1.78, and with decent averages for NHV and NT of 0.9345 and 0.8754, respectively. Given the fact that it also achieved the lowest standard deviation for dominance rank of 1.82, it may be the preferred algorithm amongst these for time critical WDSDO. The slowest running algorithm was SPEA-II, with a convergence time 1.7043 of the global average. However, it exhibited one of the most stable performances with the smallest standard deviation for NHV of 0.0096.

The top four performing algorithms in the full-length time trials were NSGA-II, $\text{TAMALGAM}_{\text{ndu}}$, $\text{TAMALGAMJ}_{\text{ndu}}$ and $\text{AMALGAMS}_{\text{ndp}}$, and were mutually non-dominated with respect to each other for the various performance metrics. NSGA-II was the best algorithm in terms of average dominance rank with a value of 1.1. $\text{TAMALGAM}_{\text{ndu}}$ produced the best average position rank value of 5.125 and the lowest standard deviation for NHV of 0.26. The best algorithm in terms of average NHV was $\text{AMALGAMS}_{\text{ndp}}$ with a value of 0.9512. The GREEDY algorithm exhibited the worst performance overall, with an average dominance rank of 410.64. This demonstrates that despite its functioning as a powerful local search, the GREEDY algorithm which mimics engineering judgement cannot compete with modern metaheuristics, which employ advanced (intelligent) strategies in order to uncover better solutions with less computational effort.

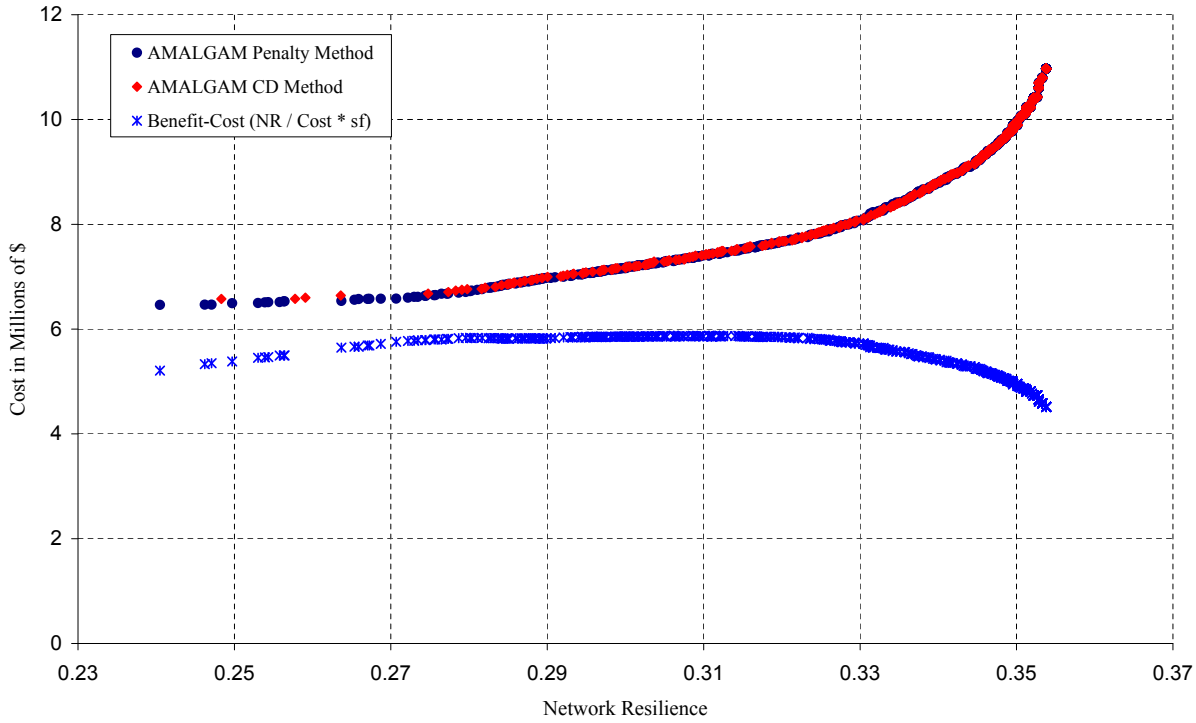


Figure 7.18: *Attainment comparison of NSGA-II using the penalty method and NSGA-II-CD using the constrained domination (CD) method for the HANOI benchmark, showing benefit-cost ratios.*

The four best algorithms from the full time trials were executed for the very large EXNET benchmark for thirty 5.4-hour runs each. AMALGAMS$_{\text{ndp}}$ proved the best algorithm on the whole, finding a much broader section of the Pareto-front than the other algorithms, although it was outperformed in the high Network Resilience region by TAMALGAM$_{\text{ndu}}$, and failed to locate Pareto-optimal solutions of very high values of Network Resilience. In light of the evidence presented, AMALGAMS$_{\text{ndp}}$ would seem to be the best algorithm at providing a broad range of solutions for difficult WDSDO problems, and TAMALGAM$_{\text{ndu}}$ appears to be one of the best choices for efficient, consistent performance and achieving solutions of high reliability.

The sub-algorithm analysis was conducted for two variants of the basic AMALGAM, namely AMALGAM$_{\text{ndu}}$ and AMALGAM$_{\text{ndug}}$, in order to demonstrate a fundamental shortcoming of the basic formulation. The variant including the GREEDY sub-algorithm exhibited a premature convergence phenomenon, caused by an excess of pressure to locate local optima. The basic AMALGAM formulation is unable to discriminate between mild improvements (successes), as delivered by GREEDY, and more significant ones generated by other metaheuristics, and consequently overly rewarded GREEDY.

The results from the constraint technique comparison of Phase 2 indicated that both the penalty term method and the constrained domination technique are viable options for general usage. Both alternatives are able to locate very similar Pareto-fronts along the region of highest benefit-cost, and the constrained domination technique may be the method of choice due to its generality and lack of requirement of user-specified parameters.

Chapter 8

Reliability Analysis

In this chapter, the results of the reliability analysis (Phase 3) are presented and discussed, examining the relationship between the *reliability surrogate measures* (RSMs) and two average demand satisfaction measures, namely a*verage demand satisfaction under uncertainty* (ADSU) and *average demand satisfaction under pipe failure* (ADSF). These measures correspond to two different types of analysis, where ADSU is calculated under conditions of stochastic demand, and ADSF is calculated across the simulation of all single-pipe failures. The RSMs analysed include the *Reliability Index* (RI), *Network Resilience* (NR), *Flow Entropy* (FE), and the novel *Mixed Surrogate* (MS) measure presented in §4.3.2. For the sake of convenience, the RSMs are sometimes referred to in the text as though they were algorithms themselves. For instance, where it is stated that Network Resilience achieved the best average ADSU, it is meant that the method utilising Network Resilience as its RSM during the bi-objective optimisation process achieved this honour. The optimisation algorithm employed in this Phase was NSGA-II. The two types of reliability analysis were conducted for the eight WDS benchmarks TRP, TLN, NYTUN, HANOI, BLACK, FOSS, PESC and MOD.

8.1 Probabilistic Reliability Simulation

Probabilistic reliability analysis under uncertain (stochastic) demand conditions was conducted in order to compare the attainment sets produced by the different RSMs. The full version of MCS for uncertain demands was not attempted during probabilistic reliability analysis, due to the associated high computational expense. Instead, 1 000 LHS samplings were conducted on final approximation set designs and used to estimate the ADSU. Gaussian PDFs were fitted to each node, with the mean taken as the specified peak nodal demand loading condition for a benchmark, and the standard deviation taken as 30% of this value. Furthermore, the nodal demand samples were rearranged using the method of Iman and Conover [128] to induce a rank correlation of 0.5 amongst the nodes for each demand loading condition (similar to the work by Kapelan *et al.* [141]), producing a set of 1 000 demand loading conditions $\mathcal{D}$.

Pressure-driven analysis was conducted by means of a demand adaptation method. The nodal demands $d_0^0, \ldots, d_n^0$, were used for an initial DDA hydraulic simulation at time $t = 0$ in order to compute the system pressures. Several iterations of DDA then followed, where demand is adapted until the required minimum pressures are achieved. During each iteration for which the nodal head was below the minimum required to satisfy demand, one of two steps were performed. If the pressure at node i was below zero, then the demand at that node was

adapted as $d_i^{t+1} = 0.99 \times d_i^t$. Otherwise, the demand at node i was adapted by taking the average of the previous iterations' demand and the actual demand achieved, using the equation $d_i^{t+1} = (d_i^t + d_i^t(\frac{h_i^t}{h_i^*})^{0.5})/2$, where h_i^t is the nodal pressure head at time t and h_i^* is the minimum pressure head required at node i, and where the second term of the numerator is the actual flow achieved in accordance with the pressure-dependant demand formula of Aoki [11]. Alternatively, if the pressure head h_i^t was in excess of the required minimum, then the demand was adapted as $d_i^{t+1} = \min(1.01 \times d_i^t, d_0^t)$. This process continued until the minimum pressures were satisfied to within 0.01% of the required minimum. One may then proceed to calculate the difference between the required initial demand and the final adapted demand, such that the demand satisfaction measure ADSU is $\text{AVG}_{\mathcal{D}}(\sum_{i=1}^{n} q_i / \sum_{i=1}^{n} d_i)$, and this was calculated across all 1 000 demand loading conditions.

8.2 Failure Analysis

Failure analysis was conducted, considering all single-pipe failure events in order to compare the solutions produced by the different RSMs under failure conditions. Failure reliability was quantified using the average percentage of demand satisfied across all failure events (ADSF). No stochastic analysis of pipe failures was conducted (it was assumed that all pipe failures are equally likely)[1]. It is reasonable to expect a performance degradation under pipe failure conditions. The degree of such failure was quantified by $\text{ADSF} = \text{AVG}_{\mathcal{F}}(\sum_{i=1}^{n} q_i / \sum_{i=1}^{n} d_i)$, where $\mathcal{F}$ is the set of all single pipe failure conditions, under a peak demand loading condition. Once again, pressure-driven analysis was conducted using the same method of demand adaptation that was used in the probabilistic reliability simulation.

8.3 Phase 3 Results: Analysis of Reliability Surrogate Measures

In Phase 3, NSGA-II was executed for thirty optimisation runs using the same time limits and hypervolume reference points as for the full time trials in Phase 1. This was done for the eight benchmarks TRP, TLN, HANOI, NYTUN, BLACK, FOSS, PESC and MOD using the four RSMs described above. The output from multiple runs was combined to produce four attainment approximation sets for each benchmark-RSM combination, ensuring that each RSM was given a fair chance to produce a good attainment set under representative stochastic conditions.

Average demand satisfaction (ADS) measures were used to quantify performance under stochastic demand and single-pipe failure conditions. It is expected that RSMs and ADS measures should be positively correlated, since RSMs strive to embody some features of hydraulic fitness. Linear regression analyses were conducted to examine the correlation and statistical significance of the relationship between the RSMs and ADS measures, and to determine whether these relationships could be sufficiently described by a linear regression model. While the R^2 and *Significance F*-values are reported for each regression fit of RSM *vs* ADSU and ADSF, the intercept and gradients are not provided, since the different RSMs are incommensurate and therefore cannot be compared in this manner. It may be observed that a linear model for the RSM-ADSU relationship is not necessarily ideal, since ADSU values reach a maximum of 1, creating a plateau in RSM-ADSU space, and this compounded by the fact that hydraulic interactions are highly nonlinear. The linear model was therefore treated in a piecewise fashion, by

[1]Size/age-dependent pipe failure probabilities are not considered here. This is a topic for future work.

first sorting solutions in order of increasing RSM-values, and then including only solutions up to the first solution with an ADSU value of 1. This problem was not encountered for ADSF.

The attainment sets for each RSM are displayed graphically in RSM-Cost space, showing the associated ADSU and ADSF percentages for those solutions, in order to demonstrate the positive correlation between the RSM and ADS values graphically. Then, in order to compare the different RSMs directly in terms of their ability to accommodate uncertain demands and pipe failures, all the attainment sets of the different RSMs are displayed in ADSF-Cost and ADSU-Cost space. The solutions of highest ADS/Cost (highest cost-benefit) are indicated by means of a large diamond, which also accompanies the name of the RSM/s in the legend for which that solution is identified.

Several performance metrics were calculated for each RSM attainment set, including the average and maximum ADSU, the average and maximum ADSF, and the highest ADSF/cost and ADSU/cost benefits. Furthermore, the results of the linear regression (analysis of variance (ANOVA) at a 95% level of significance) between the RSM and ADS measures were reported in terms of R^2 and Significance F-values, showing the goodness-of-fit and statistical significance of the regression model respectively.

Network characteristics were also reported on for the different RSM attainment sets, including the average cost, the average number of non-zero diameter sizable pipes, the average *summed diameter differences* (SDD)[2], the average *sum of squared diameter differences* (SQDD)[3], and the average *source share deviation from the mean* (SSDM)[4]. The SDD metric provides a quantitative measure of the degree of adjacent diameter disparity, which is generally undesirable in excess since it requires expensive valves and often produces unreliable loops in the network. Lower SDD values are therefore preferable. The SQDD metric further penalizes larger diameter differences, which are unwieldy in practice. The SSDM metric provides a measure of how balanced the supply distribution is amongst the sources. It is less desirable to have excessive reliance on a single source. A perfectly distributed supply will yield an SSDM of zero. In practice there is often a primary source which supplies most of the demand, but SSDM can still provide a relative measure of distribution equality.

8.3.1 TRP Reliability Analysis

The reliability analysis results of the four RSMs for the TRP benchmark are shown in Table 8.1. The solutions generated using the Mixed Surrogate obtained the highest average ADSU and ADSF values of 0.9981 and 0.9860, respectively. The solution with the maximum ADSU value of 0.9990 was located by all the algorithms, except Flow Entropy. The solution of maximum ADSF, with a value of 0.9914, was located exclusively by the Mixed Surrogate measure. The solution of highest cost-benefit in terms of both ADSU (6.468×10^{-7}) and ADSF (6.344×10^{-7}) was located using the Resilience Index and the Mixed Surrogate.

The regression analysis for TRP indicated that there was a statistically significant relationship between all RSMs and their ADS counterparts, with Significance F-values less than 0.05. The Flow Entropy measure demonstrated the highest R^2-values of 0.9736 with respect to ADSU, and 0.9753 with respect to ADSF. The lowest Significance F-values were generated by the Mixed Surrogate measure.

[2]If adjacent pipes have diameter differences, then this difference in size is added to the total.

[3]If adjacent pipes have diameter differences, then the squared difference is added to the total.

[4]The proportion of water supply provided by each reservoir is calculated, then the mean share is calculated, and finally the share deviations from the mean are summed.

The additional performance results are shown in Table 8.2. The Mixed Surrogate located by far the most solutions at 126, and the Flow Entropy measure found the fewest, at 7. Network Resilience produced the solutions with the highest average cost of 2.29×10^6, and Flow Entropy the lowest average cost of 1.71×10^6. All RSMs used an average of 11 pipes, which constitutes all the pipes available for sizing. Flow Entropy demonstrated the lowest average SDD and SQDD-values of 1 170 and 103 045, respectively. The Resilience Index obtained the best SSDM value of 0.2353, compared to the worst value of 0.2973 obtained by Flow Entropy.

Graphs of the TRP attainment fronts for the various RSMs, along with their corresponding ADSU and ADSF values indicated on the secondary vertical axis, are shown in Figures 8.1–8.4. While the shapes of the Pareto-fronts are quite different for the various RSMs, it is clear that there is a definite positive correlation between the RSM and ADS-values for each RSM.

The RSMs are compared directly in cost-ADSU-space for TRP in Figure 8.5. All of the RSMs were able to produce solutions along the Pareto-front, although Flow Entropy failed to attain representation in the high ADSU region of the front. The RSMs are compared in cost-ADSF-space for TRP in Figure 8.6. In this case there is substantial variation in performance, with the Mixed Surrogate producing the foremost front, particularly in the high ADSF region of the curve, followed by Network Resilience with a secondary front. The Resilience Index attainment is much worse than that of its competitors, failing to locate any non-dominated solutions whatsoever, while Flow Entropy locates some such solutions in the lower ADSF region of the Pareto-front.

RSM	**Avg ADSU**	**Mx ADSU**	**Mx ADSU/Cst**	**R^2 ADSU**	**Signif F**
Resilience Index	0.9968	0.9990	6.468×10^{-7}	0.9134	4.27×10^{-19}
Network Resilience	0.9967	0.9990	6.365×10^{-7}	0.9265	1.06×10^{-18}
Flow Entropy	0.9930	0.9961	6.346×10^{-7}	**0.9736**	3.89×10^{-5}
Mixed Surrogate	**0.9981**	0.9990	6.468×10^{-7}	0.8005	$\mathbf{3.12\times10^{-45}}$
RSM	**Avg ADSF**	**Mx ADSF**	**Mx ADSF/Cst**	**R^2 ADSF**	**Signif F**
Resilience Index	0.9789	0.9888	6.344×10^{-7}	0.8800	9.42×10^{-17}
Network Resilience	0.9788	0.9906	6.224×10^{-7}	0.9202	2.50×10^{-07}
Flow Entropy	0.9766	0.9818	6.232×10^{-7}	**0.9753**	3.28×10^{-05}
Mixed Surrogate	**0.9860**	**0.9914**	6.344×10^{-7}	0.8551	$\mathbf{7.58\times10^{-54}}$

Table 8.1*: Reliability comparisons of RSMs using ADS measures for the TRP benchmark.*

RSM	**Count**	**Avg Cost**	**Avg Pipes**	**Avg SDD**	**Avg SQDD**	**Avg SSDM**
Resilience Index	35	2.23×10^6	11	1 434	168 220	**0.2353**
Network Resilience	40	2.29×10^6	11	1 322	143 722	0.2476
Flow Entropy	7	1.71×10^6	11	**1 170**	**103 045**	0.2973
Mixed Surrogate	**126**	2.65×10^6	11	1 680	239 995	0.2607

Table 8.2*: Result comparisons of RSMs using WDS features for the TRP benchmark.*

8.3.2 TLN Reliability Analysis

The reliability analysis results of the four RSMs for the TLN benchmark appears in Table 8.3. The Reliability Index obtained the highest average ADSU of 0.9852. However, Network Resilience claimed the highest average ADSF of 0.7626. All methods excluding Flow Entropy located solutions of maximum ADSU 1.0. The solutions of highest cost-benefit in terms of

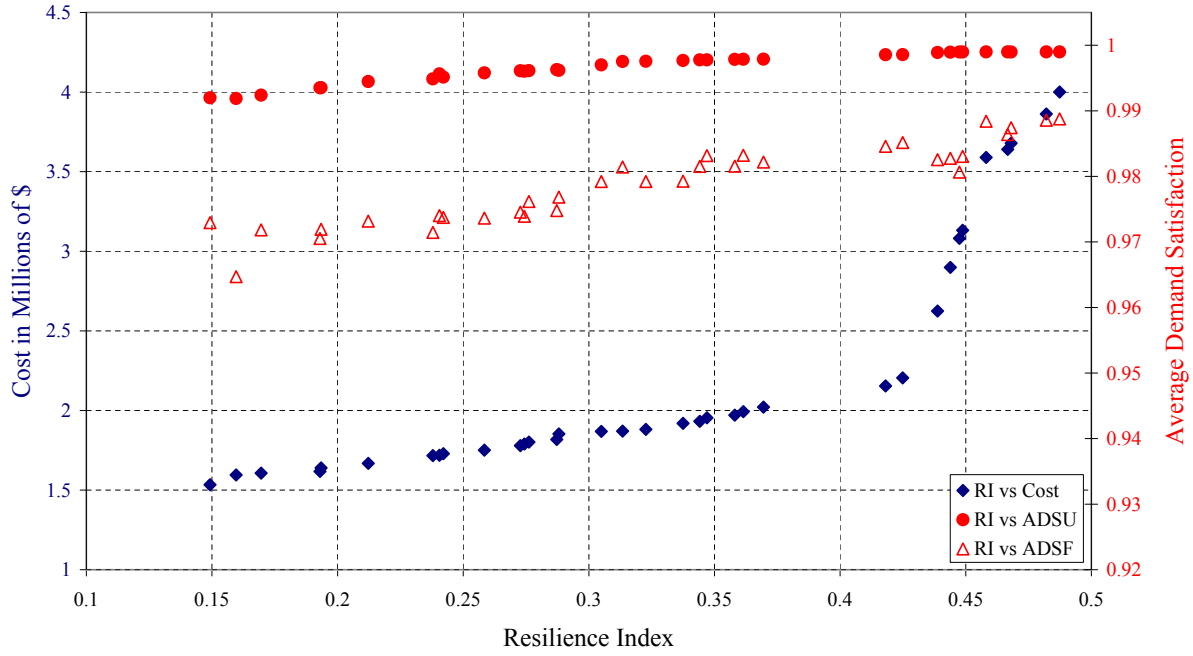


Figure 8.1: *Resilience Index versus ADSU (Uncertainty) and ADSF (Pipe Failure) for the TRP benchmark, with average demand satisfaction ratio on secondary axis.*

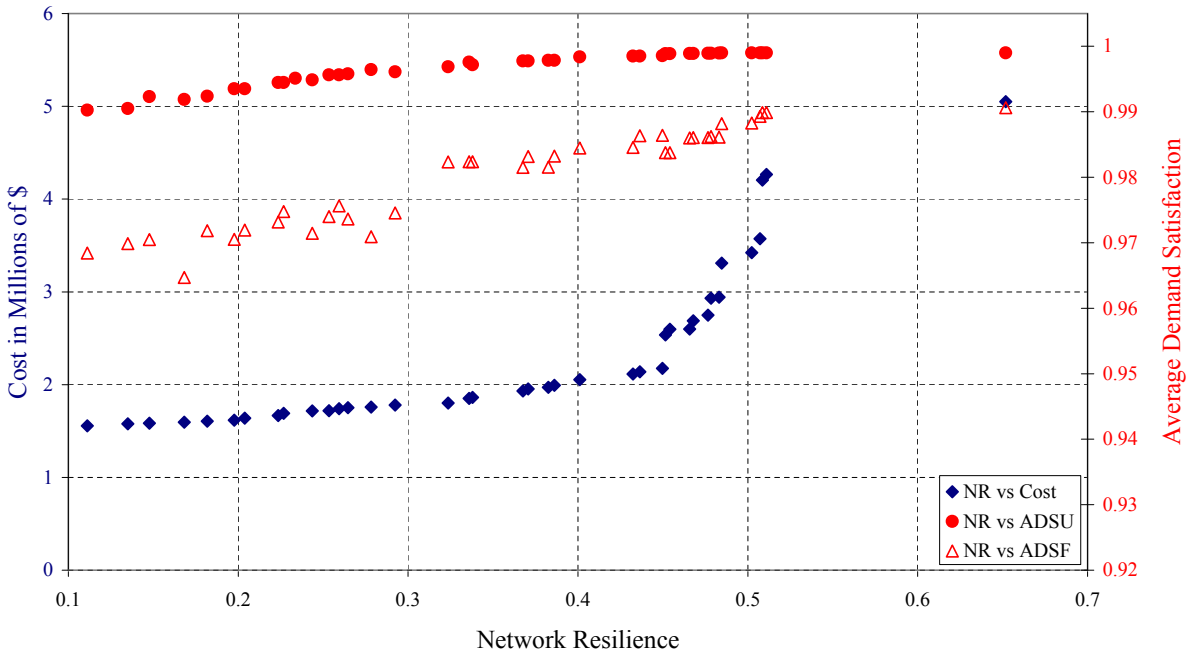


Figure 8.2: *Network Resilience versus ADSU (Uncertainty) and ADSF (Pipe Failure) for the TRP benchmark, with average demand satisfaction ratio on secondary axis.*

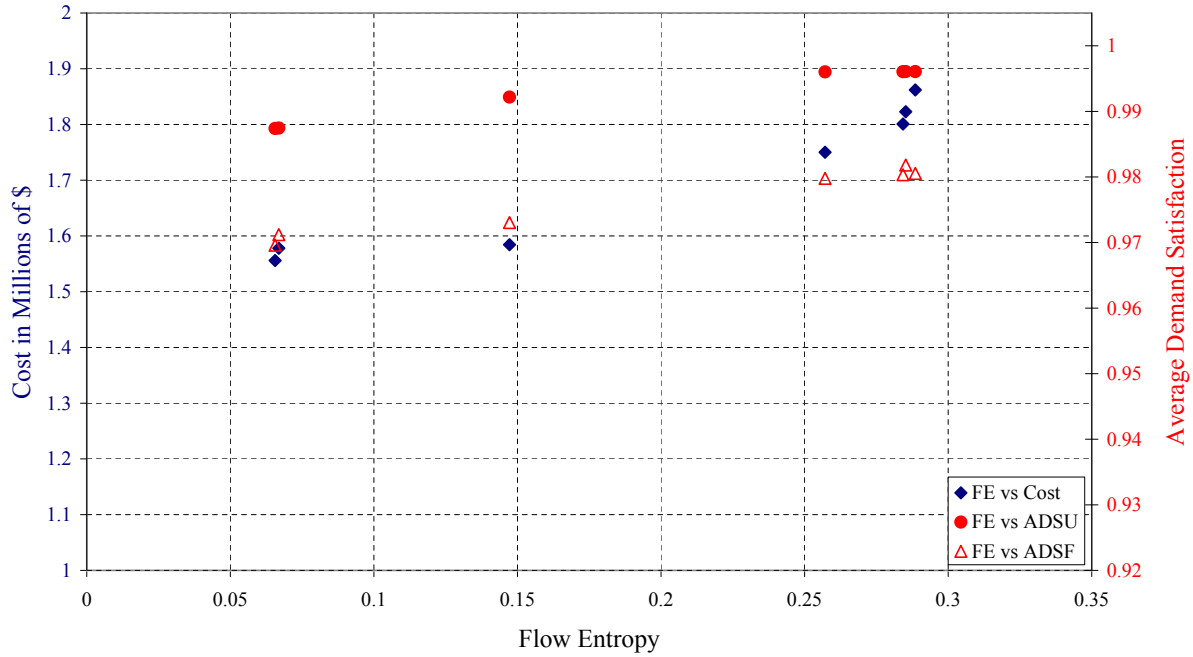


Figure 8.3: *Flow entropy versus ADSU (Uncertainty) and ADSF (Pipe Failure) for the TRP benchmark, with average demand satisfaction ratio on secondary axis.*

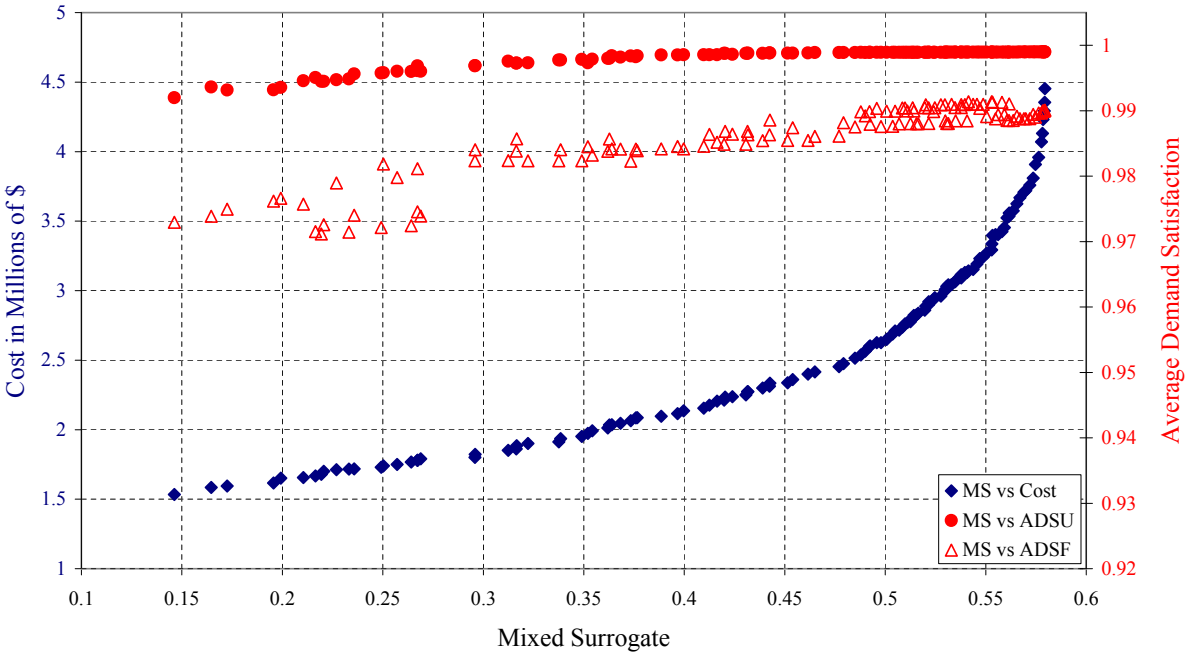


Figure 8.4: *Mixed surrogate versus ADSU (Uncertainty) and ADSF (Pipe Failure) for the TRP benchmark, with average demand satisfaction ratio on secondary axis.*

ADSU (2.26×10^{-6}) and ADSF (1.61×10^{-6}) were located by all methods, except the Mixed Surrogate.

The regression analysis for TLN indicated that there is a statistically significant relationship between all RSMs and their ADS counterparts, which had Significance F-values less than 0.05. The Network Resilience measure demonstrated the highest R^2-value of 0.9441 with respect to ADSU. The Reliability Index measure achieved the highest ADSF R^2-value of 0.6838. The lowest Significance F-values were 5.20×10^{-38} for ADSU and 5.96×10^{-27} for ADSF, generated by the Network Resilience and Mixed Surrogate, respectively.

The additional performance results are shown in Table 8.4. The Mixed Surrogate once again located far more solutions than the other methods, at 130, and the Flow Entropy measure found the fewest, at 38. The Mixed Surrogate produced the solutions with the highest average cost of 1.08×10^{6}, and Flow Entropy achieved the lowest average cost of 5.17×10^{5}. The largest average number of pipes used was 7.92 by Network Resilience, and the smallest average number was 7.74 by Resilience Index.

Network Resilience demonstrated the lowest average SDD and SQDD-values of 602 and 83 614, respectively, compared to the highest values attained by the Mixed Surrogate of 935 and 194 697. SSDM was not applicable to TLN since it has only a single water source.

Graphs of the TLN attainment fronts for the various RSMs, along with their corresponding ADSU and ADSF-values indicated on the secondary vertical axis, are shown in Figures 8.7–8.10. The positive correlation between the RSMs and the ADS measures is still clearly visible; however, it is not as strong as for TRP, particularly with respect to ADSF, which exhibits greater disorganisation in the regions of lower RSM values. The ADSF values of the Flow Entropy results rise and fall again with increasing FE.

The RSMs are compared directly in cost-ADSU-space for TLN in Figure 8.11. Only Network Resilience and Resilience Index were able to produce solutions along the full length of the Pareto-front. The Mixed Surrogate produced a secondary front which meets with the primary front for ADSU values less than 0.987 and again at ADSU values close to 1 (for cost values

RSM	**Avg ADSU**	**Mx ADSU**	**Mx ADSU/Cst**	**R^2 ADSU**	**Signif F**
Resilience Index	**0.9852**	1.0000	2.26×10^{-6}	0.9139	3.34×10^{-27}
Network Resilience	0.9831	1.0000	2.26×10^{-6}	**0.9441**	$\mathbf{5.20\times10^{-38}}$
Flow Entropy	0.9559	0.9750	2.26×10^{-6}	0.4764	1.63076×10^{-6}
Mixed Surrogate	0.9812	1.0000	2.25×10^{-6}	0.7874	5.04×10^{-33}
RSM	**Avg ADSF**	**Mx ADSF**	**Mx ADSF/Cst**	**R^2 ADSF**	**Signif F**
Resilience Index	0.6889	0.8750	1.61×10^{-6}	**0.6838**	1.22×10^{-16}
Network Resilience	**0.7626**	0.8750	1.61×10^{-6}	0.6279	6.75×10^{-17}
Flow Entropy	0.6730	0.7527	1.61×10^{-6}	0.4618	1.63×10^{-6}
Mixed Surrogate	0.7421	0.8546	1.58×10^{-6}	0.5959	$\mathbf{5.96\times10^{-27}}$

Table 8.3*: Reliability comparisons of RSMs using ADS measures for the TLN benchmark.*

RSM	**Count**	**Avg Cost**	**Avg Pipes**	**Avg SDD**	**Avg SQDD**	**Avg SSDM**
Resilience Index	62	7.51×10^{5}	7.74	880	171 299	N/A
Network Resilience	73	8.05×10^{5}	7.92	**602**	**83 614**	N/A
Flow Entropy	38	5.17×10^{5}	7.83	850	151 626	N/A
Mixed Surrogate	**130**	1.08×10^{6}	7.85	935	194 697	N/A

Table 8.4*: Result comparisons of RSMs using WDS features for the TLN benchmark.*

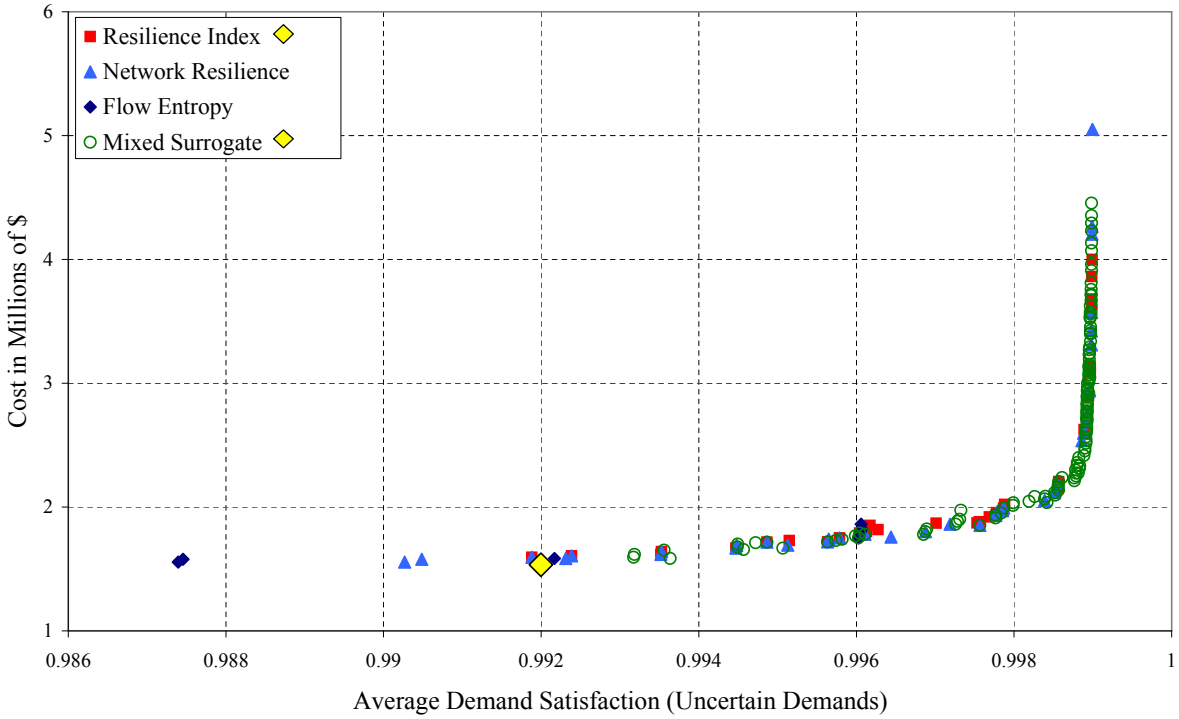


Figure 8.5: *Comparison of RSMs: ADSU vs Cost for the TRP benchmark.*

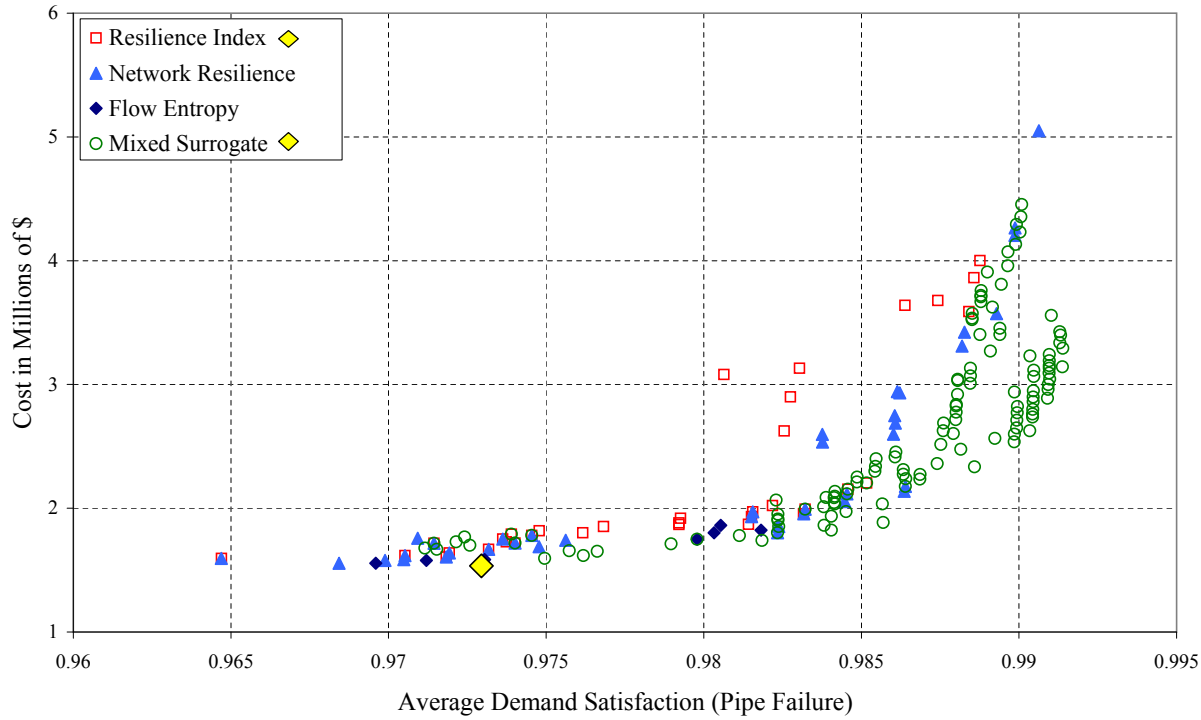


Figure 8.6: *Comparison of RSMs: ADSF vs Cost for the TRP benchmark.*

greater than $1 million). Flow Entropy found certain solutions in the lower region of the Pareto-front, but was unable to locate solutions of ADSU greater than 0.975. The RSMs are compared in cost-ADSF-space for TLN in Figure 8.12. Once again there is substantially more variation in performance than for ADSU, with Network Resilience producing the foremost front from ADSF 0.75 onwards, above which Flow Entropy was unable to locate solutions. Resilience Index seemed to take second position, particularly in the upper regions of ADSF. The Mixed Surrogate is next in line, producing a tertiary front. Flow Entropy managed to locate some solutions in the lower ADSF region of the Pareto-front.

8.3.3 HANOI Reliability Analysis

The reliability analysis results of the four RSMs for the HANOI benchmark are shown in Table 8.5. The Mixed Surrogate method obtained the highest average ADSU and ADSF-values of 0.9701 and 0.5514, respectively. The solution with the maximum ADSU-value of 0.9781 was only located using the Resilience Index, and the solution with the maximum ADSF-value of 0.7894 was uniquely found using Network Resilience. The solution of highest cost-benefit in terms of ADSU (1.55×10^{-7}) was located using Resilience Index, and the solution of highest cost-benefit in terms of ADSF (9.92×10^{-8}) was located using Network Resilience.

The regression analysis for HANOI indicated that there is a statistically significant relationship between all RSMs and their ADS counterparts, with Significance F-values less than 0.05, except for Flow Entropy with respect to ADSF, which has a Significance F-value of 0.0515. Upon further investigation it may be seen that Flow Entropy is actually negatively correlated with respect to the ADS measures for the HANOI benchmark. The Resilience Index achieved the highest R^2-value of 0.9577 with respect to ADSU, and Network Resilience achieved the highest R^2-value of 0.9156 with respect to ADSF. The lowest Significance F-values were generated by the Mixed Surrogate measure.

The additional performance results are shown in Table 8.6. The Mixed Surrogate located by far the most solutions, at 437, and Flow Entropy produced the fewest, at 6. The Mixed Surrogate produced the solutions with the highest average cost of 7.53×10^6, and Resilience Index produced the lowest average cost of 7.08×10^6. Flow Entropy produced solutions with the highest average number of pipes (33.50), and Resilience Index produced the lowest number of non-zero pipes (32.41). Network Resilience demonstrated the lowest average SDD and SQDD-values of 4 007 and 1 224 240, respectively — substantially less than the other RSMs. SSDM is not applicable to HANOI since it has only a single source.

Graphs of the HANOI attainment fronts for the various RSMs, along with their corresponding ADSU and ADSF-values indicated on the secondary vertical axis, may be found in Figures 8.13–8.16. Except for Flow Entropy, which actually has a negative correlation, all the other RSMs demonstrate a strong positive correlation between the RSM and ADS values for each RSM.

The RSMs are compared directly in cost-ADSU-space for HANOI in Figure 8.17. Flow Entropy failed to locate any non-dominated solutions, but the other RSMs are all close to the global Pareto-front and take turns claiming solutions along the front. In the low ADSU regions (less than 0.9650), Network Resilience and Resilience Index dominate. From ADSU 0.9650 onwards, the Mixed Surrogate and Reliability Index take turns locating the most non-dominated solutions. The RSMs are compared in cost-ADSF-space for HANOI in Figure 8.18. In this case there is substantial variation in performance, with the Network Resilience dominating convincingly in the upper ADSF regions, from approximately ADSF 0.64 onwards. Resilience Index and the

RSM	Avg ADSU	Mx ADSU	Mx ADSU/Cst	R^2 ADSU	Signif F
Resilience Index	0.9662	**0.9781**	**1.55**$\times 10^{-7}$	**0.9577**	1.93×10^{-163}
Network Resilience	0.9657	0.9768	1.54×10^{-7}	0.9309	5.00×10^{-86}
Flow Entropy	0.8989	0.9592	1.47×10^{-7}	0.7403	2.79×10^{-2}
Mixed Surrogate	**0.9701**	0.9775	1.51×10^{-7}	0.9028	$\mathbf{2.61\times 10^{-222}}$
RSM	**Avg ADSF**	**Mx ADSF**	**Mx ADSF/Cst**	**R^2 ADSF**	**Signif F**
Resilience Index	0.4064	0.7831	8.63×10^{-8}	0.9034	2.82×10^{-121}
Network Resilience	0.5211	**0.7894**	$\mathbf{9.92\times 10^{-8}}$	**0.9156**	1.01×10^{-79}
Flow Entropy	0.4729	0.5953	8.93×10^{-8}	0.6537	5.15×10^{-2}
Mixed Surrogate	**0.5514**	0.6407	8.76×10^{-8}	0.7538	$\mathbf{1.84\times 10^{-134}}$

Table 8.5*: Reliability comparisons of RSMs using ADS measures for the HANOI benchmark.*

RSM	Count	Avg Cost	Avg Pipes	Avg SDD	Avg SQDD	Avg SSDM
Resilience Index	237	7.08×10^6	32.41	5 054	1 664 303	0
Network Resilience	147	7.12×10^6	33.04	**4 007**	**1 224 240**	0
Flow Entropy	6	7.43×10^6	33.50	4 707	1 722 148	0
Mixed Surrogate	**437**	7.53×10^6	32.93	4 881	1 668 709	0

Table 8.6*: Result comparisons of RSMs using WDS features for the HANOI benchmark.*

Mixed Surrogate formed secondary fronts, with Resilience Index faring the worst. Flow Entropy managed to locate four non-dominated solutions between ADSF 0.5 and 0.6.

8.3.4 NYTUN Reliability Analysis

The reliability analysis results of the four RSMs for the NYTUN benchmark are shown in Table 8.7. The Resilience Index method obtained the highest average ADSU value of 0.9923. Resilience Index and Network Resilience shared the highest average ADSF of 0.9977. All of the RMSs, except Flow Entropy, were able to locate solutions of maximum ADSU 1.0. The solution with the maximum ADSF-value of 0.9985 was located by all the algorithms, except Flow Entropy. The solution of highest cost-benefit in terms of both ADSU (2.48×10^{-8}) and ADSF (2.54×10^{-8}) was located by Resilience Index.

The regression analysis for NYTUN indicated that there is a statistically significant relationship between all RSMs and their ADS counterparts, with Significance F-values less than 0.05. The Flow Entropy measure demonstrated the highest R^2-value of 0.9716 with respect to ADSU, and Network Resilience achieved the highest R^2-value of 0.9415 with respect to ADSF. The lowest Significance F-values were obtained by Network Resilience for ADSF and the Mixed Surrogate for ADSU.

The additional performance results are shown in Table 8.8. The number of solutions found was more evenly spread. The Mixed Surrogate located the most solutions, at 221, with a close second by Resilience Index, of 208. The fewest number of solutions found was 129, by Flow Entropy. Mixed Surrogate produced the solutions with the highest average cost of 1.19×10^8, and Resilience Index the lowest average cost of 8.86×10^7. Flow Entropy used the most pipes, with an average of 14.26. Resilience Index used the fewest pipes, with an average of 8.95. The Mixed Surrogate demonstrated the lowest average SDD and SQDD-values of 2 702 and 285 516, respectively, compared to the maximum values attained by Resilience Index, of 3 207

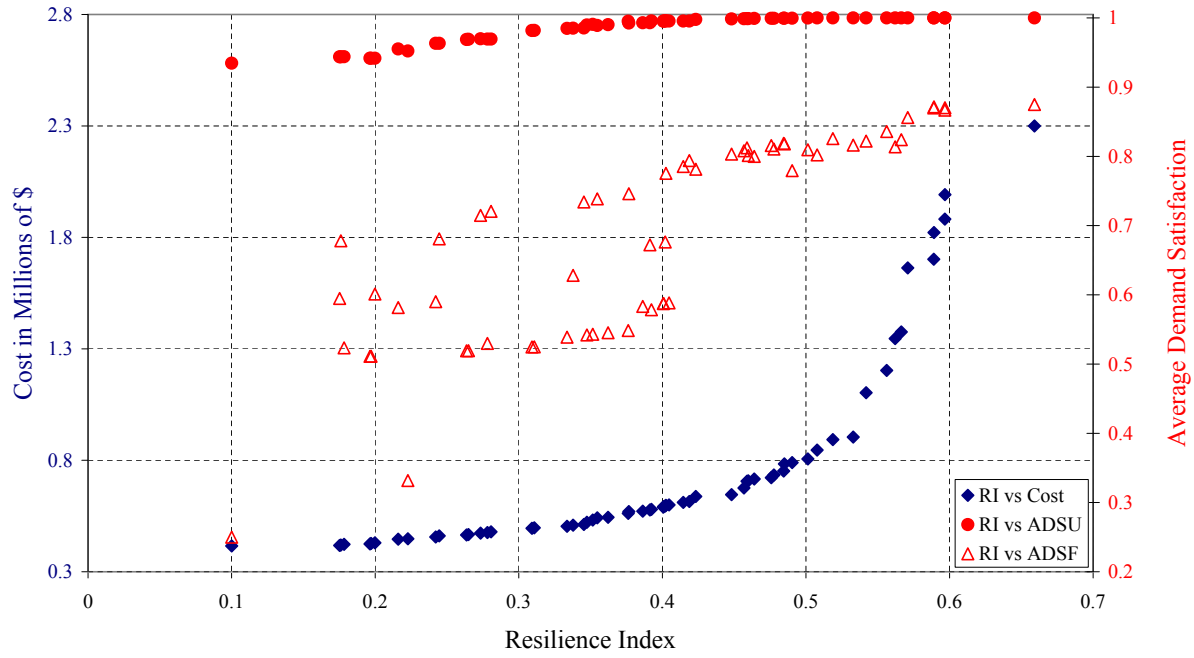


Figure 8.7: *Resilience Index versus ADSU (Uncertainty) and ADSF (Pipe Failure) for the TLN benchmark, with average demand satisfaction ratio on secondary axis.*

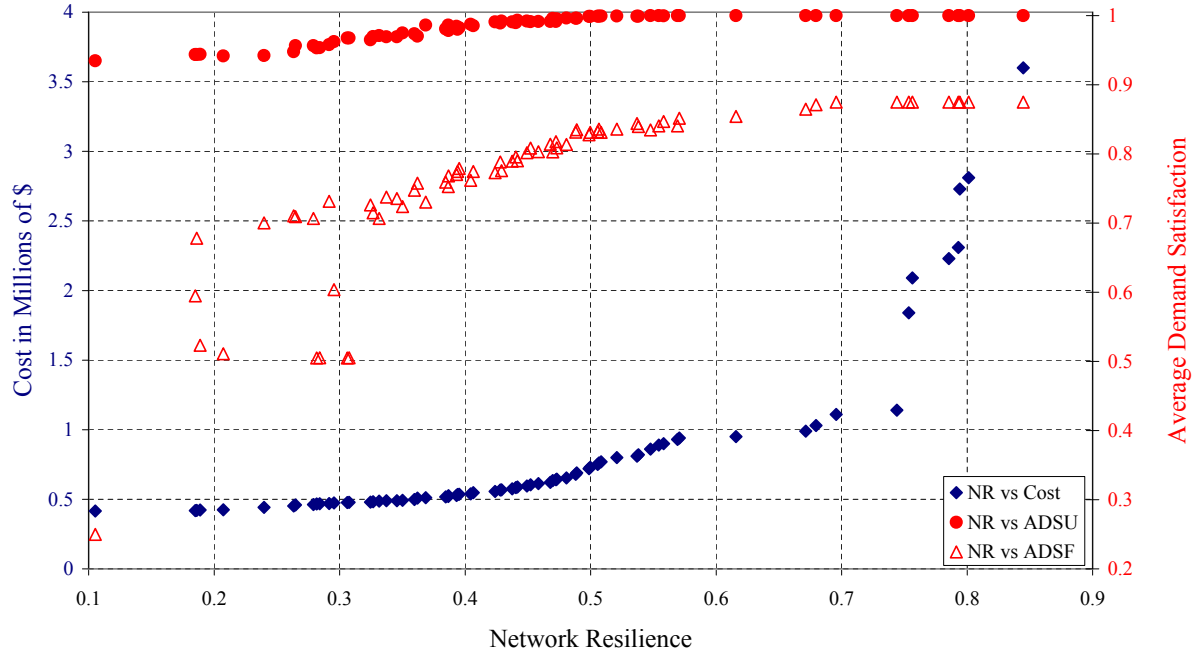


Figure 8.8: *Network Resilience versus ADSU (Uncertainty) and ADSF (Pipe Failure) for the TLN benchmark, with average demand satisfaction ratio on secondary axis.*

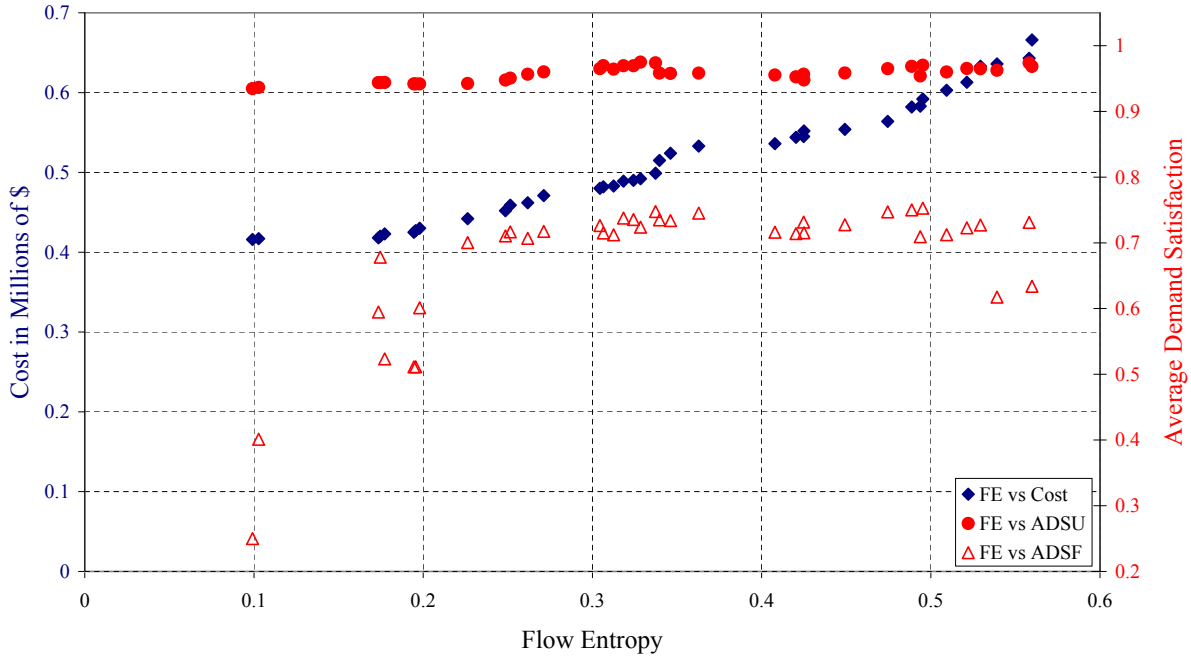


Figure 8.9: *Flow entropy versus ADSU (Uncertainty) and ADSF (Pipe Failure) for the TLN benchmark, with average demand satisfaction ratio on secondary axis.*

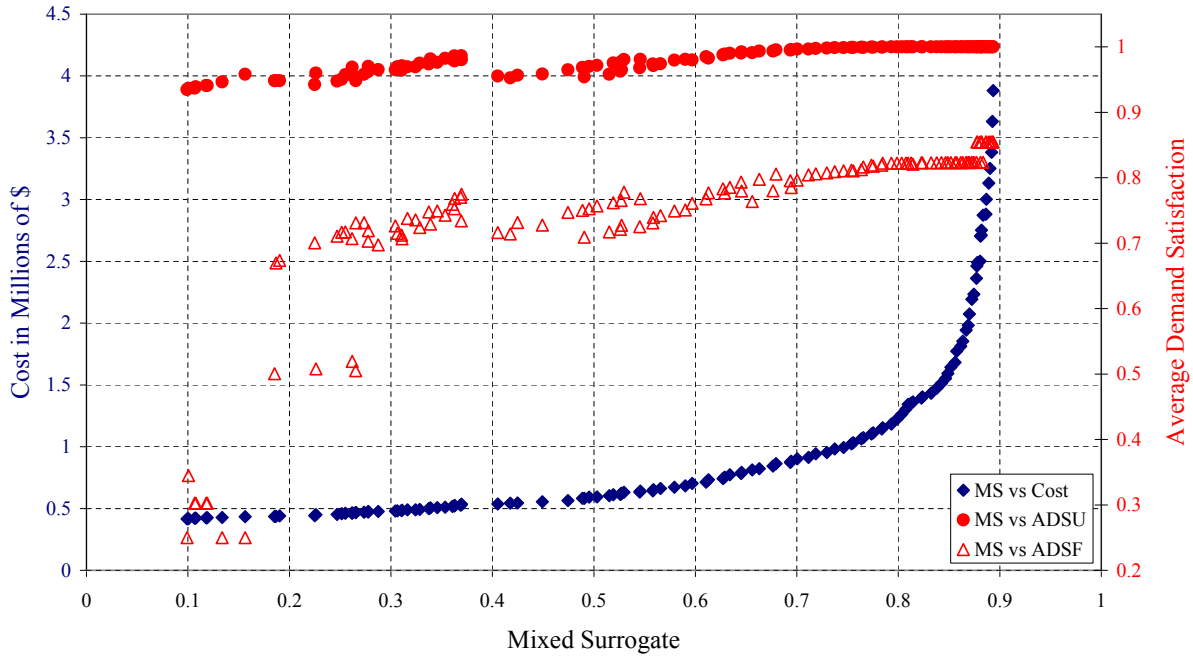


Figure 8.10: *Mixed surrogate versus ADSU (Uncertainty) and ADSF (Pipe Failure) for the TLN benchmark, with average demand satisfaction ratio on secondary axis.*

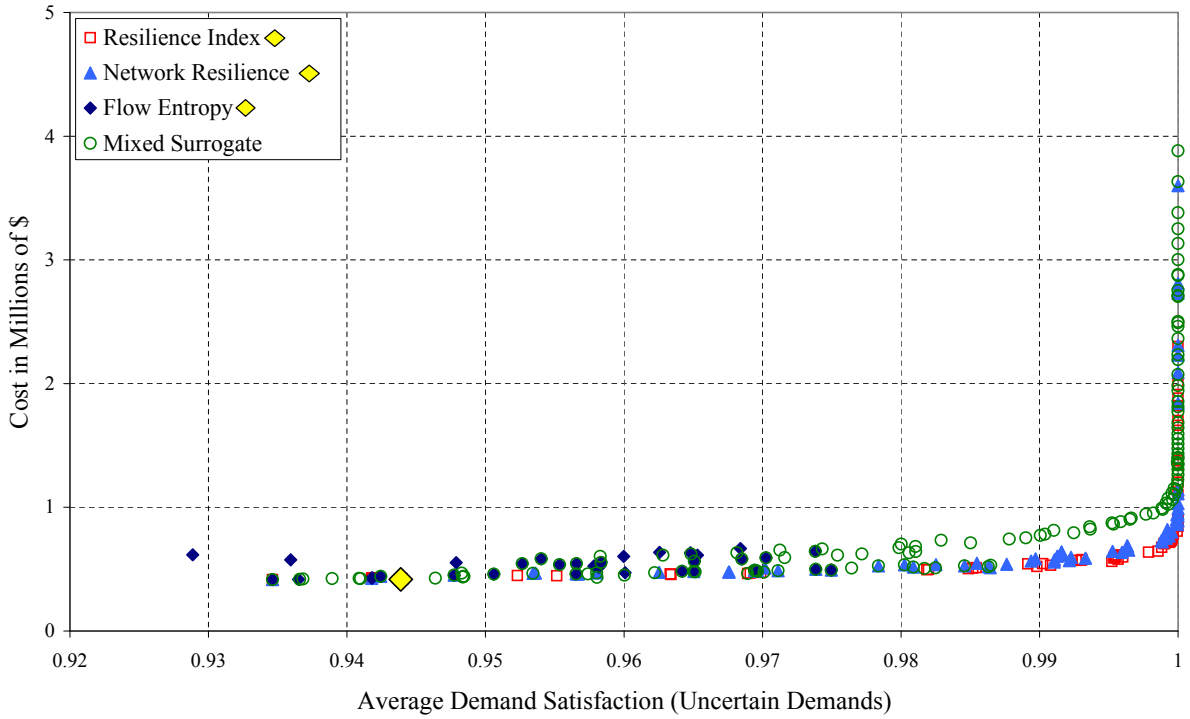


Figure 8.11: *Comparison of RSMs: ADSU vs Cost for the TLN benchmark.*

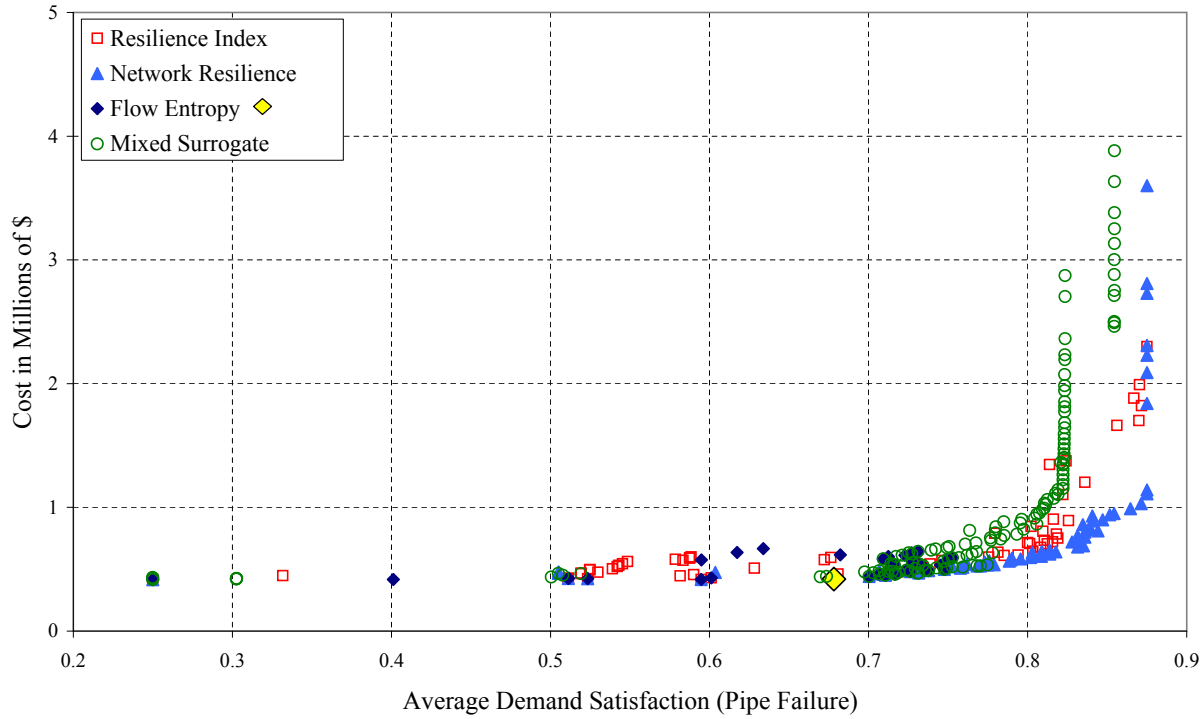


Figure 8.12: *Comparison of RSMs: ADSF vs Cost for the TLN benchmark.*

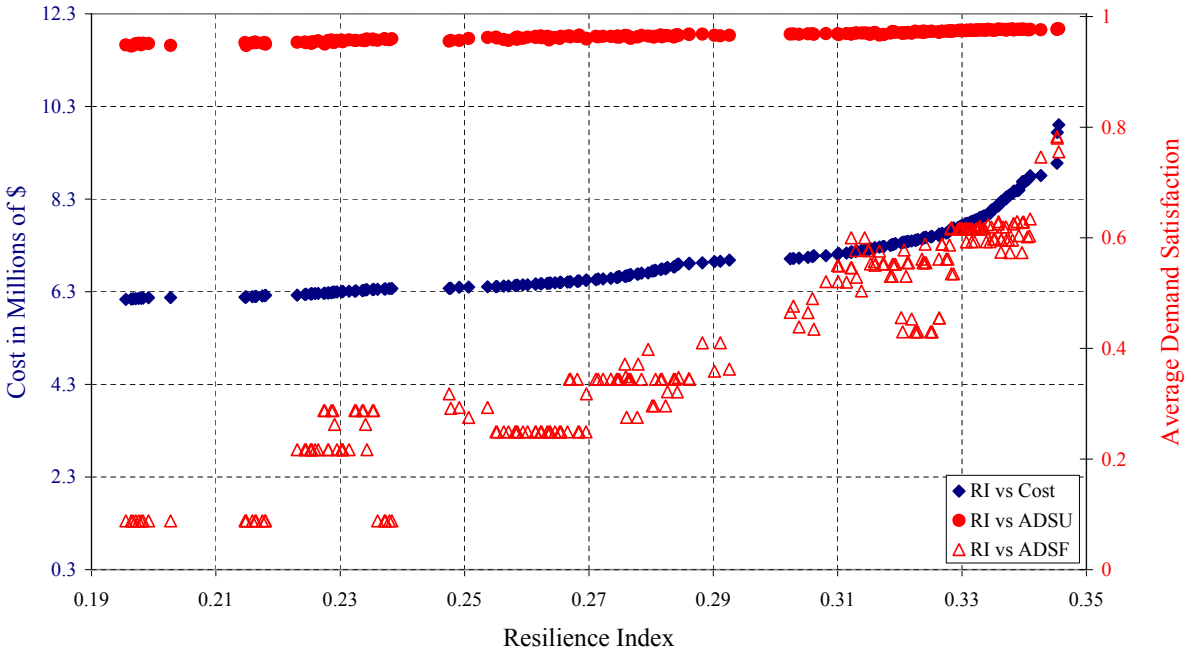


Figure 8.13: *Resilience Index versus ADSU (Uncertainty) and ADSF (Pipe Failure) for the HANOI benchmark, with average demand satisfaction ratio on secondary axis.*

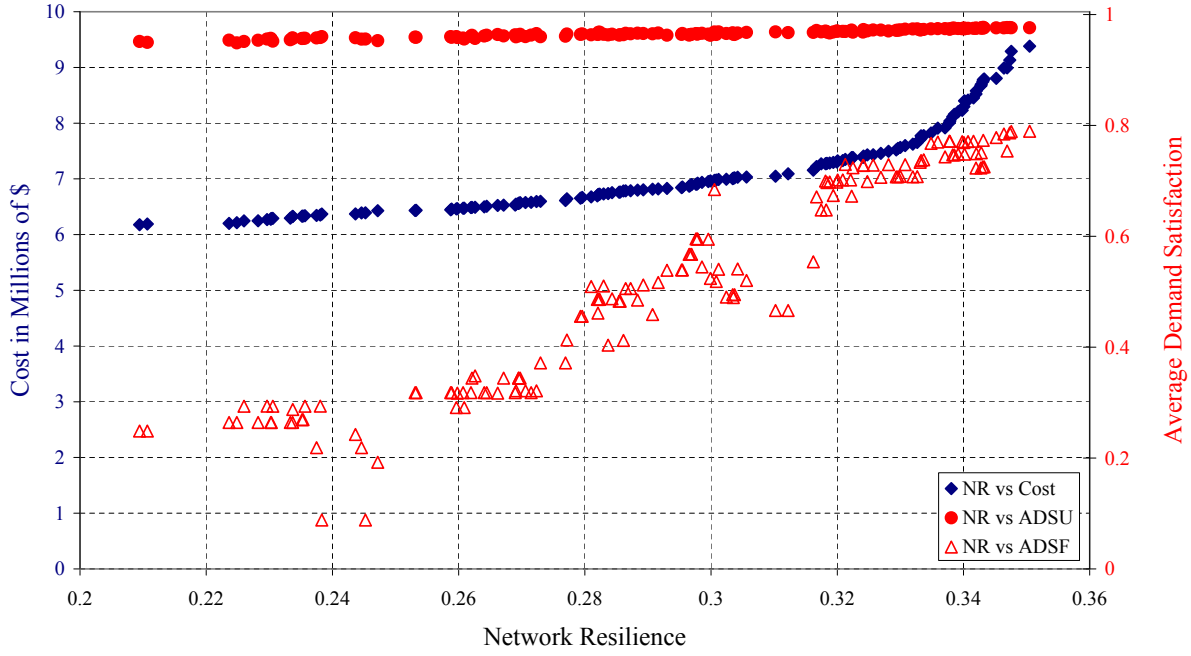


Figure 8.14: *Network Resilience versus ADSU (Uncertainty) and ADSF (Pipe Failure) for the HANOI benchmark, with average demand satisfaction ratio on secondary axis.*

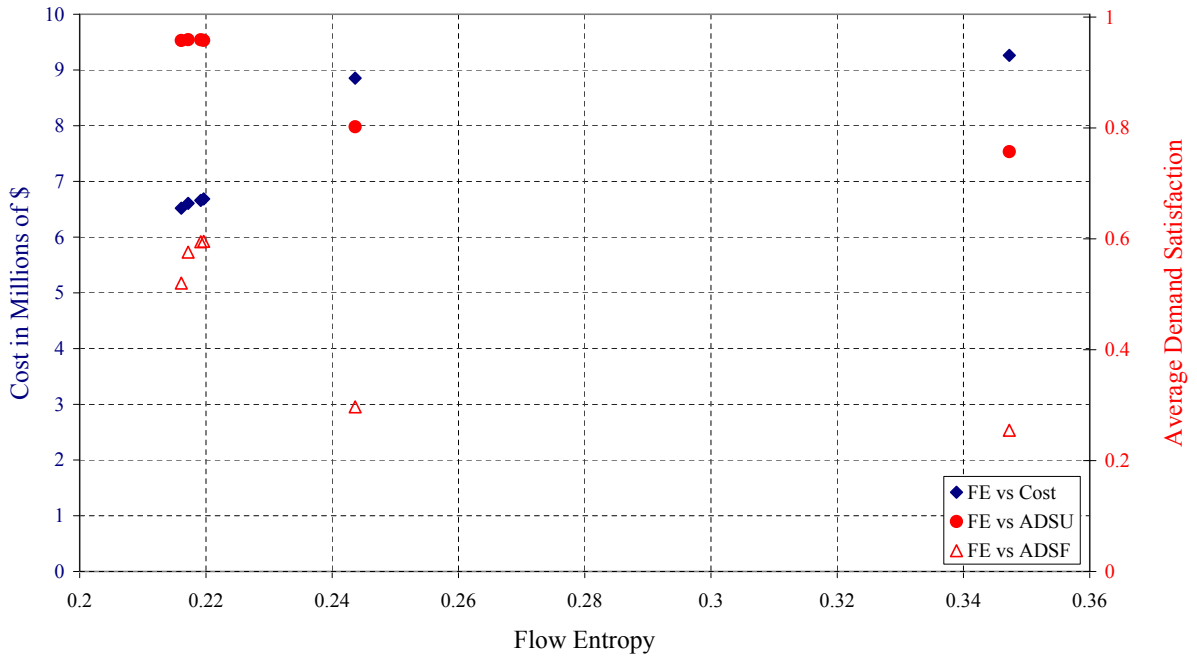


Figure 8.15: *Flow Entropy versus ADSU (Uncertainty) and ADSF (Pipe Failure) for the HANOI benchmark, with average demand satisfaction ratio on secondary axis.*

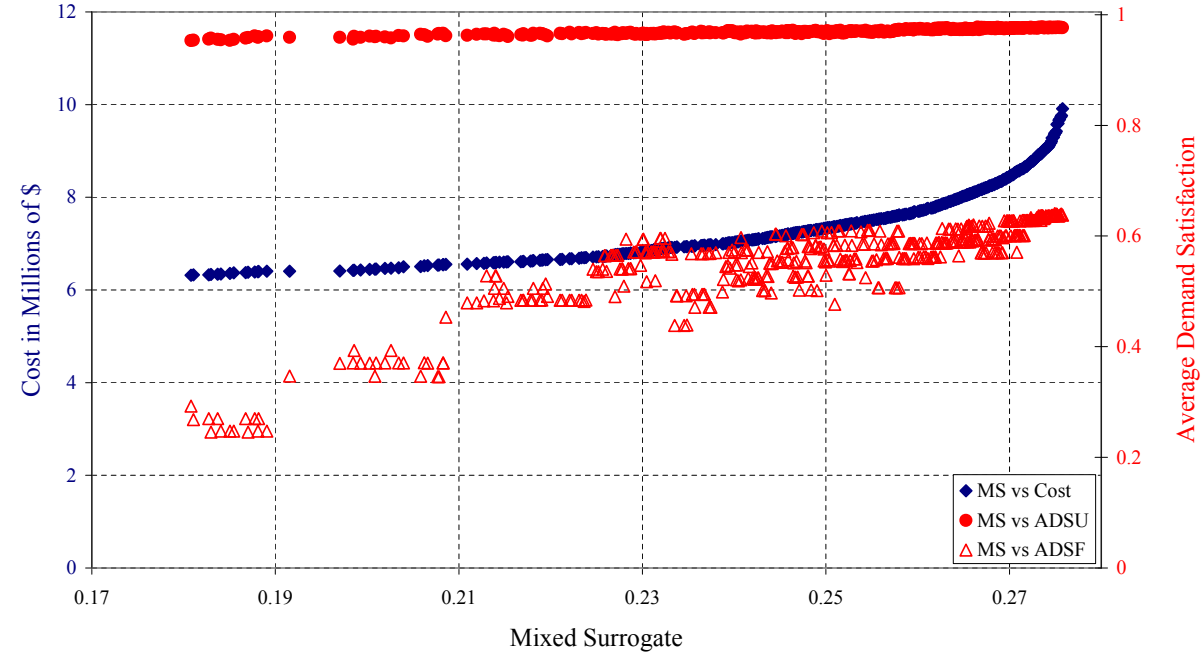


Figure 8.16: *Mixed surrogate versus ADSU (Uncertainty) and ADSF (Pipe Failure) for the HANOI benchmark, with average demand satisfaction ratio on secondary axis.*

and 404 987, respectively. SSDM is not applicable to NYTUN since it has only a single water source.

Graphs of the NYTUN attainment fronts for the various RSMs, along with their corresponding ADSU and ADSF-values indicated on the secondary vertical axis, may be found in Figures 8.19–8.22. It is clear that NYTUN is more sensitive to uncertain demands and less sensitive to pipe failures than the previous benchmarks, since the established pattern of ADSU and ADSF gradients has been swapped. What makes NYTUN different from the other WDSs in that it is essentially a pipe duplication problem, which means that almost every failed pipe is ensured a backup between the same set of nodes.

The positive correlation between RSMs and ADS-values is definitely apparent. The RSMs are compared directly in cost-ADSU-space for NYTUN in Figure 8.23. The RSMs are neatly divided into sub-fronts, with Resilience Index forming the vast majority of the Pareto-front. The second-most successful RSM is Network Resilience, forming a secondary front, followed by the Mixed Surrogate and finally Flow Entropy. It is interesting to note how most of the fronts converge in the low ADSU and high ADSU regions. The RSMs are compared in cost-ADSF-space for NYTUN in Figure 8.24. This situation is similar to the ADSU graph, except that Resilience Index improves its coverage of the Pareto-front, exclusively locating non-dominated solutions from ADSF 0.9968 onwards. Furthermore, the Mixed Surrogate squeezes past Network Resilience in the higher ADSF region. Flow Entropy fails to locate any non-dominated solutions.

RSM	**Avg ADSU**	**Mx ADSU**	**Mx ADSU/Cst**	**R^2 ADSU**	**Signif F**
Resilience Index	**0.9923**	1	$\mathbf{2.48\times10^{-8}}$	0.7021	9.46×10^{-55}
Network Resilience	0.9918	1	2.37×10^{-8}	0.9248	4.74×10^{-94}
Flow Entropy	0.9830	0.9956	1.23×10^{-8}	**0.9716**	4.78×10^{-100}
Mixed Surrogate	0.9916	1	2.26×10^{-8}	0.9446	$\mathbf{1.39\times10^{-127}}$
RSM	**Avg ADSF**	**Mx ADSF**	**Mx ADSF/Cst**	**R^2 ADSF**	**Signif F**
Resilience Index	0.9977	0.9985	$\mathbf{2.54\times10^{-8}}$	0.7461	3.10×10^{-63}
Network Resilience	0.9977	0.9985	2.42×10^{-8}	**0.9415**	$\mathbf{2.11\times10^{-110}}$
Flow Entropy	0.9969	0.9977	2.32×10^{-8}	0.9101	2.67×10^{-68}
Mixed Surrogate	0.9976	0.9985	2.31×10^{-8}	0.8527	4.97×10^{-93}

Table 8.7*: Reliability comparisons of RSMs using ADS measures for the NYTUN benchmark.*

RSM	**Count**	**Avg Cost**	**Avg Pipes**	**Avg SDD**	**Avg SQDD**	**Avg SSDM**
Resilience Index	208	8.86×10^{7}	8.95	3 207	404 987	0
Network Resilience	178	1.08×10^{8}	13.59	2 795	321 686	0
Flow Entropy	129	9.86×10^{7}	14.26	2 742	289 799	0
Mixed Surrogate	**221**	1.19×10^{8}	14.19	**2 702**	**285 516**	0

Table 8.8*: Result comparisons of RSMs using WDS features for the NYTUN benchmark.*

8.3.5 BLACK Reliability Analysis

The reliability analysis results of the four RSMs for the BLACK benchmark are shown in Table 8.9. The Mixed Surrogate obtained the highest average ADSU and ADSF-values of 0.9920 and 0.8384, respectively. All RSMs, except Flow Entropy, were able to locate solutions with the maximum ADSU-value of 1.0. The solution of highest cost-benefit in terms of ADSU (1.64×10^{-5}) was found by both Resilience Index and Flow Entropy. The solution of highest cost-benefit in terms of ADSF (1.26×10^{-5}) was located exclusively by Flow Entropy.

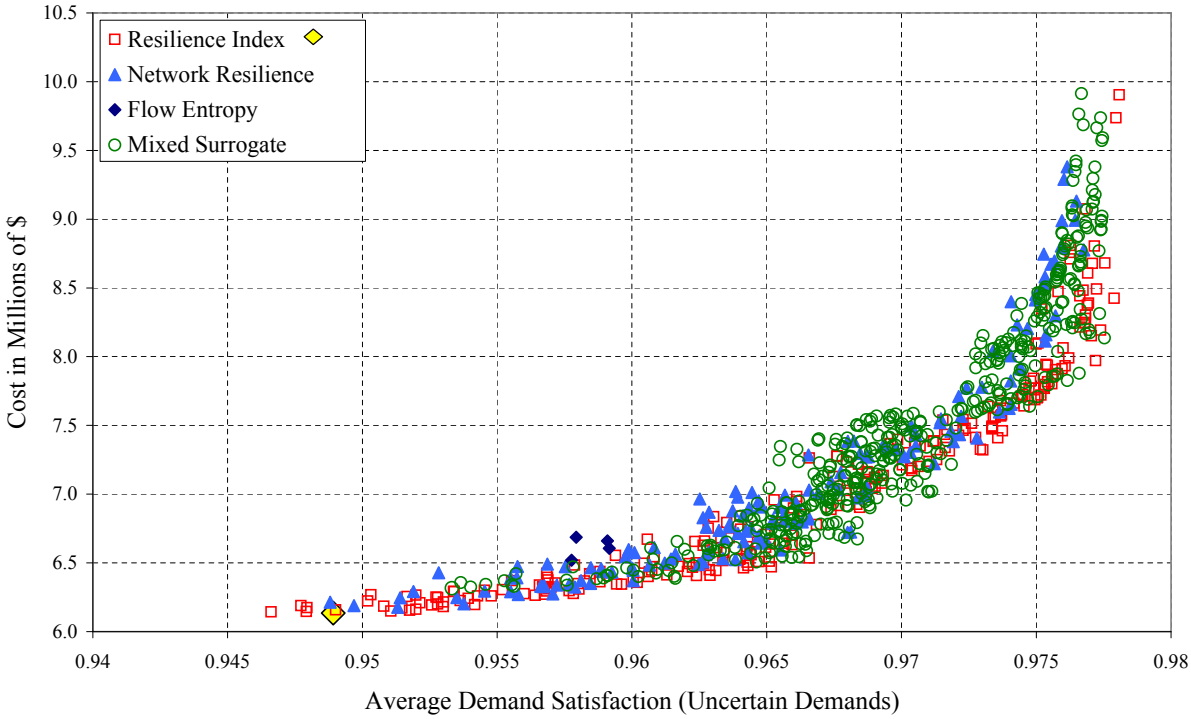


Figure 8.17: *Comparison of RSMs: ADSU vs Cost for the HANOI benchmark.*

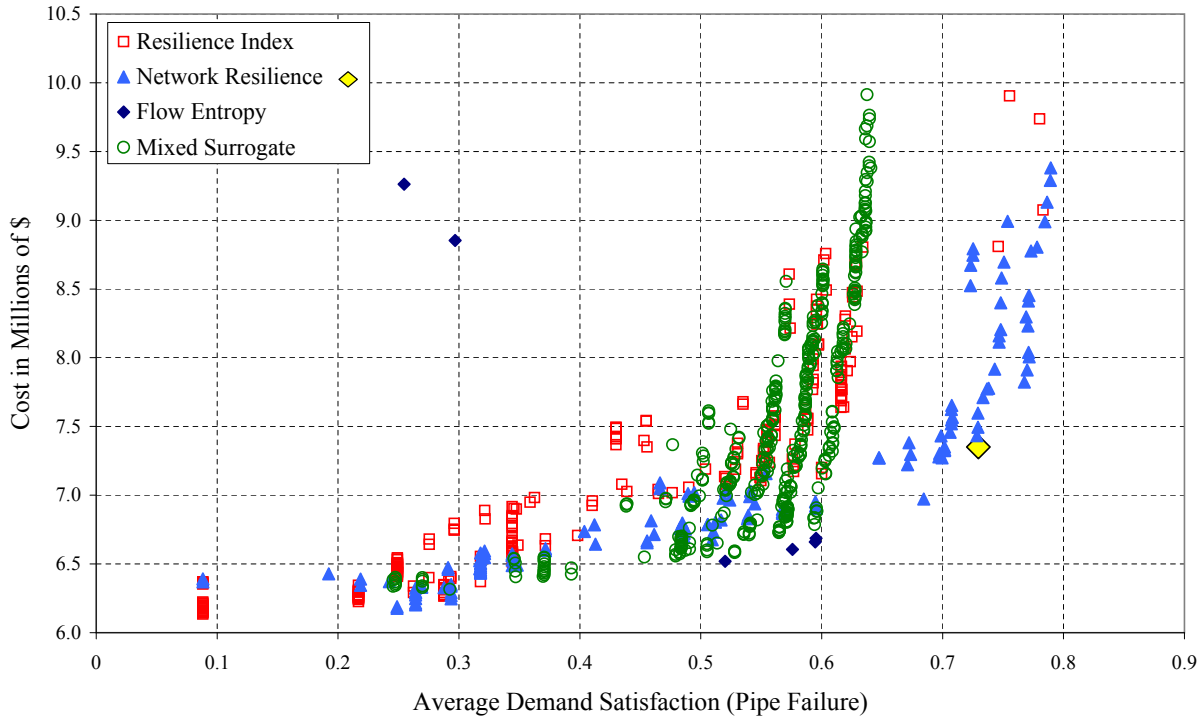


Figure 8.18: *Comparison of RSMs: ADSF vs Cost for the HANOI benchmark.*

RSM	Avg ADSU	Mx ADSU	Mx ADSU/Cst	R^2 ADSU	Signif F
Resilience Index	0.9862	1	1.64×10^{-5}	**0.8835**	3.22×10^{-21}
Network Resilience	0.9853	1	1.63×10^{-5}	0.8732	3.09×10^{-27}
Flow Entropy	0.9816	0.9967	1.64×10^{-5}	0.7595	8.89×10^{-32}
Mixed Surrogate	**0.9920**	1	1.46×10^{-5}	0.8711	$\mathbf{8.52\times10^{-84}}$
RSM	**Avg ADSF**	**Mx ADSF**	**Mx ADSF/Cst**	**R^2 ADSF**	**Signif F**
Resilience Index	0.6384	**0.8843**	1.12×10^{-5}	0.7575	4.61×10^{-16}
Network Resilience	0.7194	0.8840	1.25×10^{-5}	**0.8185**	$\mathbf{2.09\times10^{-25}}$
Flow Entropy	0.8044	0.8506	$\mathbf{1.26\times10^{-5}}$	0.2903	8.74×10^{-9}
Mixed Surrogate	**0.8384**	0.8840	1.20×10^{-5}	0.1897	5.18×10^{-10}

Table 8.9*: Reliability comparisons of RSMs using ADS measures for the BLACK benchmark.*

RSM	Count	Avg Cost	Avg Pipes	Avg SDD	Avg SQDD	Avg SSDM
Resilience Index	49	1.06×10^{5}	19.65	3 021	476 494	0
Network Resilience	66	8.46×10^{4}	21.76	**2 369**	**323 684**	0
Flow Entropy	99	8.69×10^{4}	22.37	2 658	509 024	0
Mixed Surrogate	186	1.27×10^{5}	22.97	3 286	760 716	0

Table 8.10*: Result comparisons of RSMs using WDS features for the BLACK benchmark.*

The regression analysis for BLACK indicated that there is a statistically significant relationship between all RSMs and their ADS counterparts, with Significance F-values less than 0.05. The Resilience Index measure demonstrated the highest R^2-value of 0.8835 with respect to ADSU, and Network Resilience achieved the highest R^2-value of 0.8185 with respect to ADSF. The lowest Significance F-values were generated by the Mixed Surrogate measure for ADSU and Network Resilience for ADSF.

The additional performance results are shown in Table 8.10. The Mixed Surrogate located 186 solutions, again finding the most. Resilience Index uncovered the fewest solutions, namely 49. The Mixed Surrogate produced the solutions with the highest average cost of 1.27×10^5, and Network Resilience yielded the lowest average cost solution of 8.46×10^4. The Mixed Surrogate used the highest average of 22.97 pipes, while Resilience Index used the fewest pipes at an average of 19.65. Network Resilience demonstrated the lowest average SDD and SQDD-values of 2 369 and 323 684, respectively. SSDM is not applicable to BLACK since it has only a single water source.

Graphs of the BLACK attainment fronts for the various RSMs, along with their corresponding ADSU and ADSF-values indicated on the secondary vertical axis, may be found in Figures 8.25–8.28. While the RSM-ADS curves are less well-defined than for the previous benchmarks, there is a certain positive correlation, even in the case of Flow Entropy versus ADSF, which has a mild positive gradient.

The RSMs are compared directly in cost-ADSU-space for BLACK in Figure 8.29. The situation is quite similar to that for TLN, with Resilience Index and Network Resilience forming the majority of the Pareto-front, the Mixed Surrogate forming a secondary front, followed by a third front attributable to Flow Entropy. The latter two RSMs do not contribute to the Pareto-front. The RSMs are compared in cost-ADSF space for BLACK in Figure 8.30. The tables have turned for Resilience Index which no longer participates in the Pareto-front. The other three RSMs each contribute non-dominated solutions, but Network Resilience is clearly the most widely distributed along the Pareto-front and locates the best solutions in the high ADSF region.

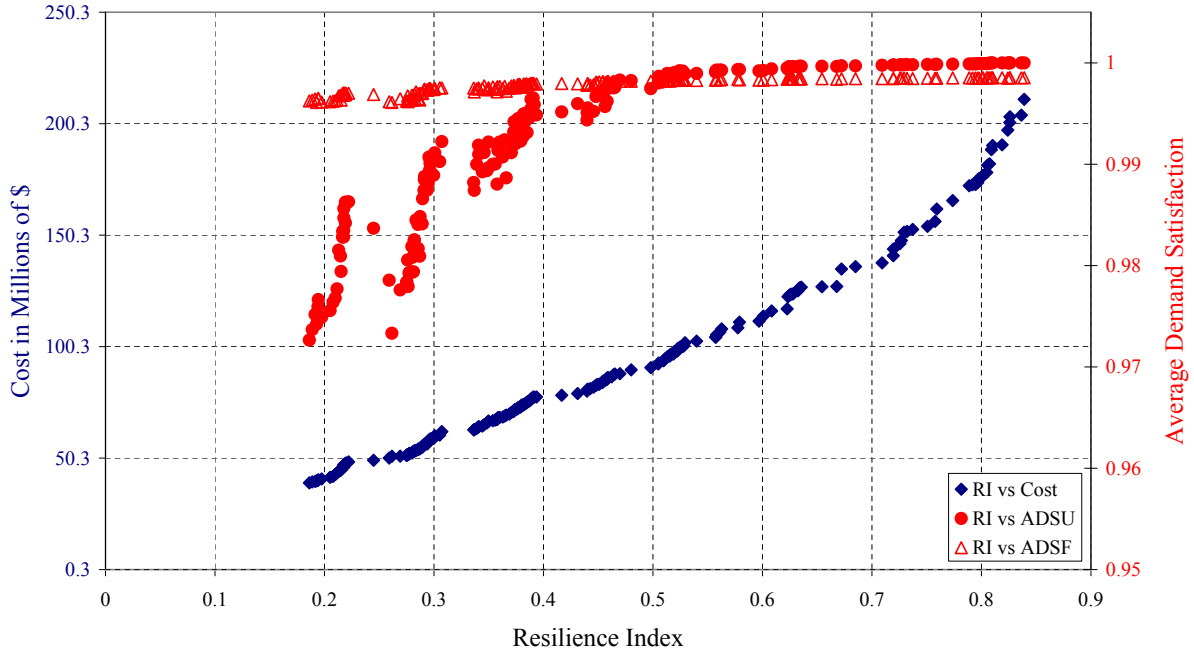


Figure 8.19: *Resilience Index versus ADSU (Uncertainty) and ADSF (Pipe Failure) for the NYTUN benchmark, with average demand satisfaction ratio on secondary axis.*

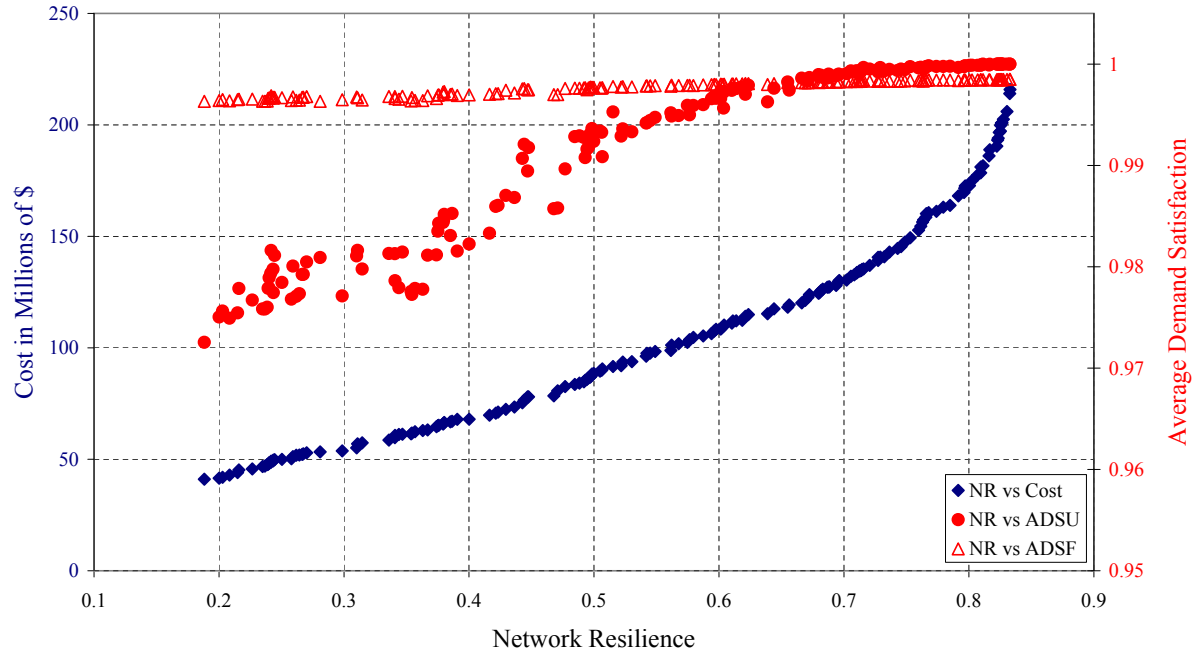


Figure 8.20: *Network Resilience versus ADSU (Uncertainty) and ADSF (Pipe Failure) for the NYTUN benchmark, with average demand satisfaction ratio on secondary axis.*

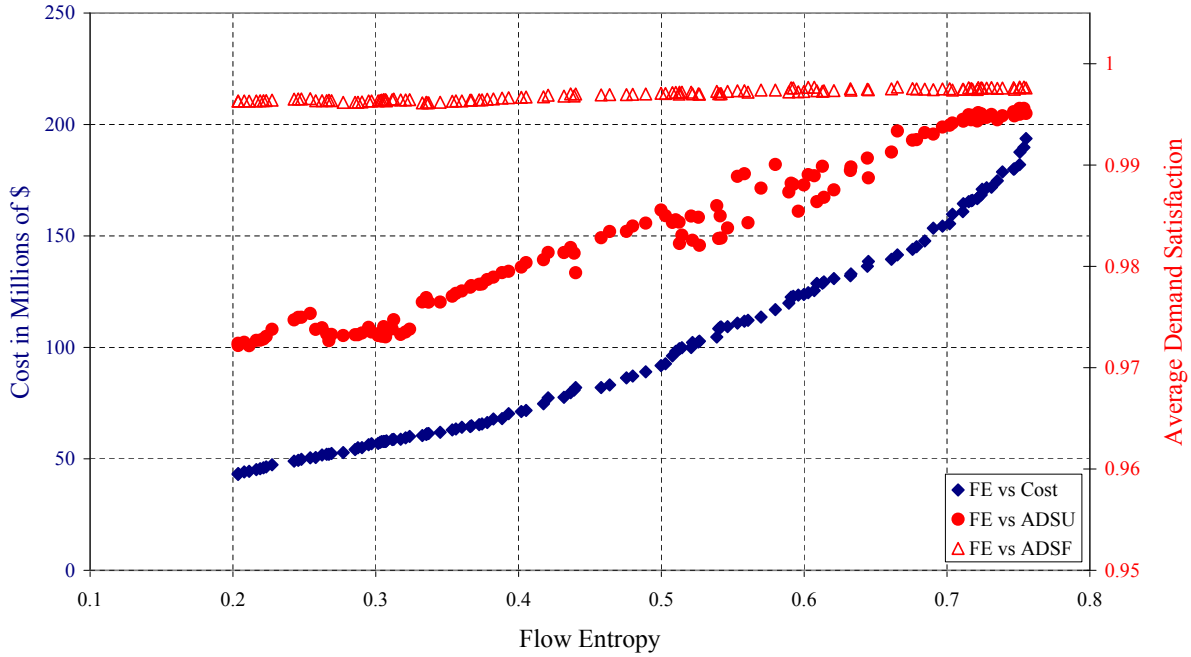


Figure 8.21: *Flow Entropy versus ADSU (Uncertainty) and ADSF (Pipe Failure) for the NYTUN benchmark, with average demand satisfaction ratio on secondary axis.*

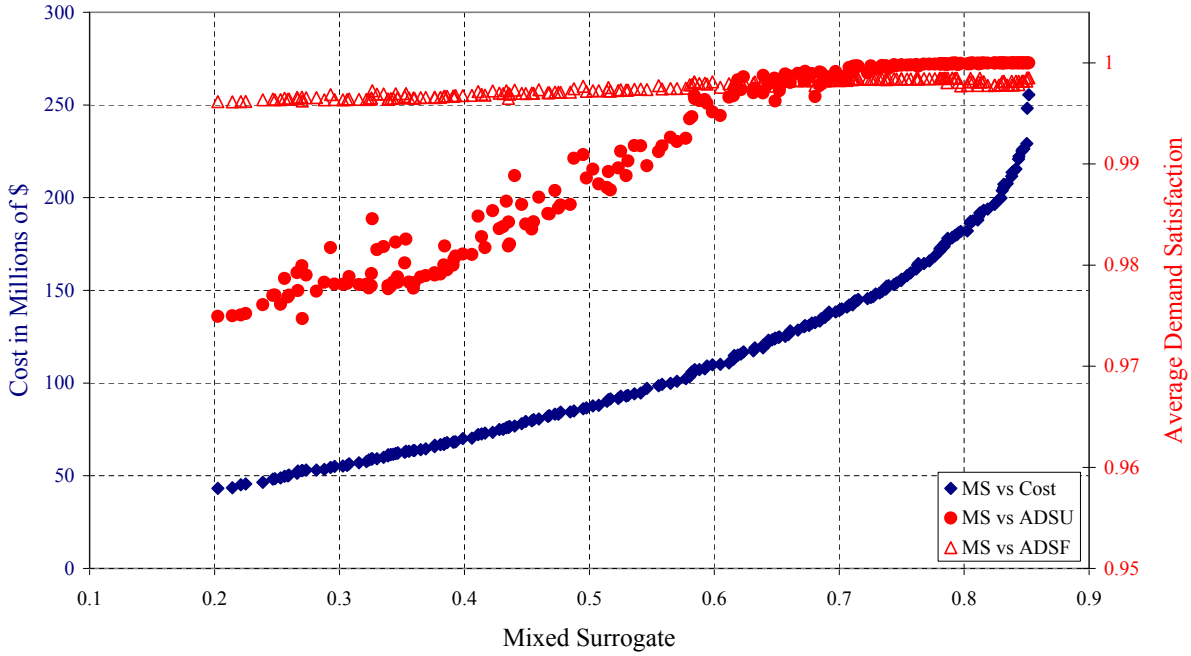


Figure 8.22: *Mixed surrogate versus ADSU (Uncertainty) and ADSF (Pipe Failure) for the NYTUN benchmark, with average demand satisfaction ratio on secondary axis.*

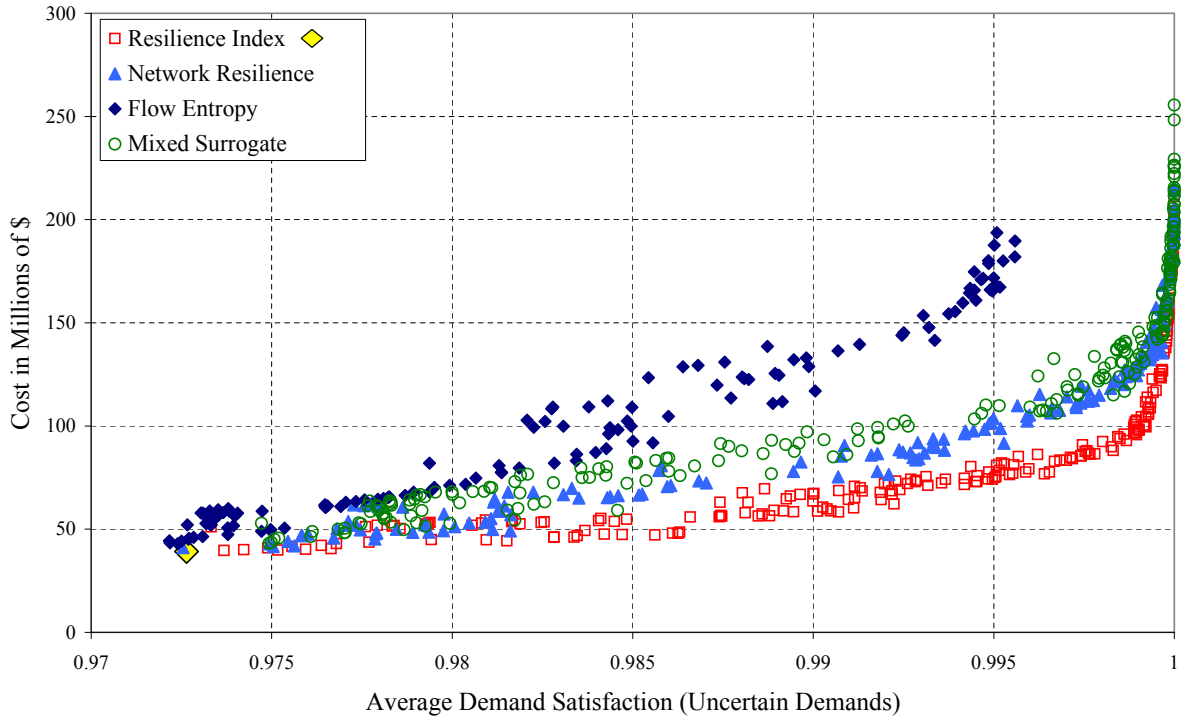


Figure 8.23: *Comparison of RSMs: ADSU vs Cost for the NYTUN benchmark.*

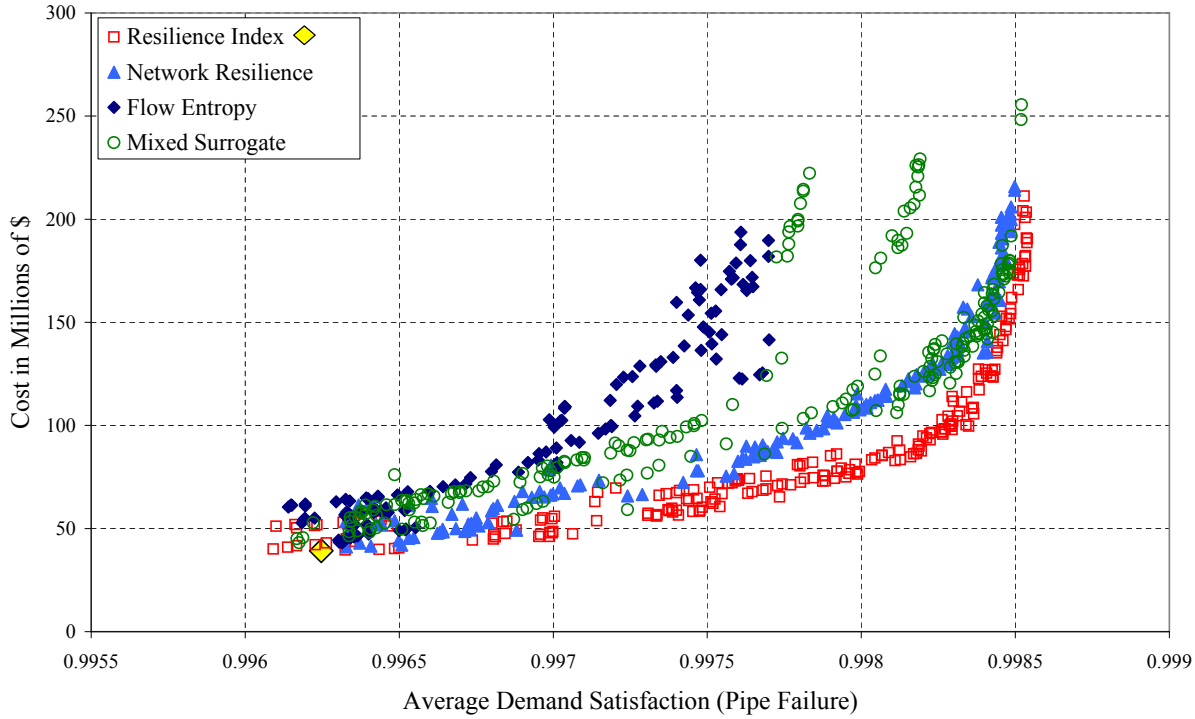


Figure 8.24: *Comparison of RSMs: ADSF vs Cost for the NYTUN benchmark.*

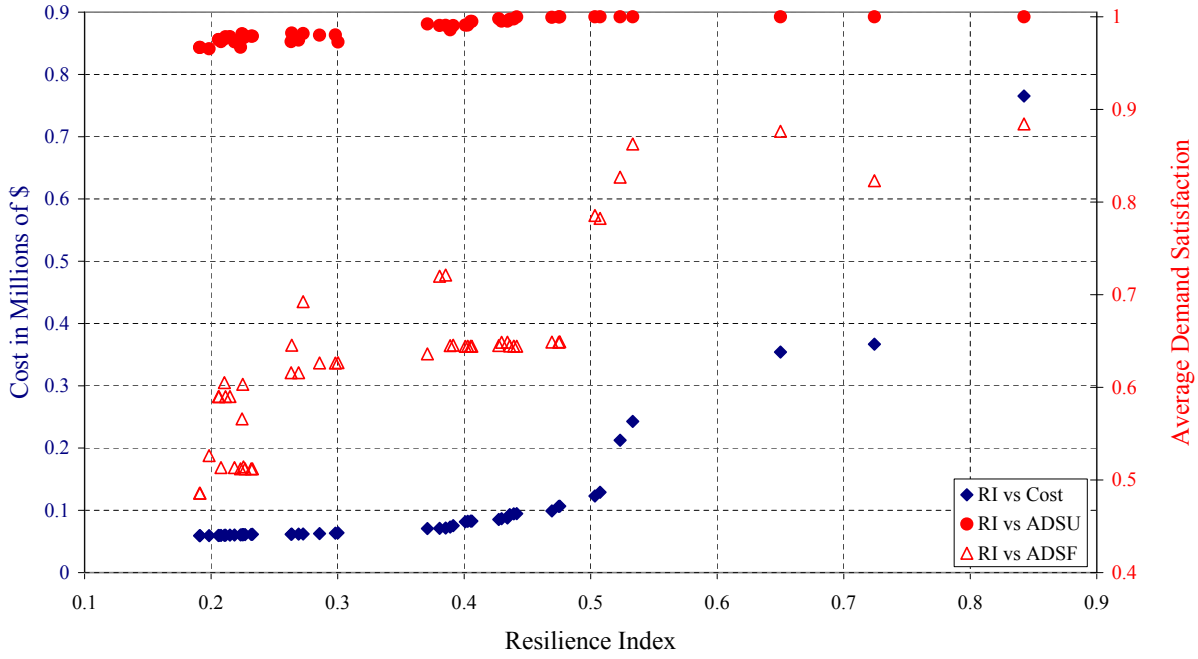


Figure 8.25: *Resilience Index versus ADSU (Uncertainty) and ADSF (Pipe Failure) for the BLACK benchmark, with average demand satisfaction ratio on secondary axis.*

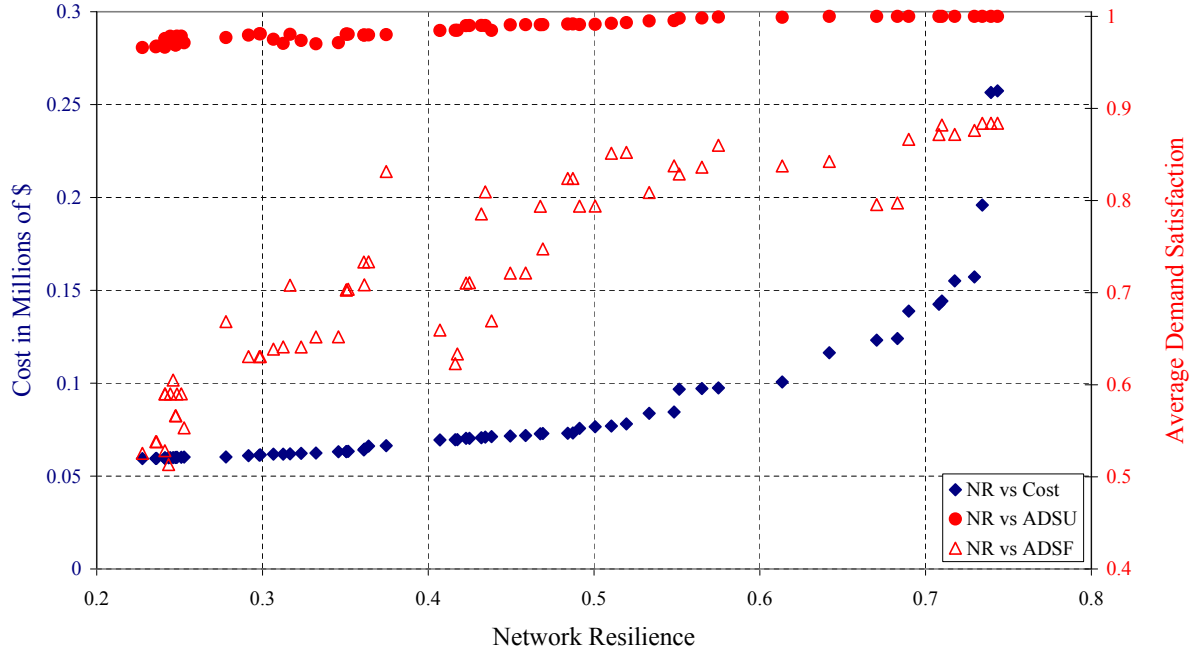


Figure 8.26: *Network Resilience versus ADSU (Uncertainty) and ADSF (Pipe Failure) for the BLACK benchmark, with average demand satisfaction ratio on secondary axis.*

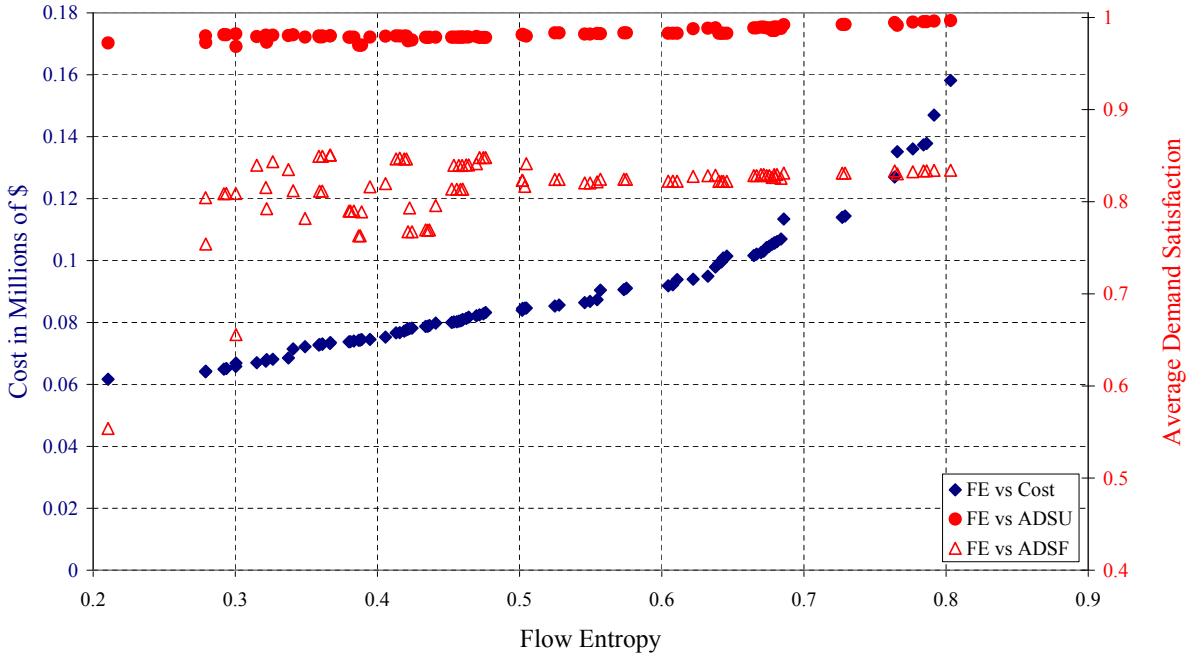


Figure 8.27: *Flow Entropy versus ADSU (Uncertainty) and ADSF (Pipe Failure) for the BLACK benchmark, with average demand satisfaction ratio on secondary axis.*

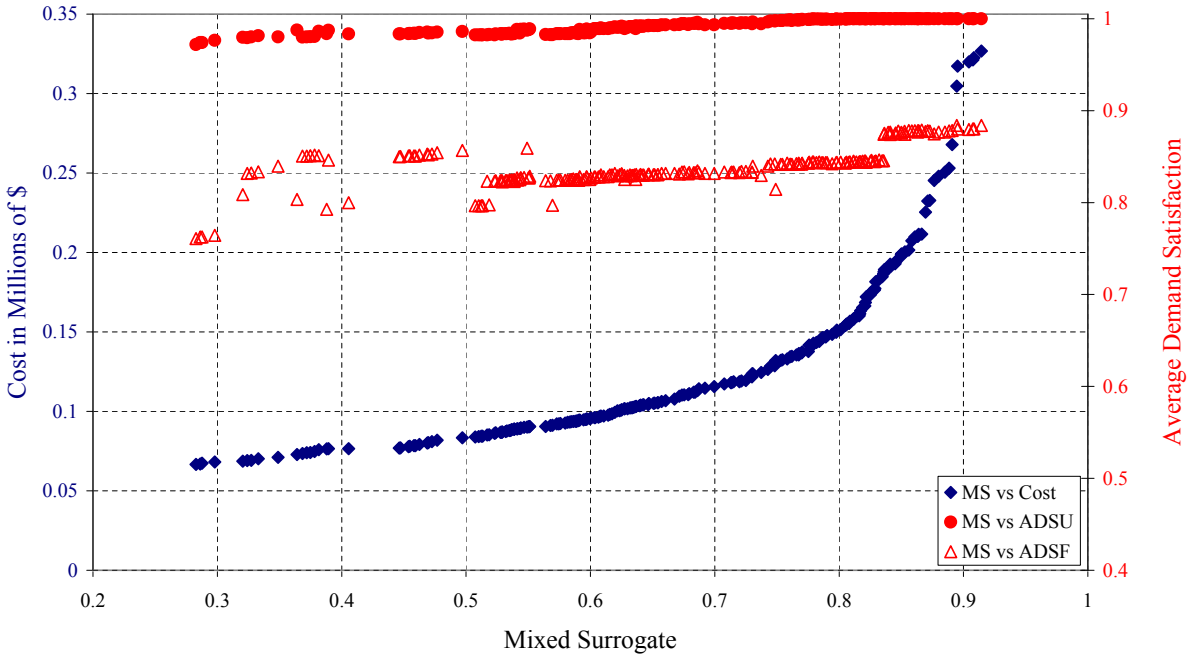


Figure 8.28: *Mixed surrogate versus ADSU (Uncertainty) and ADSF (Pipe Failure) for the BLACK benchmark, with average demand satisfaction ratio on secondary axis.*

RSM	Avg ADSU	Mx ADSU	Mx ADSU/Cst	R^2 ADSU	Signif F
Resilience Index	0.9981	1	$\mathbf{4.01\times10^{-5}}$	0.7596	2.53×10^{-13}
Network Resilience	0.9982	1	3.66×10^{-5}	**0.8075**	$\mathbf{4.51\times10^{-18}}$)
Flow Entropy	0.9820	0.9963	2.48×10^{-5}	0.2169	5.73×10^{-4}
Mixed Surrogate	**0.9989**	1	2.71×10^{-5}	0.5714	5.25×10^{-16}
RSM	**Avg ADSF**	**Mx ADSF**	**Mx ADSF/Cst**	**R^2 ADSF**	**Signif F**
Resilience Index	0.9275	0.9828	$\mathbf{3.51\times10^{-5}}$	0.6673	5.01×10^{-14}
Network Resilience	0.9679	0.9828	3.42×10^{-5}	**0.8155**	$\mathbf{5.07\times10^{-27}}$
Flow Entropy	0.9549	0.9638	2.42×10^{-5}	0.0641	7.31×10^{-2}
Mixed Surrogate	**0.9786**	0.9828	2.65×10^{-5}	0.1456	2.18×10^{-6}

Table 8.11*: Reliability comparisons of RSMs using ADS measures for the FOSS benchmark.*

RSM	Count	Avg Cost	Avg Pipes	Avg SDD	Avg SQDD	Avg SSDM
Resilience Index	54	5.11×10^4	53.64	2 666	192 345	0
Network Resilience	71	4.35×10^4	57.61	**1 570**	**64 346**	0
Flow Entropy	51	5.66×10^4	57.45	2 705	168 749	0
Mixed Surrogate	**145**	8.92×10^4	57.39	3 184	218 337	0

Table 8.12*: Result comparisons of RSMs using WDS features for the FOSS benchmark.*

8.3.6 FOSS Reliability Analysis

The reliability analysis results of the four RSMs for the FOSS benchmark are shown in Table 8.11. The Mixed Surrogate achieved the highest average ADSU and ADSF-values of 0.9989 and 0.9786, respectively. Only Flow Entropy failed to locate solutions with the maximum ADSU-value of 1.0. Resilience Index located the solution of highest cost-benefit in terms of ADSU (4.01×10^{-5}), as well as the solution of highest cost-benefit in terms of ADSF (3.51×10^{-5}).

The regression analysis for FOSS indicated that there is a statistically significant relationship between all RSMs and their ADS counterparts, except between Flow Entropy and ADSF, which had a Significance F-value greater than than 0.05. Network Resilience demonstrated the highest R^2-values of 0.8075 with respect to ADSU, and 0.8155 with respect to ADSF. The lowest Significance F-values were also generated by the Network Resilience.

The additional performance results appear in Table 8.12. The Mixed Surrogate located the most solutions by far, at 145, while the Flow Entropy measure found the fewest, at 51. The Mixed Surrogate produced the solutions with the highest average cost of 8.92×10^4, while Network Resilience produced solutions with the lowest average cost of 4.35×10^4. Network Resilience used the highest average of 57.61 pipes, and Resilience Index continued the trend of using the fewest pipes, at an average of 53.64. Network Resilience also demonstrated the lowest average SDD and SQDD-values of 1 570 and 64 346, respectively. SSDM is not applicable to FOSS since it has only a single water source.

Graphs of the FOSS attainment fronts for the various RSMs, along with their corresponding ADSU and ADSF-values indicated on the secondary vertical axis, are shown in Figures 8.31–8.34. Positive correlations are apparent between the RSM and ADS-values for Network Resilience, Resilience Index and Mixed Surrogate. Flow Entropy produces a strange effect where ADSU and ADSF increase initially with increasing Flow Entropy, then drop again, and then finally rise. The Mixed Surrogate also demonstrates an initial period of rapid ADS increase with increasing Mixed Surrogate, which is followed by long flat plateaus of ADS values.

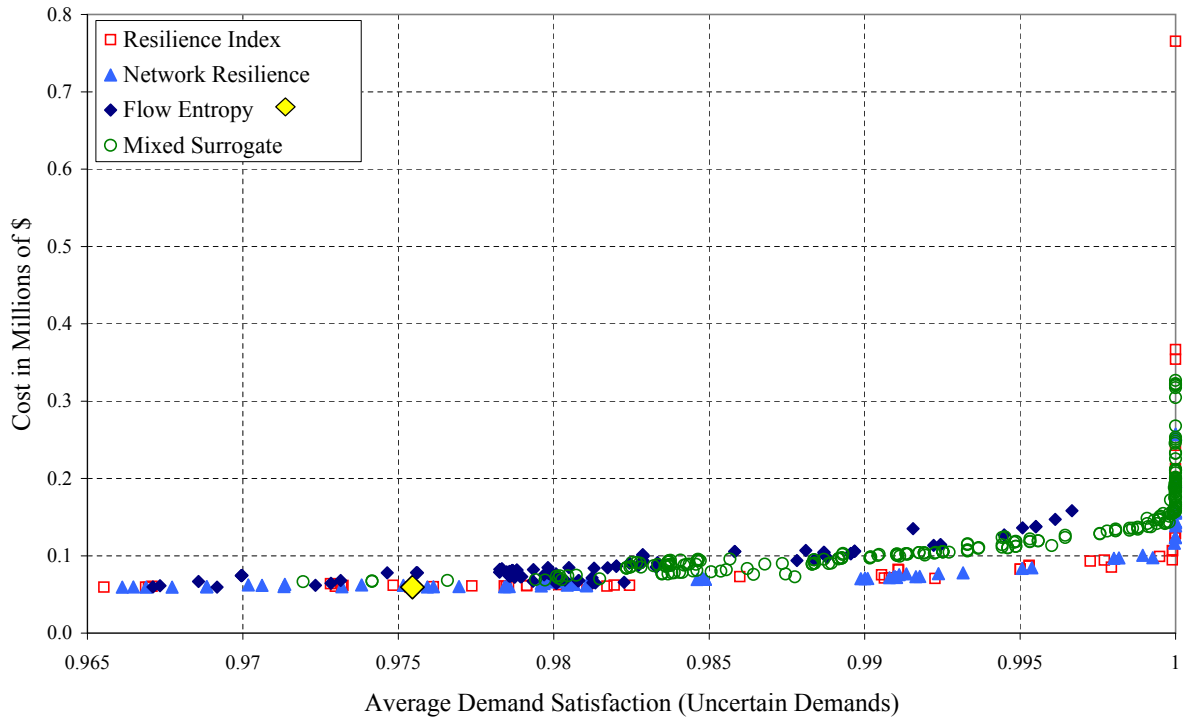


Figure 8.29: *Comparison of RSMs: ADSU vs Cost for the BLACK benchmark.*

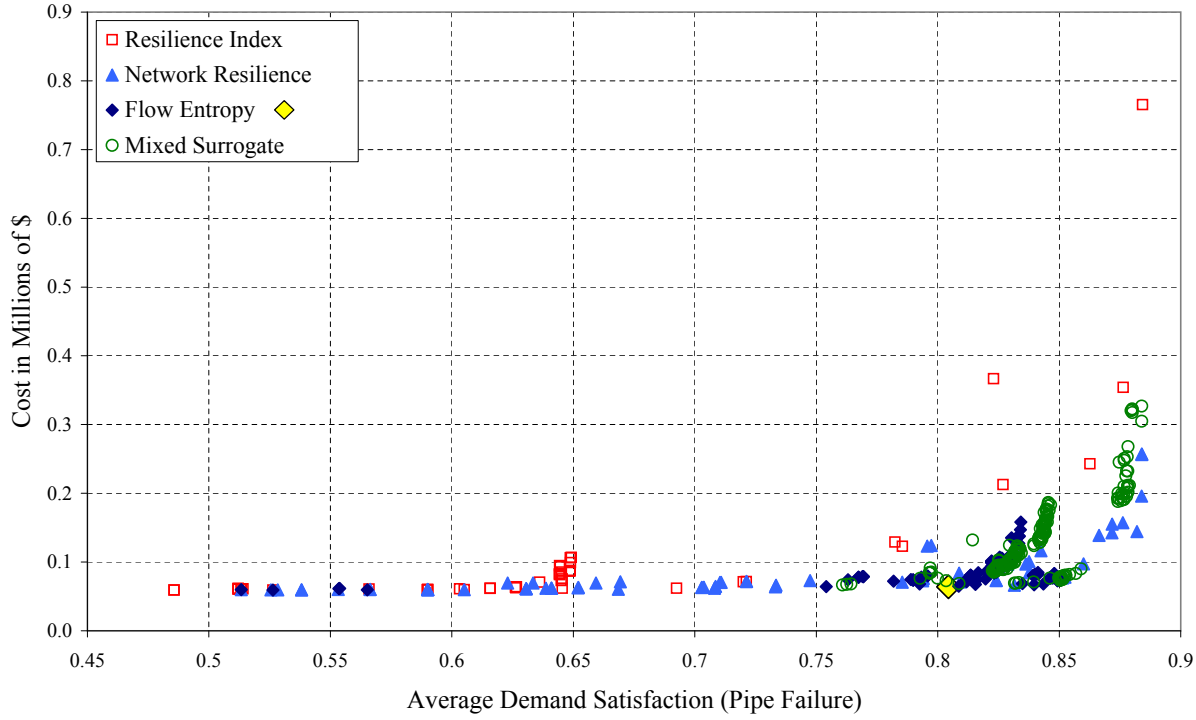


Figure 8.30: *Comparison of RSMs: ADSF vs Cost for the BLACK benchmark.*

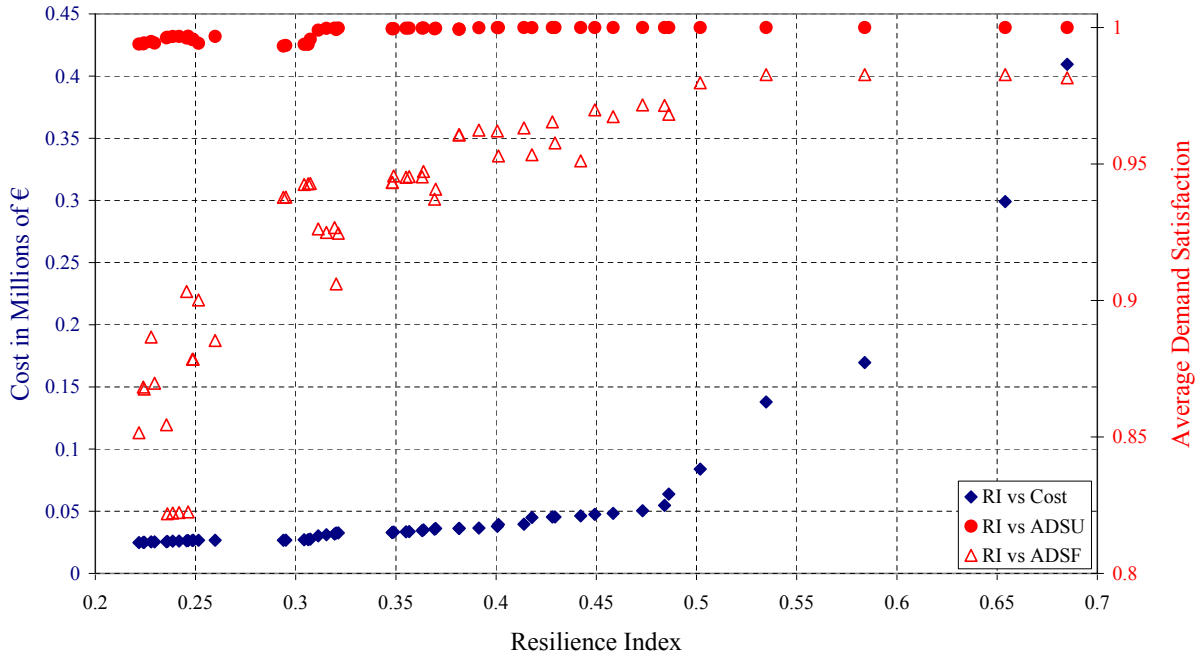


Figure 8.31: *Resilience Index versus ADSU (Uncertainty) and ADSF (Pipe Failure) for the FOSS benchmark, with average demand satisfaction ratio on secondary axis.*

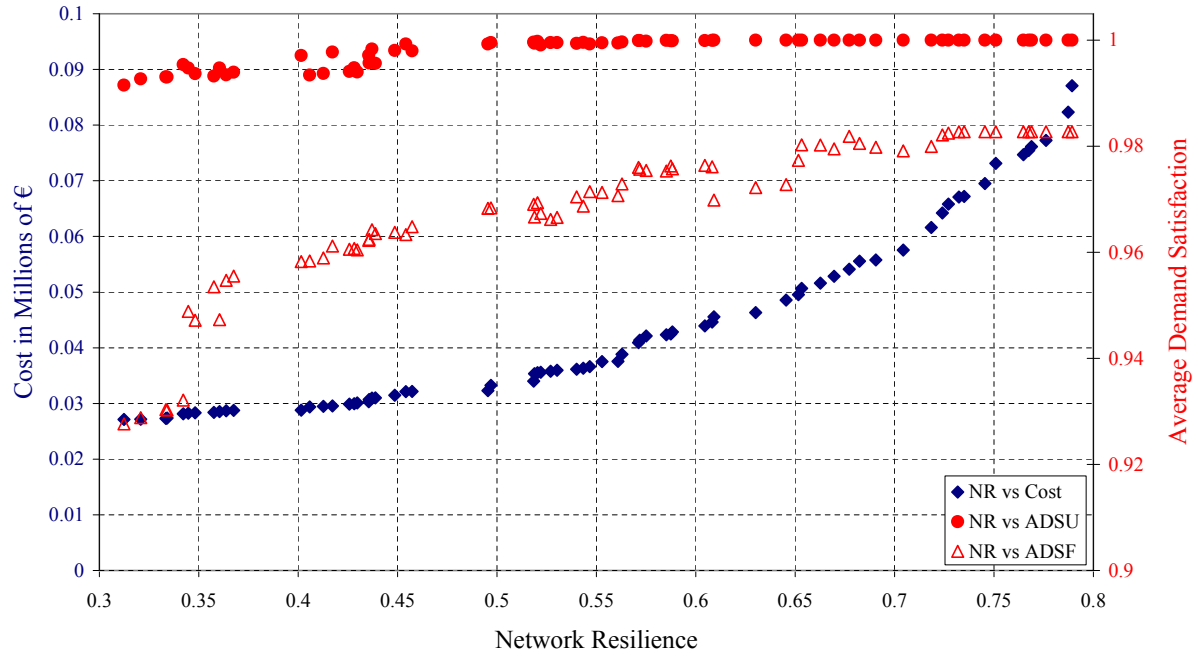


Figure 8.32: *Network Resilience versus ADSU (Uncertainty) and ADSF (Pipe Failure) for the FOSS benchmark, with average demand satisfaction ratio on secondary axis.*

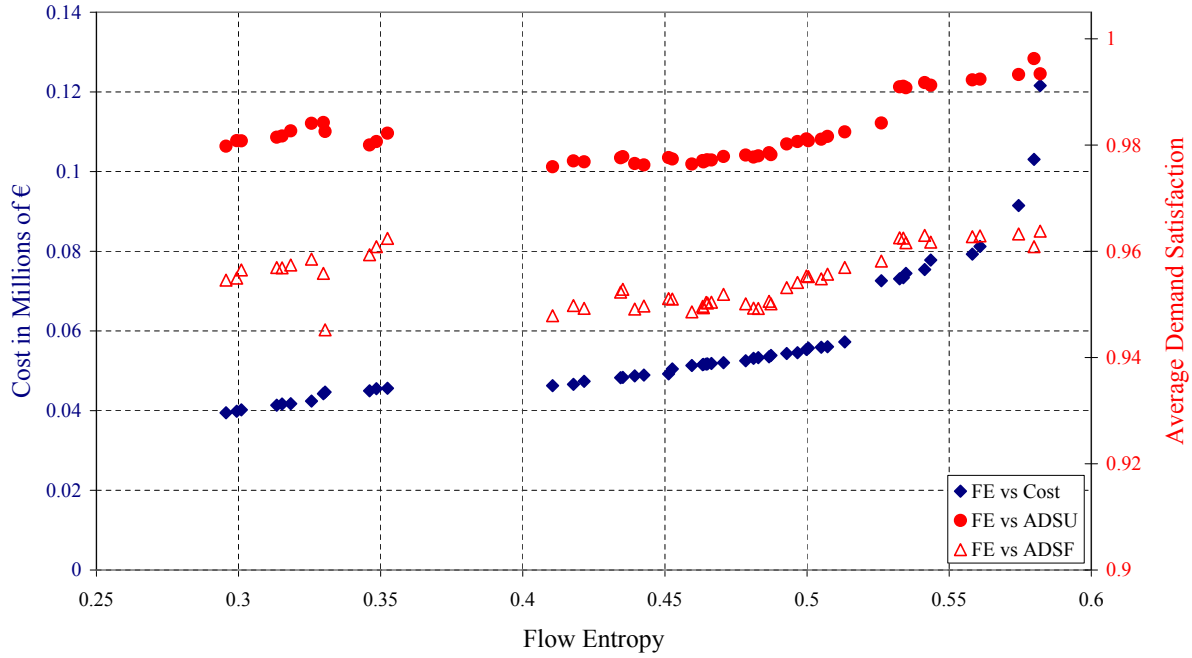


Figure 8.33: *Flow Entropy versus ADSU (Uncertainty) and ADSF (Pipe Failure) for the FOSS benchmark, with average demand satisfaction ratio on secondary axis.*

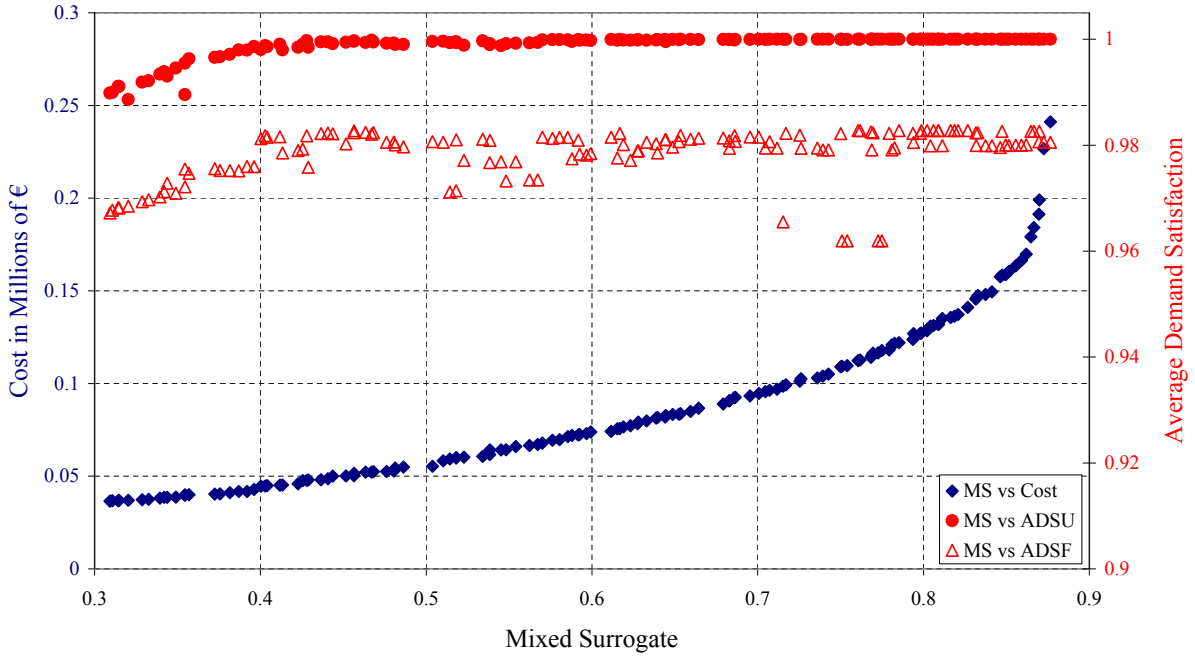


Figure 8.34: *Mixed surrogate versus ADSU (Uncertainty) and ADSF (Pipe Failure) for the FOSS benchmark, with average demand satisfaction ratio on secondary axis.*

The RSMs are compared directly in cost-ADSU space for FOSS in Figure 8.35. Only Resilience Index and Network Resilience were able to produce solutions along the Pareto-front. The Mixed Surrogate forms a disjoint secondary front, and Flow Entropy provides a badly dominated tertiary front. The RSMs are compared in cost-ADSF-space for FOSS in Figure 8.36. Resilience Index is relegated to the lower regions of ADSF-space, being outperformed by Network Resilience and Mixed Surrogate, which together make up most of the Pareto-front, particularly in the high ADSF regions. Flow Entropy is again entirely dominated in this objective space.

8.3.7 PESCARA Reliability Analysis

The reliability analysis results of the four RSMs for the PESC benchmark are shown in Table 8.13. The Mixed Surrogate method obtained the highest average ADSU and ADSF-values of 0.9980 and 0.7946, respectively. Only Flow Entropy was unable to locate a solution with the maximum ADSU of 1.0. Network Resilience found the solution with the maximum ADSF-value of 0.9484. The solution of highest cost-benefit in terms of ADSU (7.84×10^{-7}) was located by Resilience Index, while the solution of highest cost-benefit in terms of ADSF (4.72×10^{-7}) was located by the Network Resilience.

The regression analysis for PESC showed a statistically significant relationship between all RSMs and their ADS counterparts, with Significance F-values less than 0.05. The Mixed Surrogate measure demonstrated the highest R^2-values of 0.7244 with respect to ADSU, and 0.9283 with respect to ADSF. The lowest Significance F-value for ADSU was generated by Resilience Index, and the lowest value for ADSF by the Mixed Surrogate.

The additional performance results are shown in Table 8.14. As usual, the Mixed Surrogate located the most solutions, at 132, and Network Resilience located the fewest solutions, at 63. The Mixed Surrogate produced the solutions with the highest average cost of 3.44×10^6, while Resilience Index produced the solutions with the lowest average cost of 1.64×10^6. The Mixed Surrogate used the largest average number of pipes of 87.48, while Resilience Index once again used the fewest, with an average of 69.73 pipes. Flow entropy demonstrated the lowest average

RSM	**Avg ADSU**	**Mx ADSU**	**Mx ADSU/Cst**	**R^2 ADSU**	**Signif F**
Resilience Index	0.9915	1	$\mathbf{7.84\times10^{-7}}$	0.7226	$\mathbf{6.88\times10^{-21}}$
Network Resilience	0.9968	1	6.83×10^{-7}	0.2811	1.27×10^{-4}
Flow Entropy	0.9598	0.9869	5.31×10^{-7}	0.4883	8.04×10^{-13}
Mixed Surrogate	**0.9980**	1	5.97×10^{-7}	**0.7244**	1.05×10^{-20}
RSM	**Avg ADSF**	**Mx ADSF**	**Mx ADSF/Cst**	**R^2 ADSF**	**Signif F**
Resilience Index	0.3940	0.7877	2.86×10^{-7}	0.7263	2.37×10^{-25}
Network Resilience	0.7048	**0.9484**	$\mathbf{4.72\times10^{-7}}$	0.6974	1.78×10^{-17}
Flow Entropy	0.7592	0.8604	3.94×10^{-7}	0.7400	3.14×10^{-24}
Mixed Surrogate	**0.7946**	0.9250	3.92×10^{-7}	**0.9283**	$\mathbf{3.03\times10^{-76}}$

Table 8.13*: Reliability comparisons of RSMs using ADS measures for the PESC benchmark.*

RSM	**Count**	**Avg Cost**	**Avg Pipes**	**Avg SDD**	**Avg SQDD**	**Avg SRCDM**
Resilience Index	86	1.64×10^6	69.73	13 881	2 598 462	0.3870
Network Resilience	63	2.02×10^6	81.46	12 246	2 354 400	**0.3032**
Flow Entropy	79	2.33×10^6	86.70	**10 617**	**1 680 560**	0.3508
Mixed Surrogate	**132**	3.44×10^6	87.48	15 235	3 312 947	0.3079

Table 8.14*: Result comparisons of RSMs using WDS features for the PESC benchmark.*

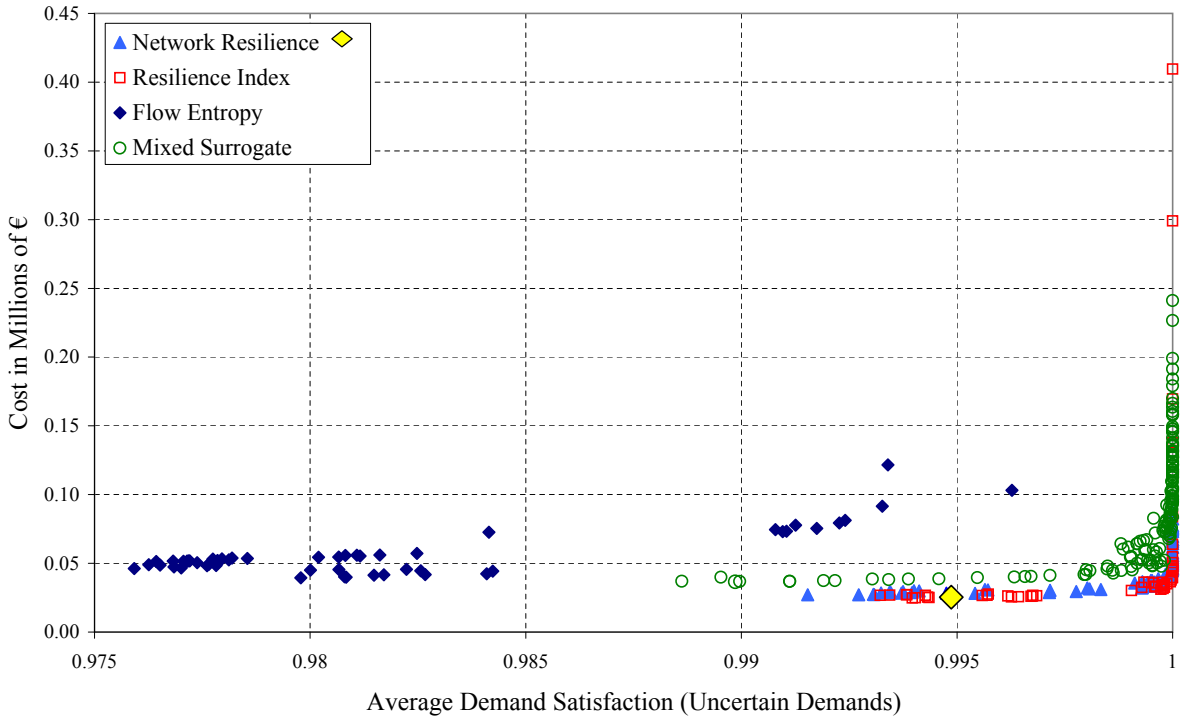


Figure 8.35: *Comparison of RSMs: ADSU vs Cost for the FOSS benchmark.*

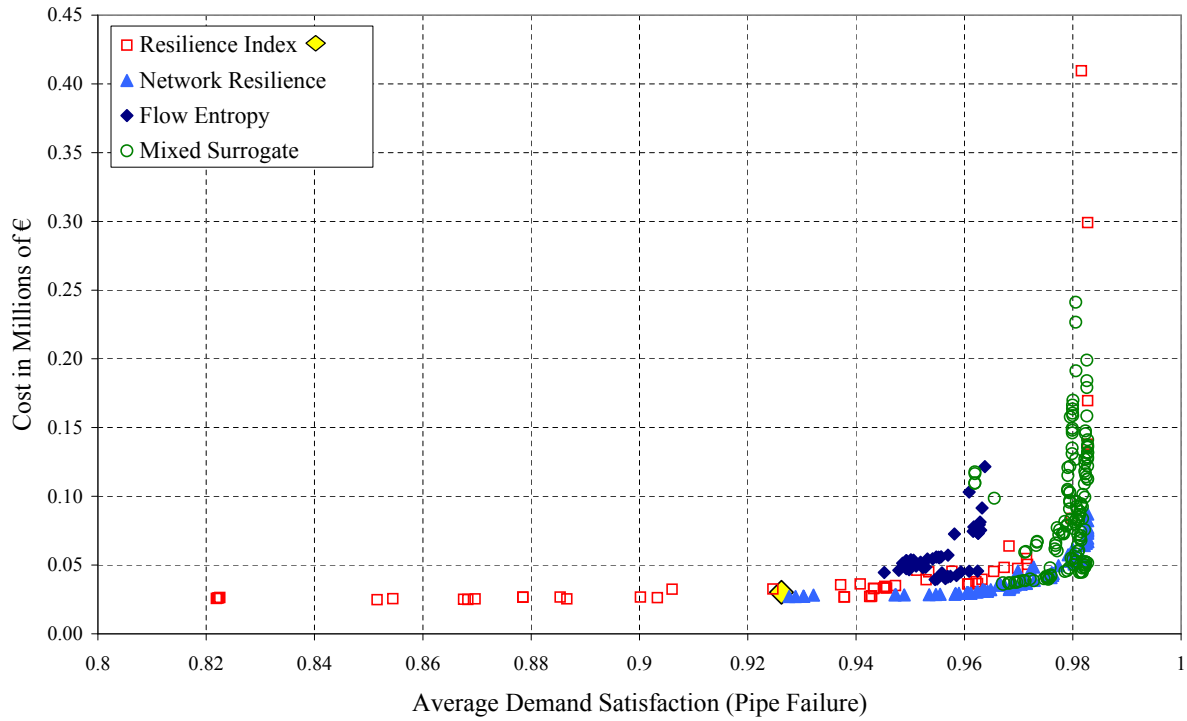


Figure 8.36: *Comparison of RSMs: ADSF vs Cost for the FOSS benchmark.*

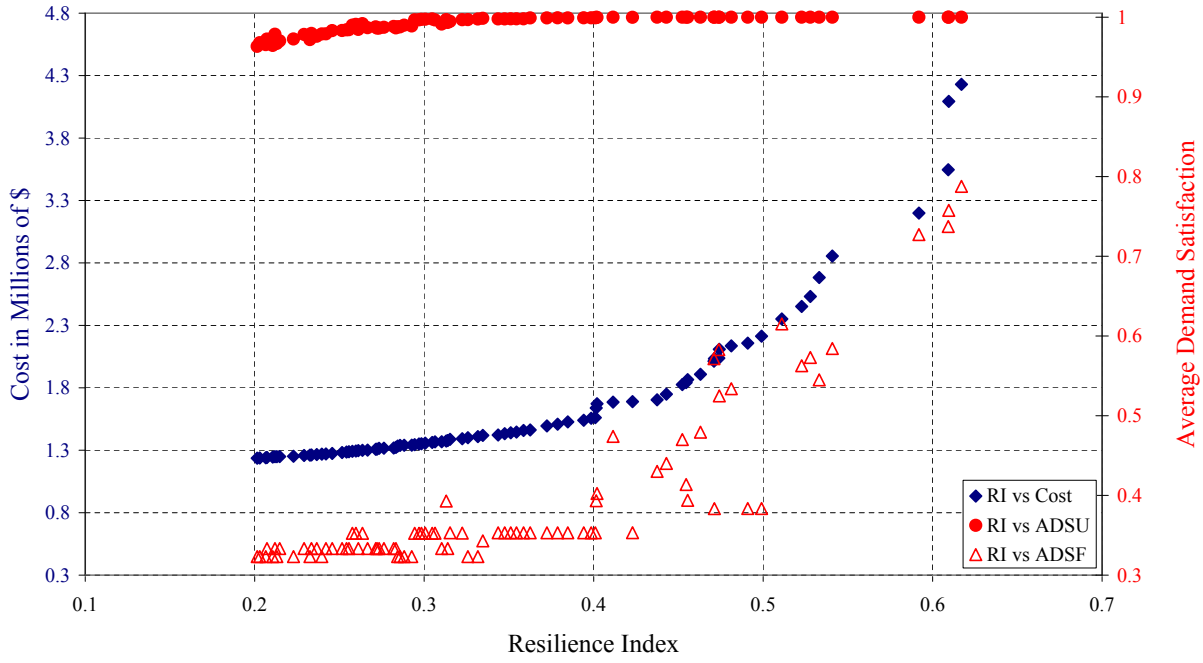


Figure 8.37: *Resilience Index versus ADSU (Uncertainty) and ADSF (Pipe Failure) for the PESC benchmark, with average demand satisfaction ratio on secondary axis.*

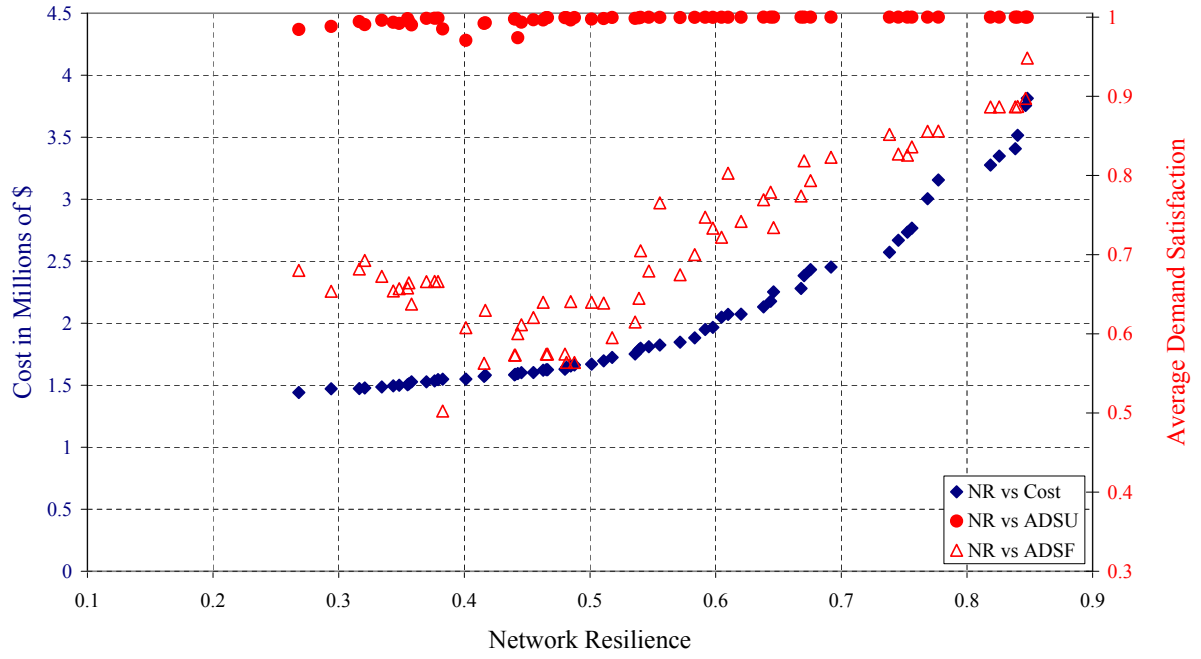


Figure 8.38: *Network Resilience versus ADSU (Uncertainty) and ADSF (Pipe Failure) for the PESC benchmark, with average demand satisfaction ratio on secondary axis.*

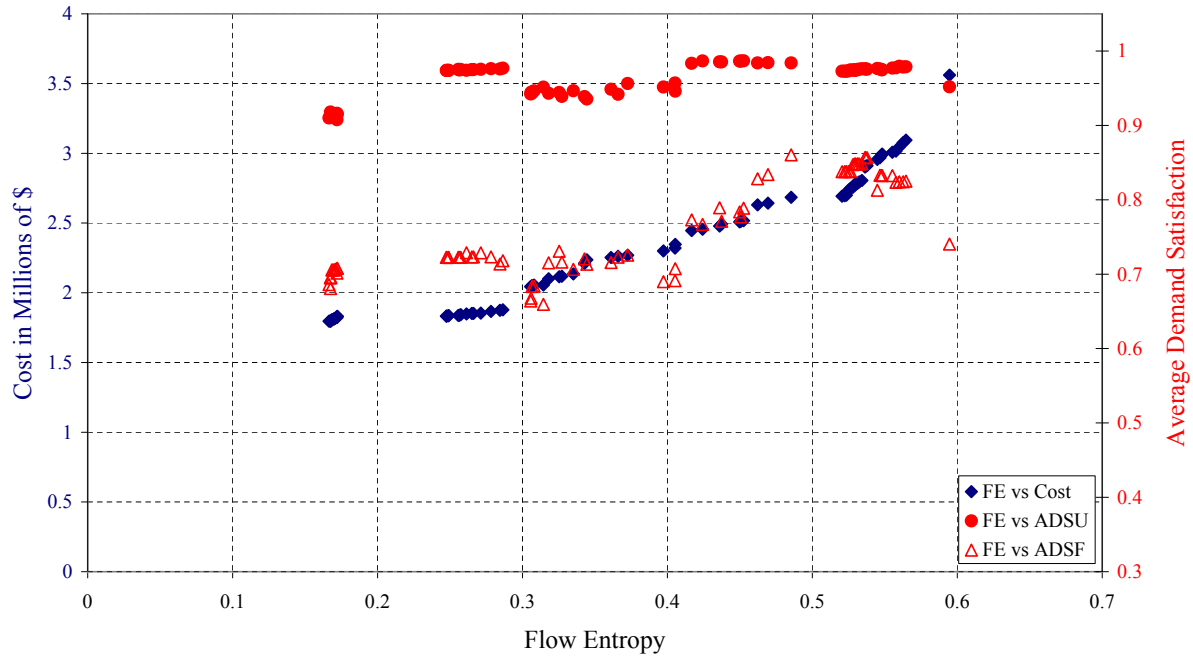


Figure 8.39: *Flow Entropy versus ADSU (Uncertainty) and ADSF (Pipe Failure) for the PESC benchmark, with average demand satisfaction ratio on secondary axis.*

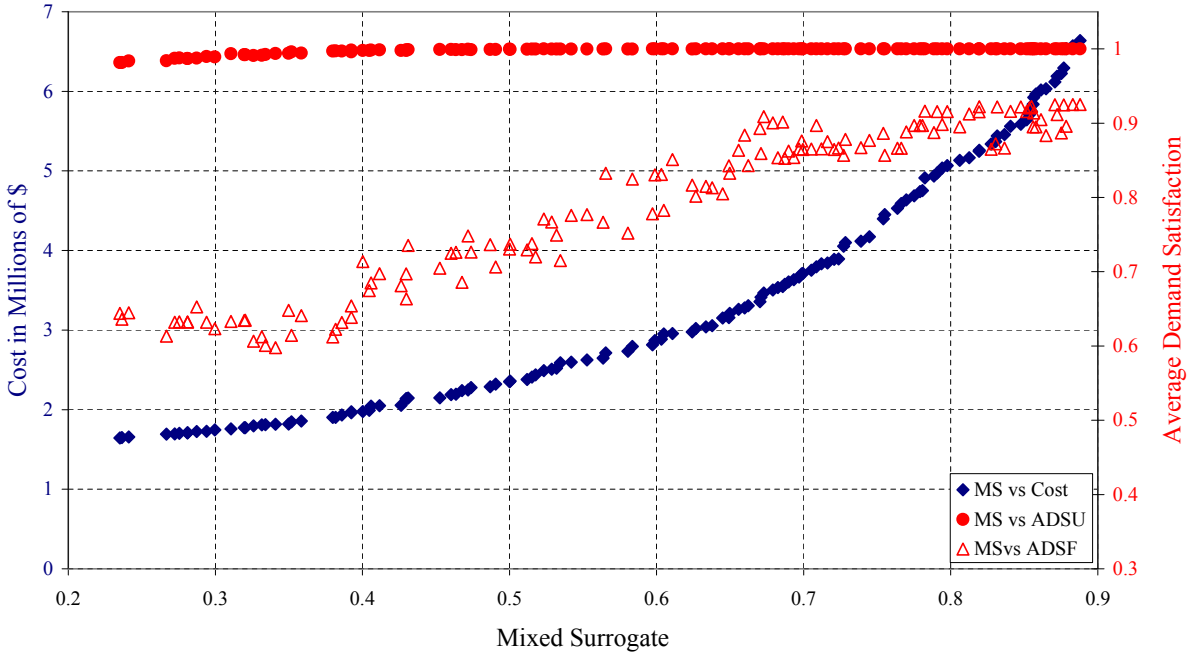


Figure 8.40: *Mixed surrogate versus ADSU (Uncertainty) and ADSF (Pipe Failure) for the PESC benchmark, with average demand satisfaction ratio on secondary axis.*

SDD and SQDD-values of 10 617 and 1 680 560, respectively. Network Resilience obtained the best SSDM value of 0.3032, compared to the worst value of 0.3870 obtained by Resilience Index.

Graphs of the PESC attainment fronts for the various RSMs, along with their corresponding ADSU and ADSF values indicated on the secondary vertical axis, are shown in Figures 8.37–8.40. A definite positive correlation between the RSM and ADS-values is visible for each RSM. Once of the most notable oddities is that Resilience Index produces many solutions of low ADSF (less than 0.4) while the other RSMs rarely produce solutions with ADSF values below 0.6.

The RSMs are compared directly in cost-ADSU-space for PESC in Figure 8.41. For the first time the RSMs are split into four almost disjoint fronts, with Reliability Index forming the full Pareto-front, Network Resilience forming the secondary Front, Mixed Surrogate the tertiary front, and finally Flow Entropy forming a badly dominated front. The RSMs are compared in cost-ADSF-space for PESC in Figure 8.42. The situation is very different, with Reliability Index showing the worst ADSF attainment. Network Resilience located most of the Pareto-front. The Mixed Surrogate formed a secondary front, although finding some non-dominated solutions in the region of ADSF around 0.9.

8.3.8 MODENA Reliability Analysis

The reliability analysis results of the four RSMs for the MOD benchmark are shown in Table 8.15. The Mixed Surrogate obtained the highest average ADSU and ADSF-values of 0.9992 and 0.9109, respectively. Only Flow Entropy was unable to locate solutions having the maximum ADSU of 1.0. Network Resilience uniquely located the solution of maximum ADSF-value, namely 0.9900. The solution of highest cost-benefit in terms of ADSU (4.155×10^{-7}) and the solution of highest cost-benefit in terms of ADSF (3.175×10^{-7}) were both located by the Flow Entropy method.

The regression analysis for MOD indicated that there is a statistically significant relationship between all RSMs and their ADS counterparts, except for Flow Entropy versus ADSF, which had a Significance F-value greater than 0.05. The Network Resilience measure demonstrated the highest R^2-values of 0.8075 with respect to ADSU, and 0.8155 with respect to ADSF. The lowest Significance F-values were also generated by the Network Resilience RSM.

The additional performance results appear in Table 8.16. The Mixed Surrogate confirmed its title of solution finder by finding the largest number of solutions, at 145, while Flow Entropy uncovered the fewest, at 51. The Mixed Surrogate produced the solutions with the highest average cost of 4.54×10^6, while Flow Entropy produced the solutions with the lowest average cost of 2.59×10^6. The Mixed Surrogate used the largest average of 307.02 pipes, while Flow Entropy used the fewest, at an average of 284.52 pipes. Flow entropy demonstrated the lowest average SDD and SQDD-values of 17 400 and 1 995 984, respectively. The Mixed Surrogate obtained the best SSDM-value of 0.2610, compared to the worst value of 0.6612 achieved by Resilience Index.

Graphs of the MOD attainment fronts for the various RSMs, along with their corresponding ADSU and ADSF-values indicated on the secondary vertical axis, may be found in Figures 8.43–8.46. The RSM-ADS curves are less well defined than for some of the other benchmarks, but there is a definite positive correlation between all RSM and ADS values.

The RSMs are compared directly in cost-ADSU-space for MOD in Figure 8.47. A now familiar pattern is repeated, with Resilience Index and Network Resilience forming the uppermost region

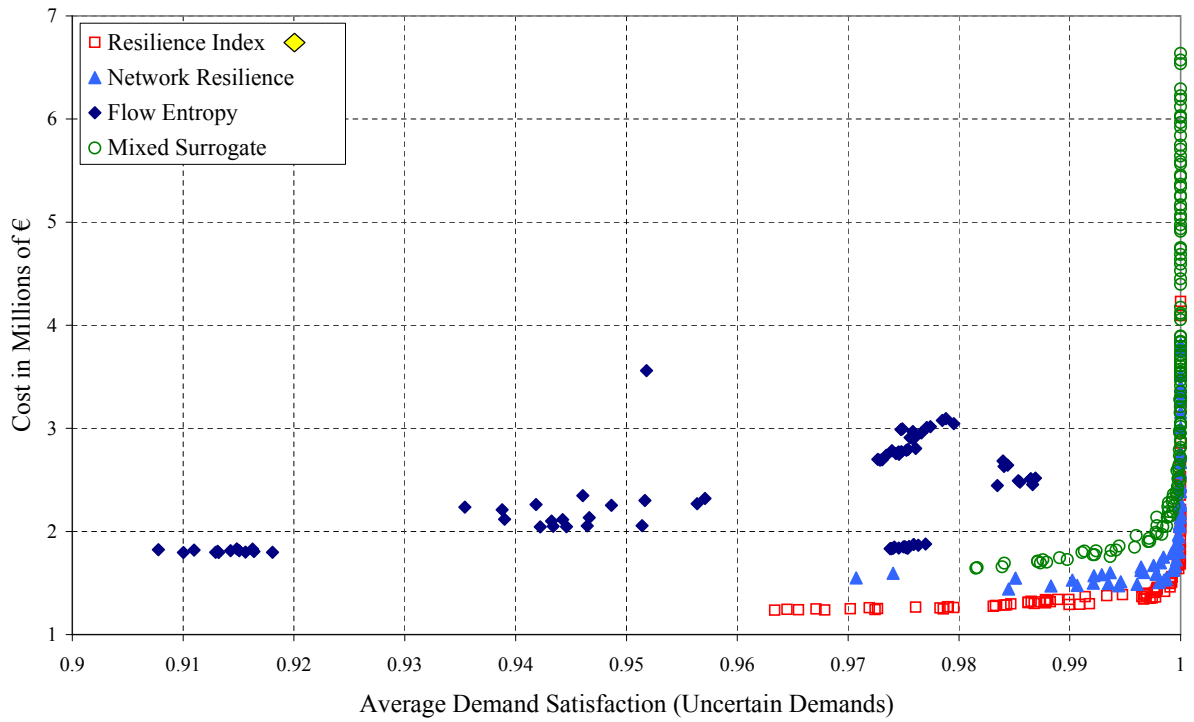


Figure 8.41: *Comparison of RSMs: ADSU vs Cost for the PESC benchmark.*

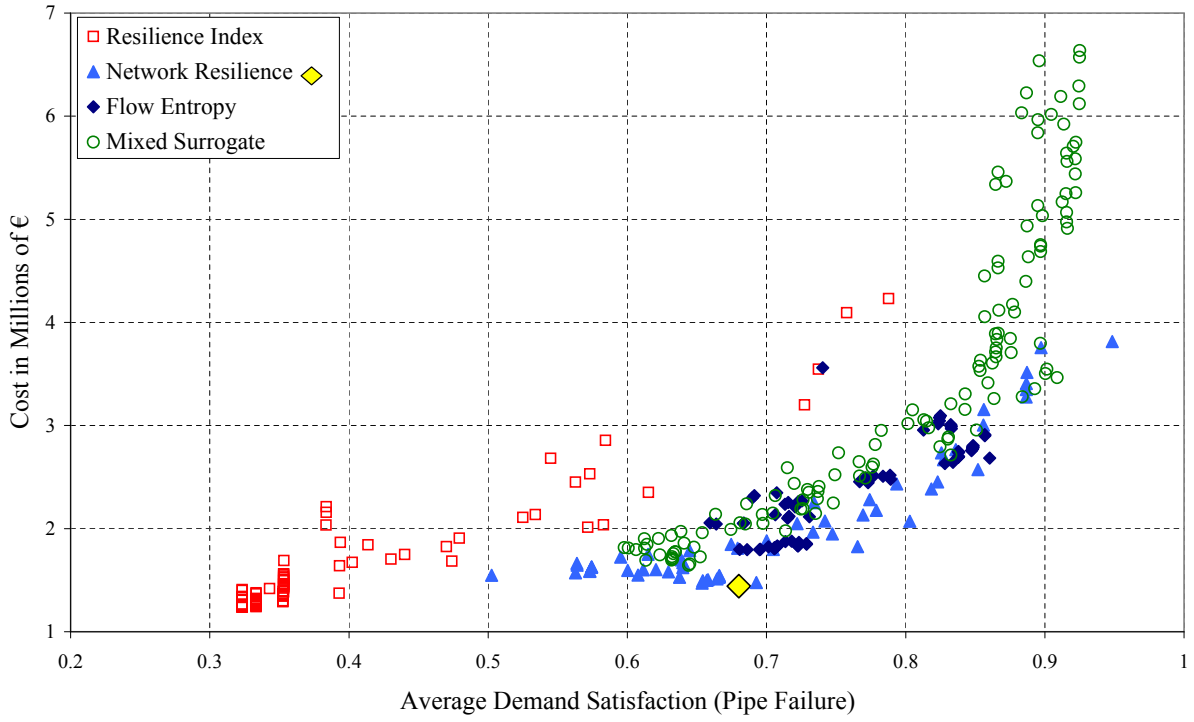


Figure 8.42: *Comparison of RSMs: ADSF vs Cost for the PESC benchmark.*

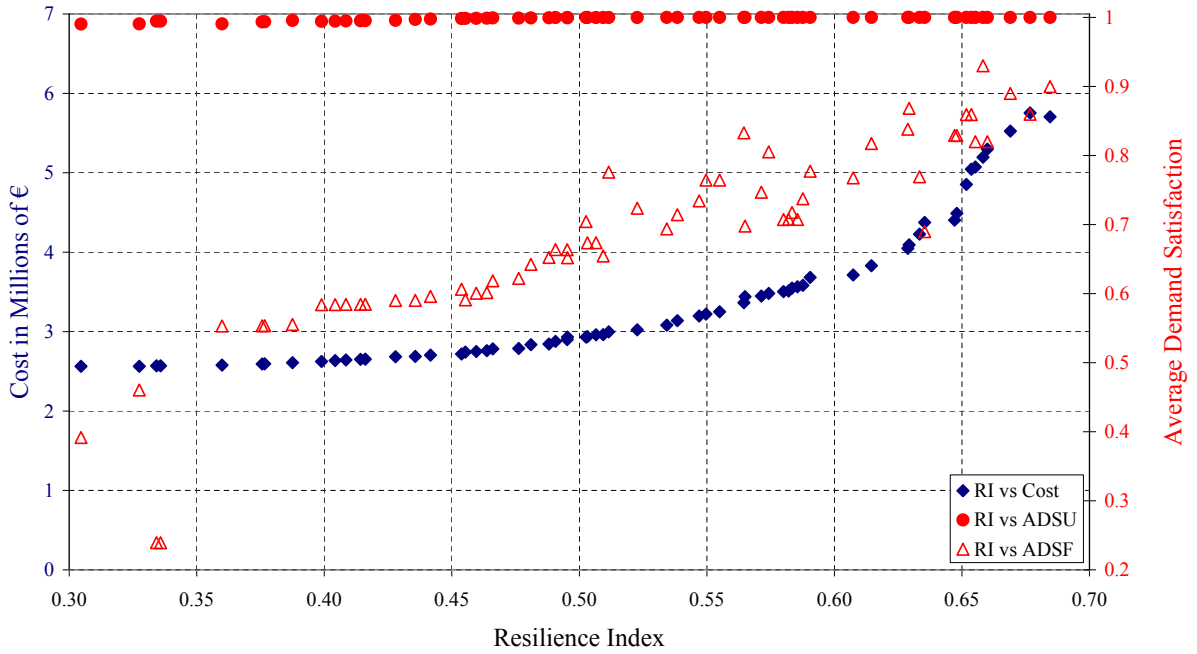


Figure 8.43: *Resilience Index versus ADSU (Uncertainty) and ADSF (Pipe Failure) for the MOD benchmark, with average demand satisfaction ratio on secondary axis.*

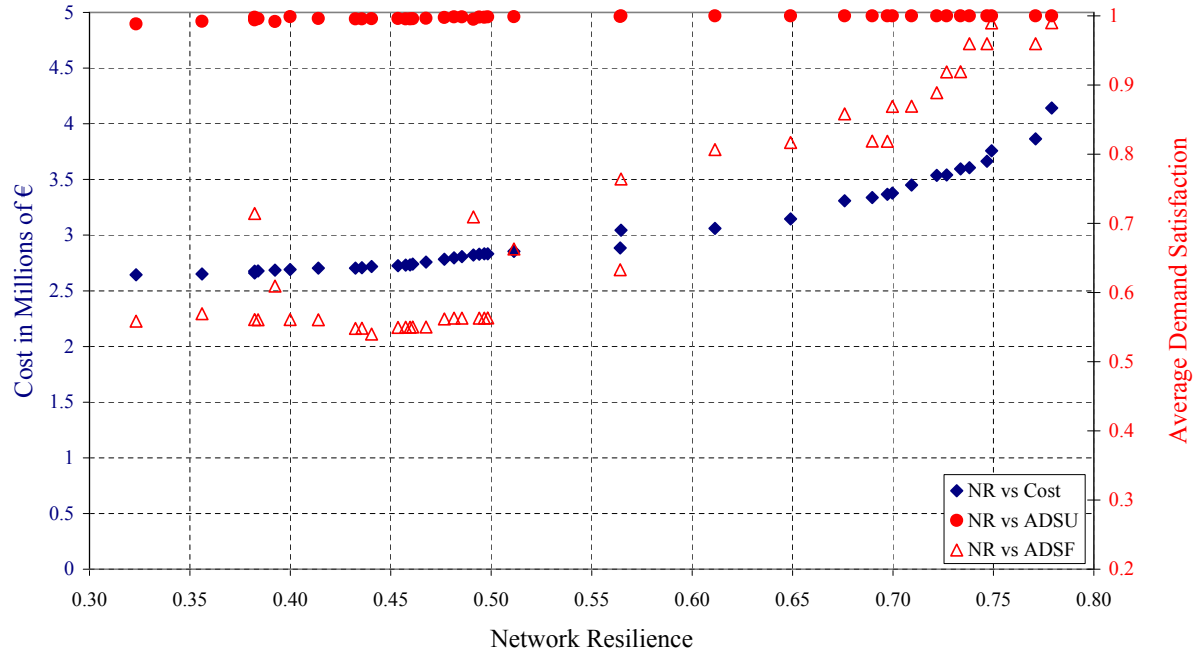


Figure 8.44: *Network Resilience versus ADSU (Uncertainty) and ADSF (Pipe Failure) for the MOD benchmark, with average demand satisfaction ratio on secondary axis.*

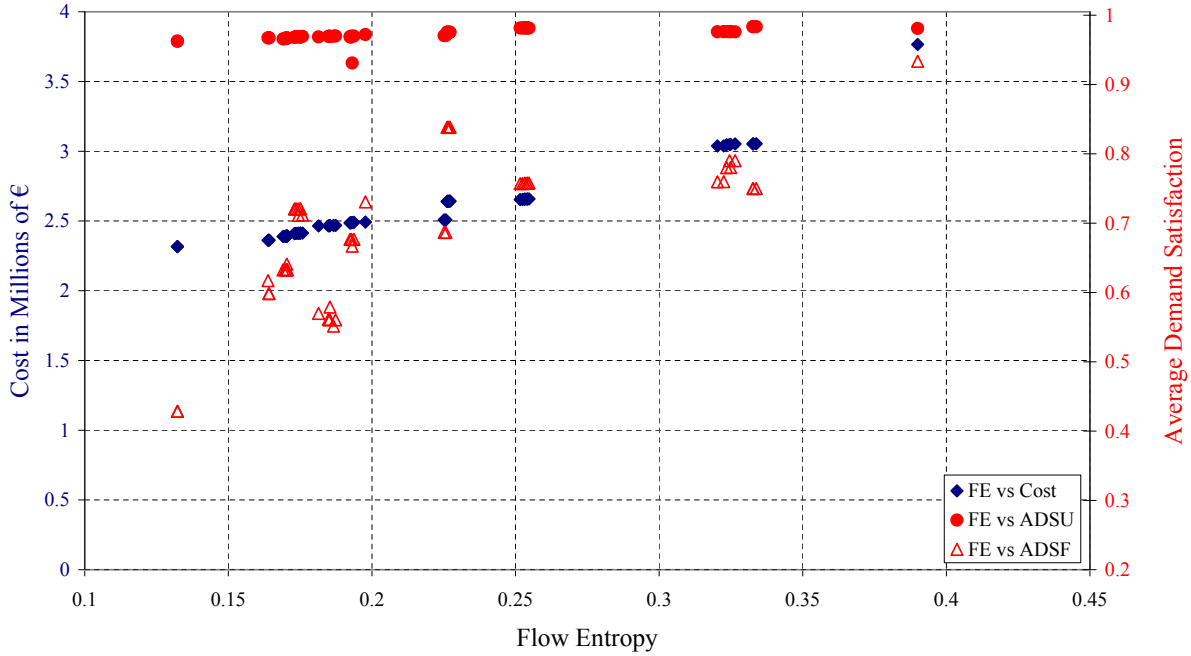


Figure 8.45: *Flow Entropy versus ADSU (Uncertainty) and ADSF (Pipe Failure) for the MOD benchmark, with average demand satisfaction ratio on secondary axis.*

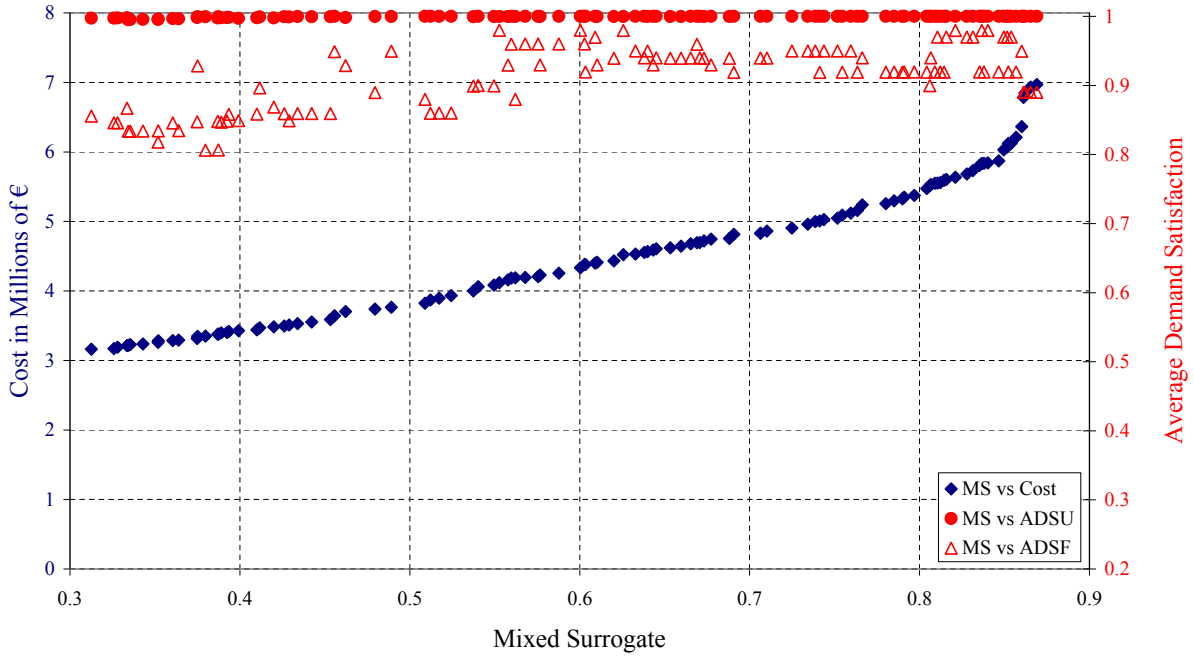


Figure 8.46: *Mixed surrogate versus ADSU (Uncertainty) and ADSF (Pipe Failure) for the MOD benchmark, with average demand satisfaction ratio on secondary axis.*

RSM	Avg ADSU	Mx ADSU	Mx ADSU/Cst	R^2 ADSU	Signif F
Resilience Index	0.9984	1	3.944×10^{-7}	0.7596	2.53×10^{-13}
Network Resilience	0.9977	1	3.766×10^{-7}	**0.8075**	$\mathbf{4.51\times10^{-18}}$
Flow Entropy	0.9718	0.9834	$\mathbf{4.155\times10^{-7}}$	0.2169	5.73×10^{-4}
Mixed Surrogate	**0.9992**	1	3.190×10^{-7}	0.5636	2.00×10^{-16}
RSM	**Avg ADSF**	**Mx ADSF**	**Mx ADSF/Cst**	**R^2 ADSF**	**Signif F**
Resilience Index	0.6847	0.9397	2.589×10^{-7}	0.6673	5.01×10^{-14}
Network Resilience	0.7038	**0.9900**	2.669×10^{-7}	**0.8155**	$\mathbf{5.07\times10^{-27}}$
Flow Entropy	0.6987	0.9336	$\mathbf{3.175\times10^{-7}}$	0.0641	7.31×10^{-2}
Mixed Surrogate	**0.9109**	0.9799	2.773×10^{-7}	0.1456	2.18×10^{-6}

Table 8.15*: Reliability comparisons of RSMs using ADS measures for the MOD benchmark.*

RSM	Count	Avg Cost	Avg Pipes	Avg SDD	Avg SQDD	Avg SSDM
Resilience Index	54	3.42×10^{6}	285.29	28 805	4 582 062	0.6612
Network Resilience	71	3.06×10^{6}	290.05	21 848	2 986 668	0.6126
Flow Entropy	51	2.59×10^{6}	284.52	**17 400**	**1 995 984**	0.4905
Mixed Surrogate	**145**	4.54×10^{6}	307.02	27 037	3 812 980	**0.2610**

Table 8.16*: Result comparisons of RSMs using WDS features for the MOD benchmark.*

of the Pareto-front. The Mixed Surrogate forms a secondary front. Although Flow Entropy lags behind, it uniquely locates solutions along the global Pareto-front in the lower ADSU regions, including the solution of highest ADSU/Cost. The RSMs are compared in cost-ADSF-space for MOD in Figure 8.48. The uppermost region of ADSF (above 0.8) is dominated by Network Resilience and the Mixed Surrogate. Below ADSF-values of 0.84, Flow Entropy dominates completely, including finding the solution of highest ADSF/Cost. The attainment set of Resilience Index is completely dominated.

8.3.9 Summary of Reliability Analysis Results

A summary of the results of the preceding reliability analyses is shown in Table 8.17. The average ADSU and ADSF-values have been computed across all eight benchmarks, as well as the standard deviations thereof. Similarly, the average and standard deviations of R^2-values for ADSU and ADSF are provided. A count of the number of statistically significant positive correlations is also provided (Signif$^+$ Cnt). The general solution characteristics are further summarized in Table 8.18, where the following metrics are provided: the average normalised solution count (AN Cnt)[5]; the average normalised cost (AN Cst)[6]; the average normalised number of pipes (AN Pipes)[7]; and similarly normalized forms for SDD (AN SDD), SQDD (AN SQDD) and average SSDM (Avg SSDM). Furthermore, the average ADSU versus the average ADSF values for each of the RMSs is plotted in Figure 8.49, while the average ADSU R^2-values versus the average ADSF R^2-values for each RSM are plotted in Figure 8.50.

From Table 8.17 it is clear that the Mixed Surrogate RSM provides the best average ADSU and ADSF-values of 0.9912 and 0.8499, respectively, across the eight benchmarks, although Resilience Index and Network Resilience are in close competition. The Mixed Surrogate also provides the smallest standard deviation for ADSU-values, namely 0.0104. This comes at the cost of lower correlation coefficients (R^2-values) of 0.7707 for ADSU and 0.5583 for ADSF,

[5]This is normalised by the largest number of solutions generated for a benchmark.
[6]This is normalised by the highest average cost for each benchmark.
[7]Where this is again normalised by the maximum average number of pipes for each benchmark.

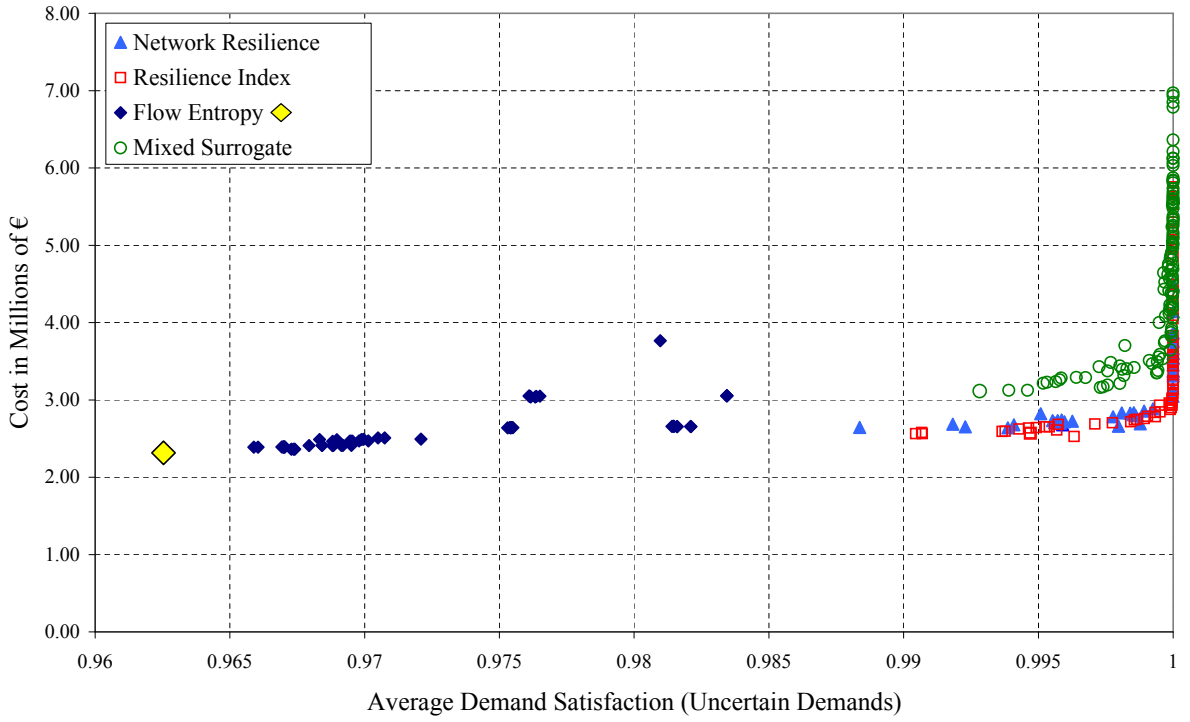


Figure 8.47: *Comparison of RSMs: ADSU vs Cost for the MOD benchmark.*

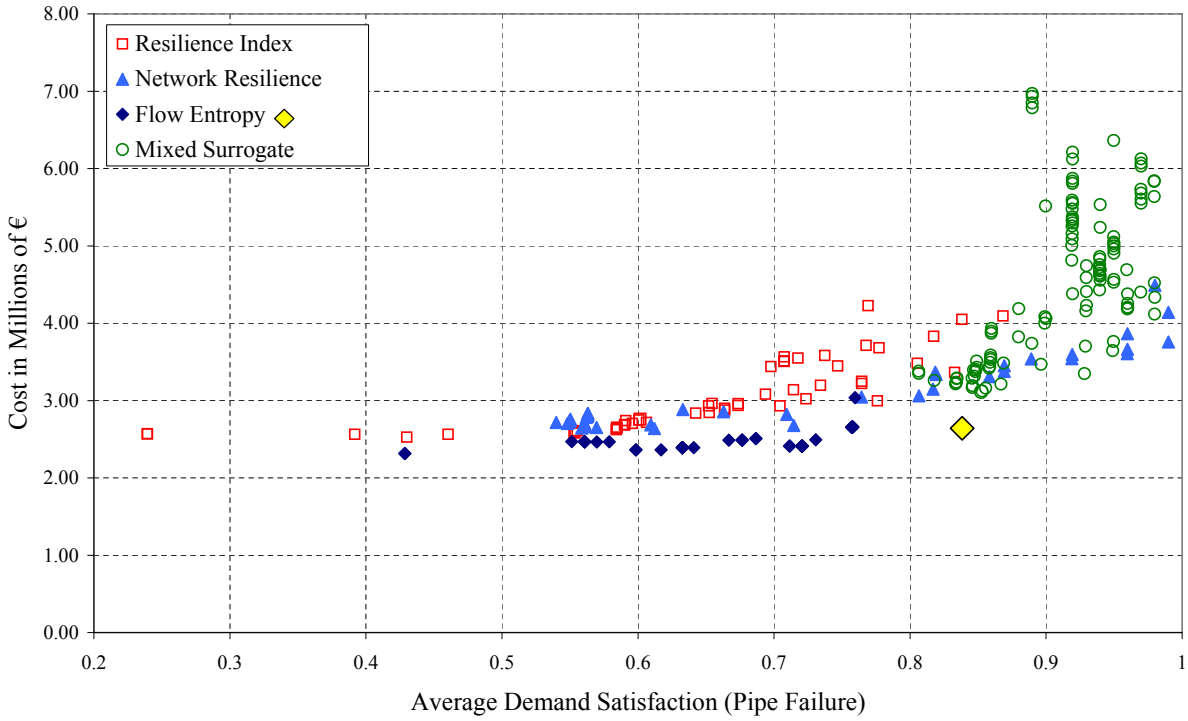


Figure 8.48: *Comparison of RSMs: ADSF vs Cost for the MOD benchmark.*

RSM	Avg ADSU	SD ADSU	Avg R^2 ADSU	SD R^2 ADSU	Signif$^+$ Cnt
Resilience Index	0.9893	0.0106	**0.8265**	**0.1006**	8
Network Resilience	0.9894	0.0112	0.8119	0.2214	8
Flow Entropy	0.9657	0.0297	0.6054	0.3579	7
Mixed Surrogate	**0.9912**	**0.0104**	0.7707	0.1433	8
RSM	**Avg ADSF**	**SD ADSF**	**Avg R^2 ADSF**	**SD R^2 ADSF**	**Signif$^+$ Cnt**
Resilience Index	0.7146	0.2392	0.7539	**0.0918**	8
Network Resilience	0.7945	**0.1391**	**0.8190**	0.1108	8
Flow Entropy	0.7921	0.1807	0.5199	0.3749	6
Mixed Surrogate	**0.8499**	0.1533	0.5583	0.3438	8

Table 8.17*: RSM summary comparison using ADS measures.*

RSM	AN Cnt	AN Cst	AN Pipes	AN SDD	AN SQDD	Avg SSDM
Resilience Index	0.4872	0.7324	0.8857	0.9327	0.8549	0.9305
Network Resilience	0.4790	0.7342	0.9704	**0.7338**	**0.5770**	0.8476
Flow Entropy	0.3494	**0.6885**	0.9848	0.7938	0.6636	0.8828
Mixed Surrogate	**1**	1	0.9958	0.9684	0.9383	**0.6890**

Table 8.18*: RSM summary comparison using network characteristics.*

respectively, indicating that the Mixed Surrogate measure might be less predictive of ADS values than either the Resilience Index or Network Resilience measures. Although the Resilience Index has the highest average R^2-value with respect to ADSU of 0.8265, the Network Resilience measure is not far behind with a value of 0.8119. Both of these values are greater than 0.8, indicating a strong correlation between the RSMs and ADS-values. Resilience Index has the lowest standard deviation for R^2 ADSU of 0.1006. Network Resilience is the RSM with the largest average R^2-value with respect to ADSF of 0.8190, compared to that of 0.7539 achieved by Resilience Index. Network Resilience also comes in second position with regards to ADSF with an average of 0.7945, demonstrates the lowest SD ADSF of 0.1391, and has a very low standard deviation for R^2 ADSF of 0.1108 (close to the best value of 0.0918 achieved by Resilience Index). This places it in the strongest position with regards to being a predictor for pipe failure reliability. This may be confirmed visually in Figures 8.49 and 8.50. If one considers that performance coordinates with the smallest Euclidean distance to the ideal point $(1, 1)$ on each graph are superior, then the Mixed Surrogate and Network Resilience are the best for the two different criteria, ADS and R^2-values. Flow Entropy can easily be disqualified from consideration as a practical RSM due to its relatively worse performance characteristics and failure to show a statistically significant positive correlation with ADS-values in several instances.

Considering the additional performance metrics in Table 8.18, the Mixed Surrogate measure locates more than double the number of solutions than the other RSMs, but these solutions are, on average, substantially more expensive (the solutions produced by Network Resilience achieve a cost of 73.42% the cost of those generated by the Mixed Surrogate, on average), and it produces the worst average normalised SDD and SQDD-values of all the RSMs. Network Resilience employs fewer pipes on average (97.04% of the maximum number, compared to 99.58% for the Mixed Surrogate), and outperforms all other RSMs convincingly in terms of average normalised SDD and SQDD-metrics. In particular, the relatively low SQDD-value of 0.5770 indicates that Network Resilience is much better than the other methods at avoiding large diameter differences. This makes it the most practical RSM in terms of real-world engineering. The Resilience Index generally produced solutions comprising far fewer pipes (88.57% of the maximum), which translates to reduced pathway redundancy and lower pipe failure reliabil-

ity. Network Resilience comes in second place after the Mixed Surrogate in terms of average SSDM, beating Flow Entropy which is expected to perform the best here since it intrinsically rewards balanced distribution. Resilience Index produces the worst SSDM-values. However, it is suggested that the number of multi-reservoir WDS cases is too few to make any general conclusions on this matter. One vital piece of information not provided in this summary is the dominance information in ADS-Cost space, where the general trend was that the Network Resilience results dominated the Mixed Surrogate results for both ADSU and ADSF. For these reasons, the author recommends Network Resilience as the RSM of choice for WDSDO, where the maximisation of a RSM is incorporated during optimization.

8.4 Chapter Summary

In this chapter, a comparison was conducted on the WDS RSMs of Resilience Index, Network Resilience, Flow Entropy, and the novel Mixed Surrogate measure, with respect to the eight WDS benchmarks TRP, TLN, HANOI, NYTUN, BLACK, FOSS, PESC and MOD. Thirty optimization runs were used to produce attainment approximation sets for each surrogate measure and each benchmark, and stochastic demand and failure reliability were calculated via simulation for these designs, using the demand satisfaction measures ADSU and ADSF. This analysis was performed in fulfilment of Dissertation Objective 9 in §1.3. In order to conduct this analysis, the implementation of this optimisation model in a software library was completed in final fulfilment of Dissertation Objective 10.

Regression analysis was used to investigate the relationship between the RSMs and their ADS reliability equivalents. While they were often strongly correlated ($R^2 > 0.8$) this only occurred consistently enough for the Network Resilience RSM such that its average R^2-values for ADSU and ADSF were both larger than 0.8, suggesting the general hypothesis of a linear relationship between the Network Resilience and the ADS measures.

It is desirable to search for WDS solutions with fewer and less severe discontinuities of size between adjacent pipes. This was quantified by summing the pipe diameter differences at all the nodes of a design (SDD), as well as by means of the sum of the squared diameter differences (SQDD). On average across the eight benchmarks, Resilience Index, Network Resilience, Flow Entropy and the Mixed Reliability measures yielded average SDD-values which were 93.27%, 73.38%, 79.38% and 96.84% of the maximum obtained for that benchmark, respectively. Having lower SDD-values provides significantly smoother pipe gradients, which are desirable in practise, in accordance with the objectives of reliable loops and flow uniformity set out in the derivations of these RSMs.

Although it is difficult to make general conclusions given a sample of only eight benchmarks, the following observations could be made:

- The Resilience Index appears to dominate the other RSMs in terms of yielding solutions reliable under pure stochastic demand variation, which agrees intuitively with its primary goal of maximizing excess system power, and its lack of limiting criteria.
- However, the Resilience Index performed relatively poorly in comparison to both Network Resilience and the Mixed Surrogate when it came to pipe failure analysis, which accords with their goal of obtaining redundancy through reliable loops.
- The Flow Entropy measure performed the worst overall, although it did yield some surprises. It is advised that Flow Entropy be used for WDS design only in combination with

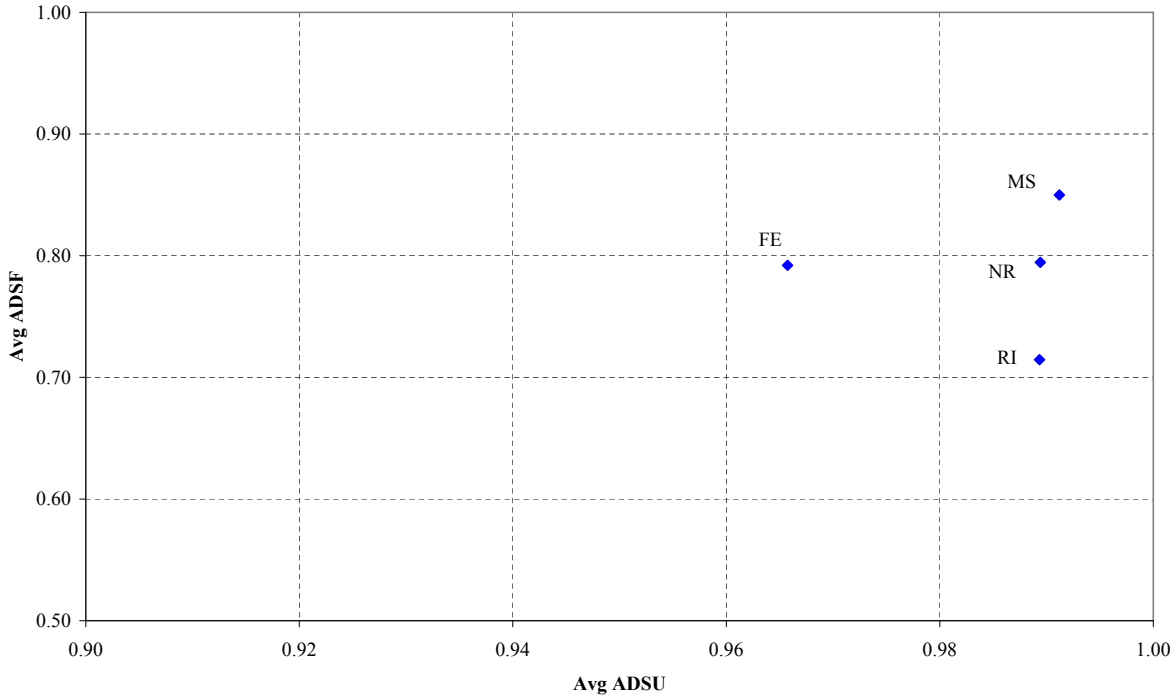


Figure 8.49: *ADS values for each RSM averaged across eight benchmarks.*

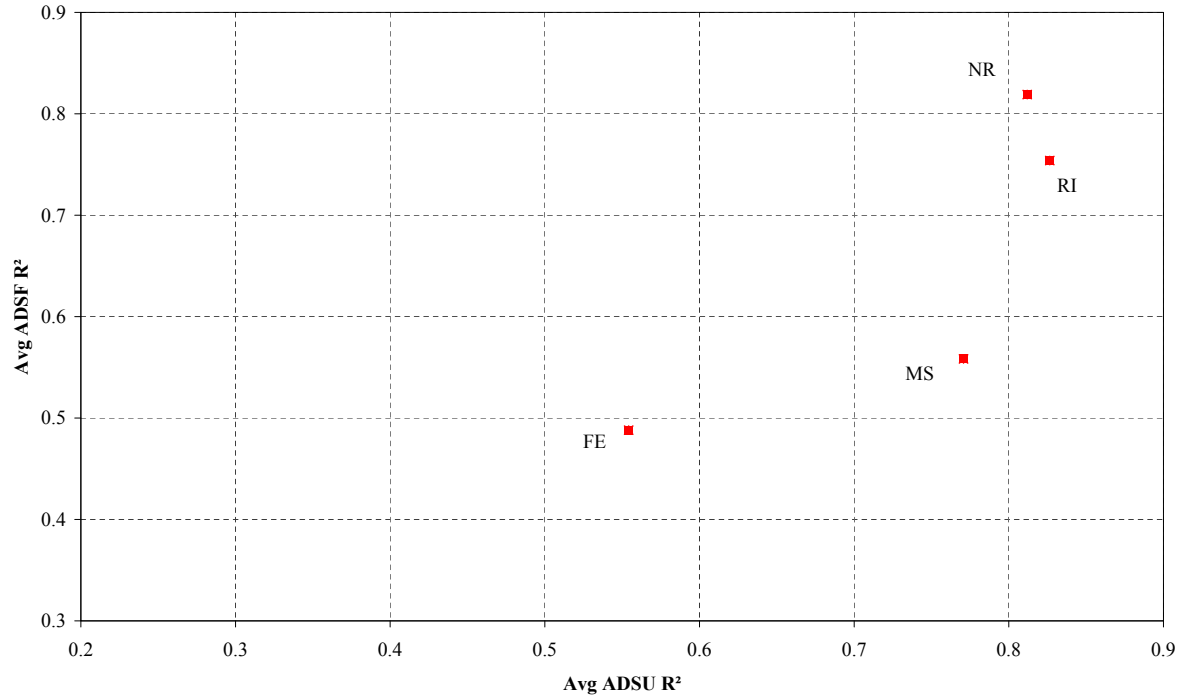


Figure 8.50: *ADS R^2 values for each RSM averaged across eight benchmarks.*

other RSMs.

- Network Resilience demonstrated very good performance with respect to both ADS measures, performing well under stochastic demands and pipe failure conditions, and in most cases dominated the other RSMs in ADSF-cost-space.
- The solutions generated using Network Resilience have the desirable properties of being less costly, on average, having greater pipe redundancy, and having superior pipe adjacency characteristics (lower SDD and SQDD-values), whereby large size discontinuities are avoided. This suggests that Network Resilience may be the most practical RSM for use in general WDS design.

Although the use of RSMs may be able to save an order of magnitude in terms of processing speed compared to traditional stochastic probabilistic reliability quantification, it is not yet clear whether they will provide sufficient quality of results compared to actually incorporating stochastic demands and failure analysis into the design optimization process. Future studies should compare the performance of RSMs to the best stochastic algorithms (such as the RNSGA-II (Kapelan *et al.* 2005) which uses LHS).

Chapter 9

The R21 Corridor WDS – A South African Case Study

The South African case study for this dissertation is a new WDS development project in Ekurhuleni (East Rand, Gauteng Province) called the R21 Corridor development area, where currently no bulk water distribution infrastructure exists. An aerial photograph of the R21 Corridor development area prior to the development appears in Figure 9.1. The objective of this problem is to design the bulk infrastructure (the mains pipes) for a *gravity WDS* (*i.e.* one without pumps), which will supply water to new residential and industrial areas, in order to minimize costs and maximize reliability.

9.1 Introduction

The basic network layout and technical information for the R21 WDS has been supplied by GLS Software (Pty) Ltd [103][1]. There is a single reservoir whose capacity has been designed separately and is taken as given [220]. The design guidelines for hydraulic limits and demand scenarios are taken from the Johannesburg Water and Tshwane Municipality guidelines (including the SANS 100090 standard for fire fighting).

The proposed system comprises 82 pipes, which makes it larger than the majority of the WDS benchmarks in the literature [157]. It has been augmented by an additional 7 pipes to create a practical, redundant layout, yielding a total of 89 pipes. The WDS layout is shown in Figure 9.2, with the 7 additional pipes represented as dotted lines. The pipes have been numbered from 1 to 89. Four nodes have also been identified by means of circled numbers 1 to 4. Two regions of interest have been magnified to show detail. Region A represents a pipe connecting two nodes which crosses a road, incurring an additional R100 000 expense. Region B represents the two unconnected pipe sections leaving the reservoir, supplying water to two major sub-networks, which may be isolated by the removal of the pipe in region A. If this pipe were to be removed, no node would have more than a single independent path to the source.

[1] A full problem specification for the R21 Corridor benchmark is available online [199].

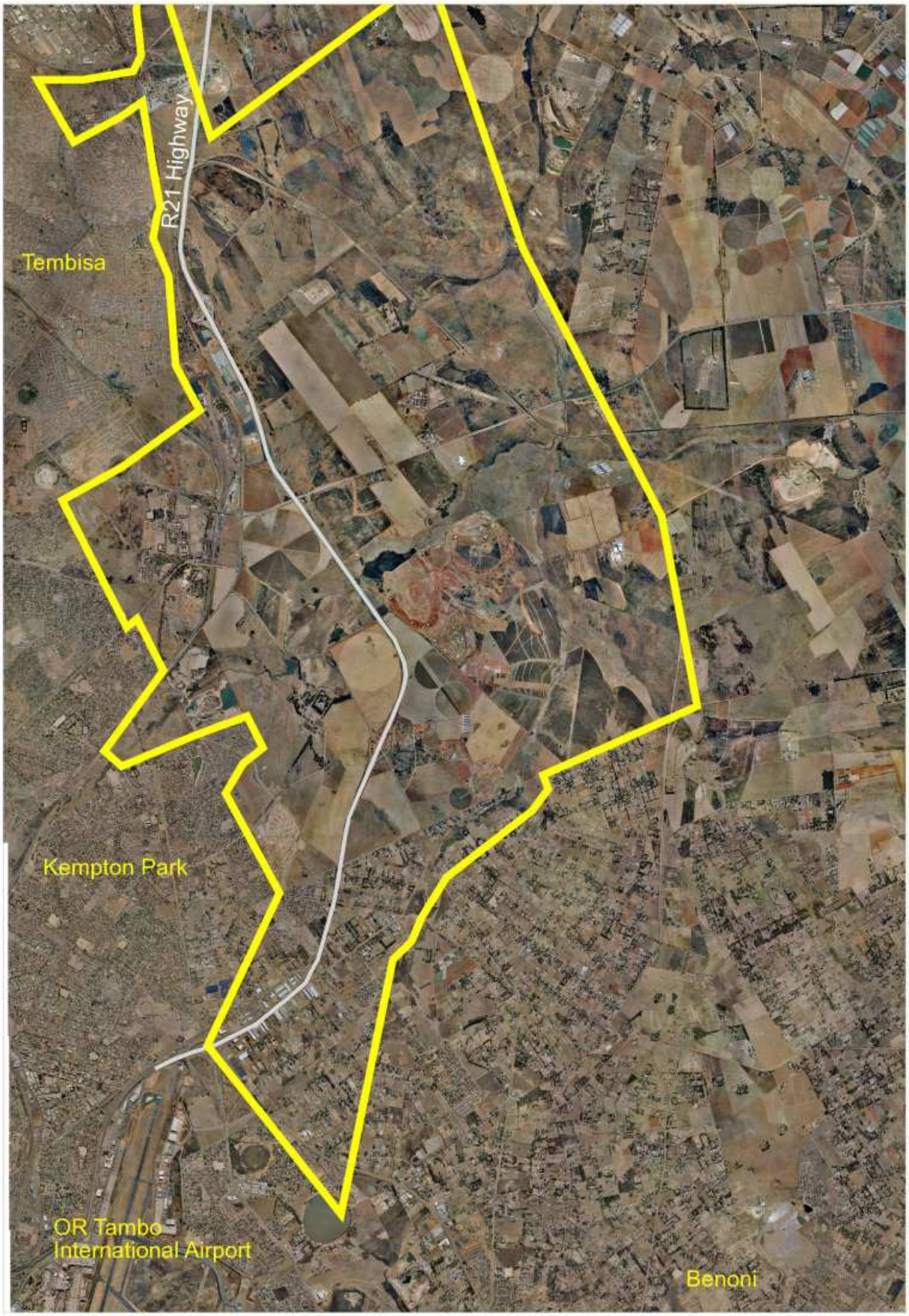


Figure 9.1: *Aerial map of the R21 Corridor development area [103].*

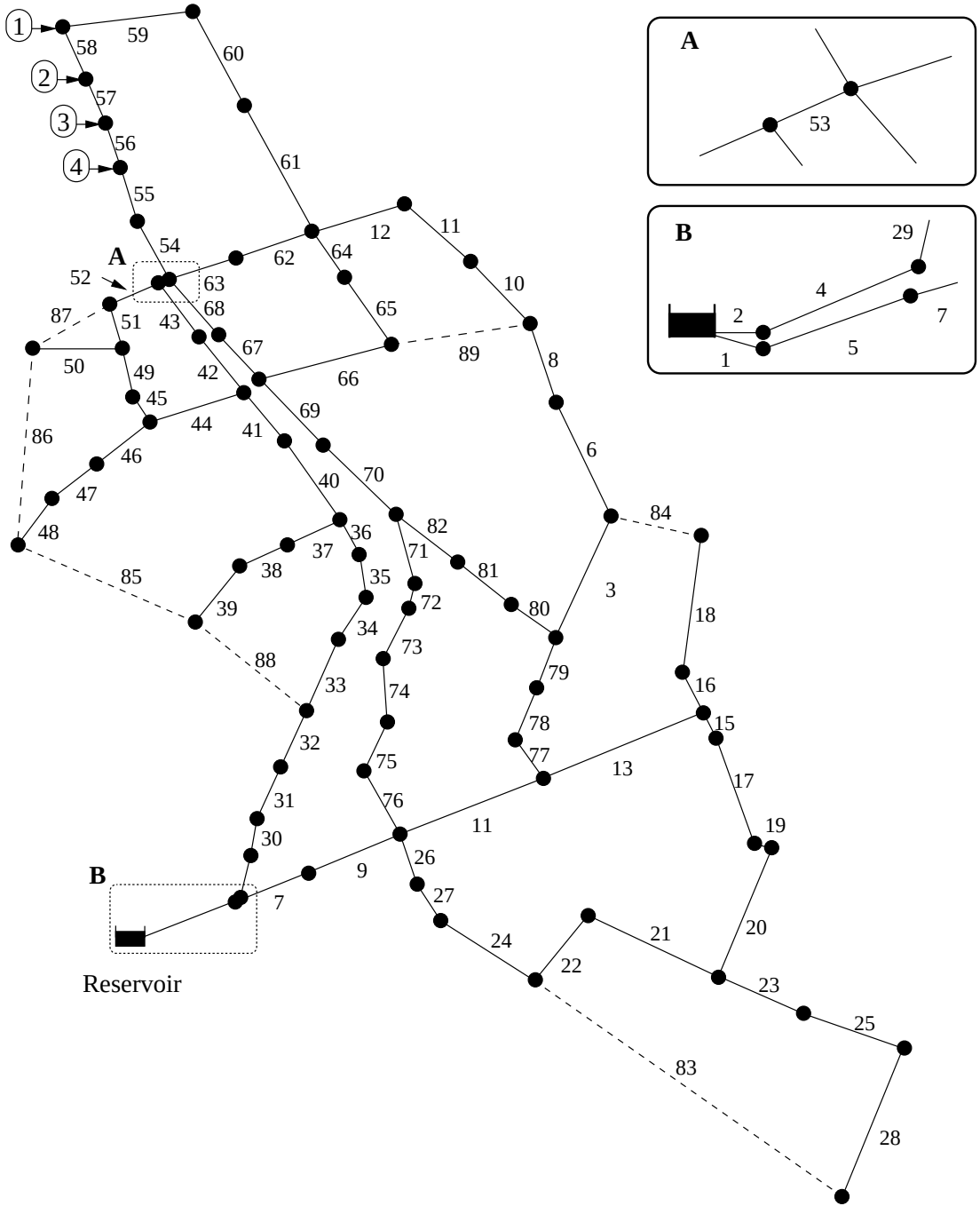


Figure 9.2: *Pipe layout for the R21 Corridor WDS case study [103].*

9.2 Pipe Sizing Options

Since each pipe can be assigned one of 26 diameter options (see Table 9.1) or the option of elimination, the size of the search space is $27^{89} \approx 10^{127}$. The pipe option costs in Table 9.1 cite both unit cost in South African Rands per meter pipe length, and the connection cost, which must be applied once-off at each junction where pipes intersect, using the value of the pipe with the largest diameter.

Diameter (mm)	Cost (R/m)	Connection Cost (R)	Diameter (mm)	Cost (R/m)	Connection Cost (R)	Diameter (mm)	Cost (R/m)	Connection Cost (R)
127	263	31 000	530	1 684	166 000	976	3 539	387 000
145	293	36 000	574	1 832	185 000	1 074	4 564	446 000
182	374	46 000	626	2 228	207 000	1 176	5 078	511 000
227	500	59 000	675	2 346	230 000	1 366	7 599	643 000
286	714	77 000	726	2 557	254 000	1 568	9 551	798 000
322	869	89 000	777	2 687	279 000	1 773	10 634	971 000
363	1 058	103 000	828	3 060	306 000	1 970	13 426	1 153 000
428	1 353	126 000	878	3 062	332 000	2 174	14 688	1 356 000
479	1 472	146 000	929	3 335	361 000	—	—	—

Table 9.1*: Pipe internal diameter options together with unit costs and connection costs for the R21 Corridor WDS.*

9.3 Water Demand Loading Conditions

Three types of demand nodes are considered, classified as *industrial*, *mixed* and *residential*, each with different average demands and minimum head specifications. Different demand loading conditions were considered, each employing typical expected values of the *average annual daily demand* (AADD) of the different node classes (multiplied by the area of the region being serviced by the node), and factors by which the hourly demand is multiplied. Two typical 24-hour demand patterns with hourly factors were utilised simultaneously; one for residential zones and another for the other zones. The peak hourly factor is the maximum of these multipliers, typically taken as 4 for residential zones. The following demand loading scenarios are applicable:

1. A 24-step hourly time series using typical demand daily patterns for residential and industrial zones at a residential peak factor of $4 \times$ AADD/24. The residential zone demand multipliers are

$$\mathcal{M}^1 = \{0.6, 0.6, 0.6, 1, 1.4, 1.8, 3, 4, 3, 2.4, 1.8, 1.6, 2, 2, 2.6, 3, 3.2, 3.2, 3.4, 2.2, 1.6, 1.4, 1, 0.6\}\,.$$

   The industrial and mixed zone demand multipliers are

$$\mathcal{M}^2 = \{1, 1.2, 1.2, 1.2, 1.4, 2, 2.6, 2.8, 3, 2.8, 2.6, 2.6, 2.6, 2.6, 2.8, 3, 3, 2.4, 2, 1.2, 1, 1, 1, 1\}\,.$$

2. A static flow condition (zero-flow at all demand zones) to derive the maximum nodal pressures.

3. An industrial fire scenario using 25 ℓ/s additional water from each of 4 hydrants simultaneously at a peak demand factor of $2 \times$ AADD/24. This is applied at the nodes labeled 1–4 in Figure 9.2.

However, since no tank design is required for this system in the current problem formulation, a designer need only employ the most extreme demand loadings during optimisation. This would include the two mutually non-dominated periods of peak hourly demand, namely the first period having demand multipliers of 3.2 and 3 for residential and industrial demand, respectively, and the second period having demand multipliers of 4 and 2.8 for residential and industrial demand, respectively. In combination with the static flow condition (*maximum pressure scenario*), and the fire flow scenario under peak average day (*asymmetric emergency demand scenario*), this yields a total of four demand loading conditions used in designing the R21 Corridor WDS.

9.4 Hydraulic Parameters

The various applicable hydraulic parameters appear in Table 9.2. A maximum velocity limit of 2.7 m/s (8.8 ft/s)is used as per the recommendation in [136]. Minimum head limits of 25, 30 and 35 m are used for the residential, mixed and industrial nodes, and a maximum head limit of 90 m is applicable to all nodes. All pipes have a Darcy-Weisbach absolute roughness coefficients of 0.025 mm.

Parameter	**Value**	**Description**
$v_{\max}$	2.7 m/s	Max pipe velocity
$h_{\max}$	90 m	Max head for all nodes
$h^1_{\min}$	25 m	Min head for all residential nodes
$h^2_{\min}$	30 m	Min head for all mixed nodes
$h^3_{\min}$	35 m	Min head for all industrial nodes
AADD^1	$\pm$12 (kℓ/ha/day)	AADD residential demand
AADD^2	$\pm$18 (kℓ/ha/day)	AADD mixed demand
AADD^3	$\pm$20 (kℓ/ha/day)	AADD industrial demand
D-W	0.025	Pipe Darcy-Weisbach coefficient values

Table 9.2*: Hydraulic parameter values for the R21 Corridor WDS.*

9.5 Setup and Optimisation Parameters

As mentioned, an initial configuration for the R21 Corridor case study was supplied by an engineer at GLS Software (Pty) Ltd [103] as an input to the optimisation routine, and was included as an individual in the original population P_0. This input configuration is a preliminary design (excluding the additional pipes) developed by means of the human-assisted partial enumeration method (PEM) [99, 103]. This design has a cost of R72 100 093 and a minimum Network Resilience of 0.507 495. However, because it was designed for less stringent pressure limits, this solution is actually not feasible in the context of this implementation, since it experiences minimum head constraint violations at seven nodes, with a maximum nodal head deficit of 5.878 m.

The optimisation parameter values used were a population size of 128, an empirically derived penalty factor $\alpha_p = 30\,156\,608\,915$, epsilon precisions of $\epsilon_{\text{C}} = 500\ 000$, and $\epsilon_{\text{R}} = 0.001\,250$, and a hypervolume reference point of (0,2 000 000 000).

9.6 Optimisation Trial Runs

An initial hypervolume convergence test was conducted with five optimisation runs, using a convergence criterion of less than 0.05% improvement in hypervolume for 200 consecutive generations, considering objectives of maximising minimum Network Resilience and minimising cost. This yielded an average time to convergence of 12.593 minutes, and produced the attainment front in cost-Network Resilience space for the combined five optimisation runs (labeled "Short Run") that appears in Figure 9.3. An average normalised hypervolume of 0.821 along with a standard deviation of 0.040 was achieved, showing reasonably consistent performance.

A longer set of five optimisation runs was then conducted, using a convergence criterion of less than 0.05% improvement in hypervolume for 1 000 consecutive generations. This yielded good improvements over the short run, in an average time to convergence of 57.733 minutes, and produced the superior attainment front (labeled "Long Run") in Figure 9.3. An average normalised hypervolume of 0.879 along with a standard deviation of 0.017 was achieved, demonstrating very consistent performance.

The objective function vector of the preliminary design appears in Figure 9.3 as a cross, clearly showing that AMALGAMS$_{\text{ndp}}$ produces superior results in a relatively short time. It should be emphasized again that this original design is actually infeasible for this constraint set, while the solutions produced by AMALGAMS$_{\text{ndp}}$ are all feasible.

Two AMALGAMS$_{\text{ndp}}$ solutions are highlighted and marked with their minimum Network Resilience values. The solution with a reliability of 0.516 646 (Alternative Design 1) incurs a cost of R60 222 900, resulting in a significant saving of R11 877 193 for a feasible design with a slightly higher reliability than the preliminary design. The solution with a reliability of 0.745 693 (Alternative Design 2) incurs a cost of R71 856 500 and is the most reliable solution found with a cost less than that of the preliminary design, for a significant increase in Network Resilience.

These three configurations are presented for comparison in Tables 9.3–9.5, which show the pipe internal diameter assignments. Note that pipes 83–89 are the 7 additional pipes which all have a zero diameter in the preliminary solution (Table 9.3). In Alternative Design 1, non-zero diameters have been assigned to additional pipes 84, 85, 86, 87 and 89, and pipes 17 and 66 have been eliminated. In Alternative Design 2, non-zero diameters have been assigned to the same additional pipes (similar or larger diameters than in Alternative 1) and no pipes have been eliminated. It is interesting to note that pipe 53 in region A has not been eliminated in any of the solutions produced, despite the additional cost incurred, thus ensuring that the bulk of the nodes have at least two independent paths to the source. Both alternative designs have larger diameters for this pipe. Furthermore, the pipes 83 and 88 are not cost-effective enough to be included in the solutions found in the lower Network Resilience region of the Pareto Front, but have nonzero diameters for some of the expensive solutions of high Network Resilience.

9.7 Summary of Results

In summary, it was demonstrated that AMALGAMS$_{\text{ndp}}$ is able to improve rapidly and substantially upon a preliminary engineered design for the R21 Corridor WDS, both in terms of cost and reliability (a significant saving of 16.47% of the project cost (*i.e.* R11 877 193) was achieved by Alternative Design 1). Furthermore, AMALGAMS$_{\text{ndp}}$ was quickly able to find feasible solutions. The technique of design with redundant layouts proved fruitful in providing alternative layouts for the system. The duration of optimisation is sufficiently short for such a

medium-sized WDS that AMALGAMS$_{\text{ndp}}$ may easily be employed in practice.

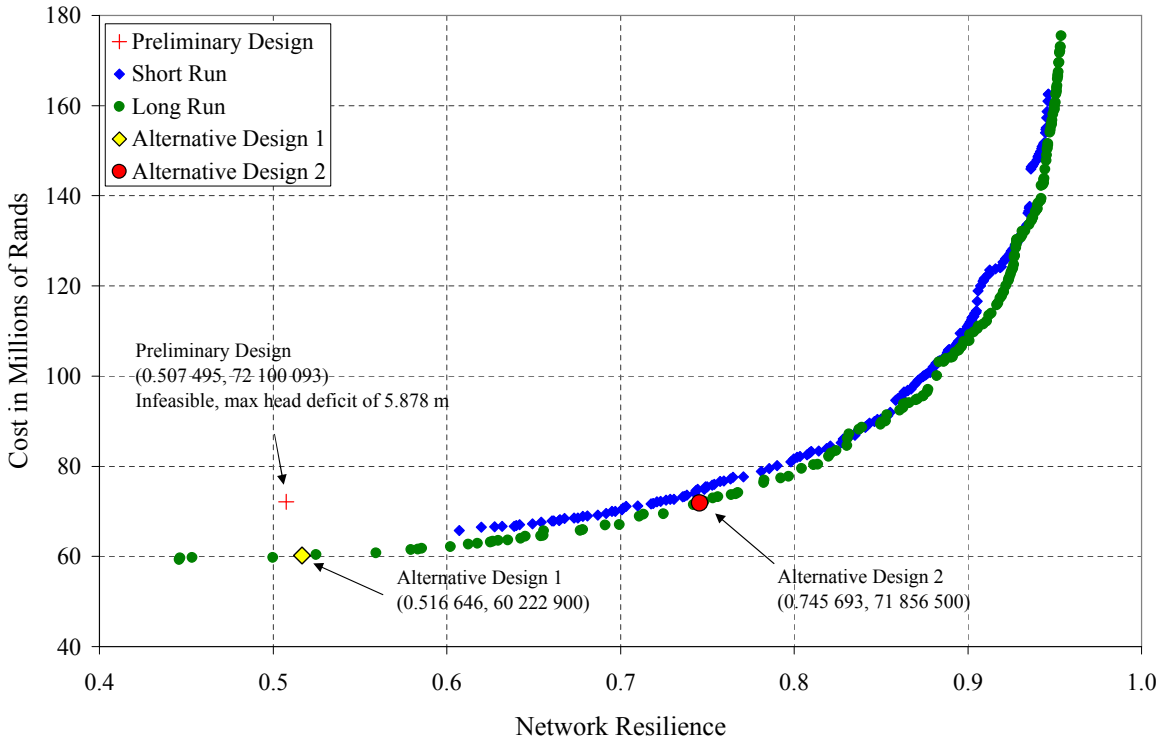


Figure 9.3: *Results obtained by AMALGAMS$_{\text{ndp}}$ for the R21 Corridor WDS case study. Aggregated results from five short runs and aggregated results from five long runs [103].*

ID	Diam	ID	Diam	ID	Diam	ID	Diam	ID	Diam	ID	Diam
1	726	16	227	31	726	46	227	61	286	76	675
2	976	17	182	32	726	47	227	62	322	77	675
3	322	18	227	33	726	48	227	63	322	78	675
4	726	19	182	34	726	49	322	64	227	79	675
5	976	20	182	35	726	50	227	65	227	80	675
6	322	21	675	36	726	51	322	66	227	81	675
7	976	22	675	37	227	52	322	67	322	82	675
8	322	23	227	38	227	53	428	68	322	83	0
9	976	24	675	39	227	54	286	69	322	84	0
10	322	25	227	40	726	55	286	70	322	85	0
11	777	26	675	41	726	56	286	71	675	86	0
12	322	27	675	42	322	57	286	72	675	87	0
13	286	28	227	43	322	58	286	73	675	88	0
14	322	29	726	44	322	59	286	74	675	89	0
15	182	30	726	45	322	60	286	75	675	-	-

Table 9.3*: R21 Corridor pipe diameter assignment for the preliminary design with* $I_{\text{n}} = 0.507\,495$ *and* $C = \text{R}72\,100\,093$.

ID	Diam	ID	Diam	ID	Diam	ID	Diam	ID	Diam	ID	Diam
1	929	16	227	31	777	46	145	61	227	76	574
2	976	17	0	32	777	47	182	62	363	77	574
3	363	18	227	33	777	48	145	63	322	78	530
4	777	19	227	34	726	49	145	64	227	79	479
5	976	20	227	35	777	50	145	65	227	80	286
6	286	21	479	36	777	51	145	66	0	81	227
7	976	22	479	37	286	52	227	67	182	82	227
8	286	23	322	38	227	53	574	68	145	83	0
9	976	24	574	39	182	54	428	69	182	84	227
10	286	25	227	40	626	55	363	70	182	85	145
11	479	26	574	41	675	56	322	71	227	86	127
12	286	27	574	42	675	57	322	72	363	87	145
13	286	28	227	43	574	58	227	73	363	88	0
14	227	29	777	44	227	59	182	74	363	89	182
15	286	30	777	45	182	60	182	75	530	-	-

Table 9.4*: R21 Corridor pipe diameter assignment for Alternative Design 1, with* $I_{\text{n}} = 0.516\,646$ *and* $C = \text{R}60\,222\,900$.

ID	Diam	ID	Diam	ID	Diam	ID	Diam	ID	Diam	ID	Diam
1	878	16	363	31	878	46	182	61	227	76	574
2	976	17	286	32	878	47	145	62	363	77	574
3	363	18	363	33	878	48	145	63	363	78	574
4	878	19	286	34	777	49	145	64	227	79	530
5	976	20	227	35	777	50	145	65	227	80	286
6	286	21	479	36	878	51	145	66	227	81	286
7	976	22	479	37	286	52	227	67	227	82	227
8	286	23	322	38	227	53	626	68	227	83	0
9	976	24	574	39	227	54	428	69	227	84	363
10	286	25	286	40	878	55	479	70	286	85	145
11	878	26	574	41	777	56	363	71	322	86	127
12	286	27	574	42	675	57	286	72	363	87	182
13	530	28	227	43	777	58	322	73	428	88	0
14	286	29	777	44	286	59	227	74	428	89	227
15	286	30	878	45	182	60	227	75	530	-	-

Table 9.5*: R21 Corridor pipe diameter assignment for Alternative Design 2, with* $I_{\mathrm{n}} = 0.745\,693$ *and* $C = \mathrm{R}71\,856\,500$.

Chapter 10

Conclusion

This dissertation was concerned with the topic of multi-objective water distribution systems design optimisation using metaheuristics towards the objectives of minimizing cost and maximizing surrogate reliability. The primary problem examined was that of cost-effective pipe diameter specification for WDSs, allowing some scope for layout modification, in order to satisfy the expected consumer demands within the required pressure limits. An overview of the dissertation contents and the conclusions of the study are presented in this chapter. A presentation of the main dissertation contributions is also provided, followed by an appraisal of these contributions.

10.1 Dissertation Summary

A brief introduction to the topic of WDS design was provided in Chapter 1, establishing the context for multi-objective WDS design optimization by means of metaheuristics and providing a motivation for the study. The objectives of the dissertation were outlined in §1.3. These objectives were fulfilled in the ensuing chapters. The first chapter concluded with an outline of the organisation of material contained in the dissertation.

The second chapter contained a review of the required fluid mechanics theory for WDS analysis, including preliminary hydraulics concepts in §2.1 (such as pressure, flow, the control volume approach, hydraulic continuity and energy equations, pipe hydraulics, head loss, flow in simple networks, and transient analysis), and the topic of hydraulic systems theory in §2.2, discussing the various methodologies for conducting hydraulic simulation by solving the nonlinear system of hydraulic equations (*e.g.* the Hardy Cross method, the Gradient Algorithm, the Linear Theory Method, and pressure driven analysis). These hydraulic systems simulation methods were then compared, and WDS model calibration and implementation were discussed. The public domain hydraulic simulator EPANET2 was selected as the software used for this study. This chapter was included in fulfilment of Dissertation Objective 1.

Chapter 3 constituted a broad discussion of the WDS design optimization problem. It began in §3.1 by considering the WDSDO model application — whereby a model is developed, calibrated, and alternative designs are tested as part of an optimisation exercise — and introducing various techniques that are applicable in this context. The WDS benchmarks to be analyzed in this dissertation were also introduced here. An overview of the applicable optimisation methods was presented next in §3.2, including exact methods, heuristics, metaheuristics and hyperheuristics; and the optimization-simulation framework for WDSDO was illustrated. A generic mathemat-

ical formulation of the WDS design problem in terms of least-cost optimisation was given in §3.3, catering for the design of all standard WDS components and allowing for constraints on hydraulic performance. Thereafter a discussion of practical design considerations for WDSs followed in §3.5, including the existence of uncertainty in water demands and infrastructure costs, the necessity of over-designing a WDS, the inclusion of running costs and staged development in the problem formulation, the requirement of extended period analysis for pump schedule and tank design, designing for fire-flows, leakage, pipe failures, and the enforcement of reliable loops for redundancy in the network. These initial sections were provided in partial fulfilment of Dissertation Objective 2.

A concise history of the WDSDO problem was provided in §3.6, detailing the development of the problem from early simple computer models using dynamic and linear programming, to advanced multi-objective design paradigms and metaheuristics such as MOEAs. An in-depth survey of single-objective optimisation methods used previously for solving the least-cost WDSDO problem was included in §3.7. The methods used for least-cost optimisation were discussed in some detail, and included Partial Enumeration, Linear and Nonlinear Programming, Simulated Annealing, Tabu Search, Genetic Algorithms, Ant Colony Optimisation, Shuffled Complex Evolution, Particle Swarm Optimisation, and the Shuffled Frog Leaping Algorithm. However, it was noted that the least-cost design paradigm has been invalidated by experts in the field, due to the lack of robustness of the resulting designs, and that there has been a paradigm shift to the requirement of achieving some acceptable trade-off between system costs and benefits. These final two sections of the chapter fulfil Dissertation Objective 3(a).

In Chapter 4, some essential topics regarding WDSDO were discussed and the stage was set with respect to the consideration of WDS design objectives other than cost minimisation. The main goal in this chapter was to review a number of topics required in the development of a realistic multi-objective WDSDO model, which may be used to design systems which have benefits in excess of the minimum required. The chapter opened with a discussion of the numerous sources of uncertainty in WDSs (§4.1), including water demands, pipe roughnesses, boundary conditions such as reservoir levels, and revealing the necessity for designing with these factors in mind. The difficult problem of demand estimation was the topic of §4.2, which opened with a discussion of the standard water balance, as documented by the International Water Association [131], whereby all the inputs and outputs for a WDS must be accounted for. In §4.2.1 the assignment of baseline demands was presented, whereby mean demands are assigned to WDS nodes for different land-use types (either based on water usage records for an existing system, or using government water-use guidelines, such as the South African CSIR Red Book [44], and municipal guidelines (*e.g.* the Johannesburg Water and Tshwane Metropolitan Municipality guidelines [103]). The essential topic of temporal demand variation was considered in §4.2.2, which deals with the application of typical daily or weekly patterns, or seasonal demand variation, for different land-use types. This allows one to estimate peak demands and design time-dependent WDS components or operational policies (such as tank location and sizing, and pump scheduling). Fire-flow analysis was discussed next in §4.2.3. Here various international and South African standards for incorporating fire-flows into WDS design were presented, and an automated procedure for fire-flow analysis was proposed. The consideration of emergency demands was described briefly in §4.2.4. This was followed by a description of how to accommodate handling of demand uncertainty (§4.2.5) and correlated demands (§4.2.6). This section details the techniques that have been used in the literature, such as the independent stochastic sampling of nodal demands from a normal distribution (Kapelan *et al.* [141]), or using an integration method with safety factors (Babayan *et al.* [16]). These methods are flawed in that they ignore the temporal correlation of demands, which may be

remedied by inducing a rank correlation on the demand samples, using for example the method of Iman and Conover [128]. Finally, the demand estimation section was concluded in §4.2.7 with a discussion of the difficulty of projecting future demands. It was suggested that several population growth scenarios be considered, or that models incorporating estimated future land use and population growth be used.

The topic of WDS reliability estimation was examined in some detail in §4.3, focusing primarily on probabilistic reliability (which is the probability that hydraulic requirements are met under a range of demand conditions and failure events) and reliability surrogate measures (which are designed to have a positive correlation with probabilistic reliability). Probabilistic models were presented in §4.3.1. The analytical probabilistic models investigated included the MVFOSM and FORM models used by Xu and Goulter [267, 268] and the integration-based method of Babayan *et al.* [16]. The Monte-Carlo based model developed by Kapelan *et al.* [141] was also discussed, which uses sampling reduction techniques such as LHS. The conclusion was made that a reduced sampling methodology combined with an evolutionary algorithm (with distributed sampling across generations) may the be best way of approaching the incorporation of probabilistic reliability. The issue of demand satisfaction was highlighted as a critical issue, since we may still be interested in the degree of failure of an infeasible system. ADS measures were introduced, namely ADSU for uncertain demands and ADSF for pipe failure analysis. The calculation of these values requires pressure driven analysis, or the unique use of a DDA model. Reliability surrogate measures were dealt with in §4.3.2. The RSMs included the Resilience Index of Todini [227], Network Resilience by Prasad and Park [194] (both of which focus on excess hydraulic power, the latter including a uniformity factor to promote reliable loops), Flow Entropy by Tanyimboh and Templeman [225] (which maximizes pipe flow uniformity), and a novel mixed surrogate measure that combines the Resilience Index and Flow Entropy. Graph theoretic methods of reliability quantification were also mentioned briefly in §4.3.3, but were dismissed for general use on the grounds that they involve solving NP-hard combinatorial optimisation problems.

Pipe layout design was discussed in §4.4, whereby redundant layouts are provided along with the capability to eliminate pipes during the course of optimization. Several additional topics were considered in §4.5, including the inclusion of running and maintenance costs in the objective function, tank and pump design, designing for maximal water quality and leakage abatement, uncertainty in pipe characteristics, valve design and transient analysis. Section 4.5 constitutes material for possible future work. The material in Chapter 4 was included in final fulfilment of Dissertation Objective 2, and in partial fulfilment of Dissertation Objectives 3(b) and 5.

Chapter 5 is a self-contained introduction to the topic of multi-objective optimisation, with a focus on MOEAs. The chapter opened with a discussion of key MOO topics in §5.1, defining Pareto-optimality and Pareto-approximation sets, and highlighting classical techniques towards MOO, and current generation techniques which employ Pareto-based fitness assignment schemes. A concise history of the multi-objective WDSDO was provided in §5.2, detailing notable developments in the field over the last decade. Highlights of this included: the development of the structured messy genetic algorithm for WDS design by Halhal *et al.* [116], the investigation of various MOEAs (such as NSGA-II [61] and SPEA-II [277]) towards WDSDO by numerous researchers [86, 87, 88, 182, 194, 266], the adaptation of existing MOEAs to cater for the specific needs of WDSDO such as the RNSGA-II algorithm by Kapelan *et al.* [141], and the very successful application of other modern algorithms such as EDAs [185] and hybrid algorithms which combine evolutionary optimization techniques with machine learning [68] (*i.e.* ParEGO and LEMMO).

A generic multi-objective formulation of the WDSDO problem was presented in §5.3, with the

primary goals of cost minimisation, reliability maximisation, and penalty function minimisation (where this is a function of constraint violations, including nodal pressure head limits and velocity limits); although it incorporates the same constraints as the least-cost formulation in (3.1), scope was provided for additional design variables, objectives and constraints. A number of formative concepts in population-based metaheuristics were reviewed in §5.4, illustrated using the algorithms NSGA-II and SPEA-II. These concepts included: Pareto-based fitness assignment strategies such as Pareto-rank and raw fitness (§5.4.1), diversity preservation mechanisms such as crowding distance and k-th nearest neighbour (§5.4.2), parental and environmental selection schemes, employing elitism and other forms of population management (§5.4.3), constraint handling by means of the penalty term method and the constrained domination method (§5.4.4), a wide range of variational operators and solution encoding techniques (§5.4.5); population sizing methodologies (§5.4.6), epsilon-domination and grid-based optimisation schemes (§5.4.7), and adaptive population sizing techniques (§5.4.8). Performance evaluation for MOAs was discussed next in §5.5, advising on how one might compare Pareto-approximation sets produced by different algorithms in multi-objective space. The topic of convergence to a stable population using change in hypervolume was suggested in §5.5.1. Algorithmic parameter tuning was discussed in §5.5.2. A general strategy for algorithmic comparison in the multi-objective metaheuristic domain was developed in §5.5.3, using the idea of convergence trials, followed by full time trials to assess the two performance criteria of speed and solution quality. Finally, solution quality assessment was detailed in §5.5.4, where the Pareto-compliant performance metrics of dominance rank and hypervolume were introduced, as well as the novel spread metric of ϵ-archive size.

Suitable classes of population-based metaheuristics for WDSDO were reviewed in §5.6–§5.8, and all of the algorithms analysed in this dissertation were described in detail. In particular, three MOEAs were described and their algorithms were presented in pseudocode form, namely NSGA-II (§5.6.1), SPEA-II (§5.6.2), and DE (§5.6.3). Alternative population-based metaheuristics were also discussed and presented in pseudocode form, including a novel greedy algorithm (GREEDY) in §5.7.1 (GREEDY was developed by combining several heuristic design techniques from the literature in order to mimic the design strategy of a human engineer, which enables the algorithm to function as an effective local search), a Multi-objective Particle Swarm Optimisation algorithm in §5.7.2, an EDA called the Univariate Marginal Distribution (UMD) algorithm and a novel variant called the Partitioned UMD in §5.7.3, the Dynamic Multi-objective Evolutionary algorithm and a new version adapted for WDSDO called ADMOEA in §5.7.4, and a novel self-adaptive evolutionary algorithm called ANIMA in §5.7.5.

Finally, the evolutionary hyperheuristic AMALGAM, developed in 2007 by Vrugt and Robinson [244], was presented in §5.8. AMALGAM allows for the simultaneous incorporation of multiple metaheuristics within a generic evolutionary framework, in the hope of improving performance and efficiency by adopting the philosophy of strength in diversity. Remarkably, the effectiveness of this strategy was shown by Vrugt and Robinson on a benchmark suite of continuous multi-objective optimisation problems. Owing to shortcomings in the original formulation, several variants of AMALGAM were developed for this dissertation, namely AMALGAMS, which uses the SPEA-II framework, TAMALGAM, which incorporates sub-algorithm running times into the offspring partitioning formula, AMALGAMI and AMALGAMJ, which both use different metrics to quantify algorithm success in the offspring partitioning formula. Chapter 5 was presented to address Dissertation Objective 4 and partially fulfil Dissertation Objectives 3(b), 5 and 7.

At the heart of this dissertation are Chapters 6–9. In these chapters twenty-three metaheuristics are compared with respect to WDSDO, reliability analysis is conducted in order to compare the suitability of four reliability surrogate measures, and the most successful algorithm is applied

in a real South African case study.

Chapter 6 concerned the implementation of multi-objective WDSDO for this dissertation, presenting the overall testing strategy, followed by the WDS benchmark descriptions and WDSDO problem formulations for each, and finally providing technical details with respect to global MOO concerns and specific algorithmic implementation details. A four-phase testing strategy was detailed in §6.1. The first phase of testing entailed the comparative testing of algorithmic performance towards WDSDO for the twenty-three metaheuristics on nine WDS benchmarks from the literature. This involves thirty convergence time trials for each algorithm-benchmark combination, followed by full trials with thirty optimisation runs using the 90^{th} percentile of the average convergence time limits.

The second phase is a comparison of two constraint handling methods for the first eight benchmarks, excluding EXNET. The metaheuristics compared in Phases 1 and 2 included ADMOEA, AMALGAM$_{\text{ndp}}$, AMALGAM$_{\text{ndu}}$, AMALGAM$_{\text{ndug}}$, AMALGAMI$_{\text{ndp}}$, AMALGAMI$_{\text{ndu}}$, AMALGAMI$_{\text{ndug}}$, AMALGAMS$_{\text{ndp}}$, AMALGAMS$_{\text{ndu}}$, ANIMA, DE, GREEDY, NSGA-II, PSO, PUMDA, SPEA-II, TAMALGAM$_{\text{ndp}}$, TAMALGAM$_{\text{ndu}}$, TAMALGAM$_{\text{ndug}}$, TAMALGAMJ$_{\text{ndp}}$, TAMALGAMJ$_{\text{ndu}}$, TAMALGAMJ$_{\text{ndug}}$, and UMDA.

Phase 3 of testing involved the comparison of four RSMs with each other and with respect to probabilistic and failure reliability in terms of their ability to yield designs that are robust against water demand variation and pipe failure. As far as the author is aware, this was also the first systematic study involving a number of WDS benchmarks in which regression analysis is used to compare surrogate reliability measures with estimates of *probabilistic reliability* derived via simulation. Phase 4 of testing is the South African developmental case study, namely the R21 Corridor WDS, where a PEM engineered design requiring human intervention was compared to design using the AMALGAM hyperheuristic. The technical specifications of the nine WDS benchmarks from the literature followed in §6.2, also including a discussion of the WDSDO model adaptations required to implement each successfully. The WDS benchmarks described were the Two Reservoir Problem (TRP) in §6.2.1, the Two Loop Network (TLN) in §6.2.2, the New York Tunnel System (NYTUN) in §6.2.3, the Hanoi Network (HANOI) in §6.2.4, the Blacksburg Network (BLACK) in §6.2.5, the Fossolo Network (FOSS) in §6.2.6, the Pescara Network (PESC) in §6.2.7, the Modena Network (MOD) in §6.2.8, and the Exeter System (EXNET) in §6.2.9. The global optimisation parameter values used were discussed in §6.3.1, as well as the individual parameter values used for the individual metaheuristics and benchmarks in §6.3.2–§6.3.5. Finally, some important programming considerations were mentioned in §6.3.6. The contents of this chapter completed fulfilment of Dissertation Objective 5, and set the ground work for satisfying Dissertation Objectives 6–10 by detailing the experimental setup and forms of analysis, the WDS benchmarks, and the various optimisation parameters used.

In Chapter 7 the results from the first two phases of testing were presented. The algorithmic performance analysis of Phase 1 was divided into different sections for the eight WDS benchmarks TRP, TLN, NYTUN, HANOI, BLACK, FOSS, PESCA, and MOD (7.1.1–7.1.8). Thirty hypervolume convergence time trials (using convergence criteria of less than 0.05% change in hypervolume for 200 consecutive generations) were conducted for each algorithm-benchmark pair in order to compare algorithmic efficiency. Thirty fair time trials were then used to conduct the optimisation (with time limits computed using the 90^{th} percentile of average convergence times) in order to compare the solution quality produced by the various algorithms. The results for the first eight benchmarks were summarised in §7.1.9. The following general observations were made regarding the convergence trials:

- The fastest algorithm in terms of convergence was ADMOEA, also achieving acceptable

hypervolume performance. However, it fared relatively poorly in terms of dominance rank and standard deviations of all metrics, indicating that it is one of the less stable algorithms.

- Other non-dominated algorithms in terms of convergence time and average hypervolume were AMALGAM$_{\text{ndp}}$, AMALGAMI$_{\text{ndp}}$, and AMALGAM$_{\text{ndu}}$, with the latter producing the highest average normalized hypervolume in a convergence time 0.8218 of the global average.

- The algorithm with the best dominance performance was TAMALGAM$_{\text{ndu}}$ with lowest dominance rank and standard deviation for dominance rank, along with fairly decent hypervolume performance in an average normalised time 0.8754 of the global average.

- Given its excellent dominance and stable performance, TAMALGAM$_{\text{ndu}}$ was suggested as the preferred algorithm for time-critical WDSDO amongst those tested.

- The slowest running algorithm was SPEA-II, with a normalised convergence time 1.7043 of the global average. However, it exhibited one of the most stable performances.

The following general observations were made regarding the full-length time trials, excluding EXNET:

- The top four performing algorithms in the full-length time trials were NSGA-II, TAMALGAM$_{\text{ndu}}$, TAMALGAMJ$_{\text{ndu}}$ and AMALGAMS$_{\text{ndp}}$, and were mutually non-dominated with respect to each other for the various performance metrics.

- NSGA-II was the top performing algorithm in terms of average dominance rank with a value of 1.1.

- TAMALGAM$_{\text{ndu}}$ achieved the best average position rank value of 5.125 and the lowest standard deviation for NHV of 0.26.

- The best algorithm in terms of average NHV was AMALGAMS$_{\text{ndp}}$ with a value of 0.9512.

- The GREEDY algorithm exhibited the worst performance overall, with an average dominance rank of 410.64. This demonstrates that despite its functioning as a powerful local search, a GREEDY algorithm which mimics engineering judgement cannot compete with modern metaheuristics, which employ advanced (intelligent) strategies in order to uncover better solutions with less computational effort.

- The PUMDA outperformed the UMDA on average, although neither fared well compared to the MOEAs. This is probably due to the fact that they lack innovation, only exploiting the solution building blocks in the original population. However, when hybridized with MOEAs, overall performance was improved.

The four best algorithms from the full time trials were executed for the very large EXNET benchmark for thirty 5.4-hour runs. AMALGAMS$_{\text{ndp}}$ proved the best algorithm overall, finding a much broader section of the Pareto-front than the other algorithms, although it was outperformed in the high Network Resilience region by TAMALGAM$_{\text{ndu}}$, and failed to locate Pareto-optimal solutions of very high Network Resilience. In conclusion, AMALGAMS$_{\text{ndp}}$ would seem to be the best algorithm at providing a broad range of solutions for difficult WDSDO problems, and TAMALGAM$_{\text{ndu}}$ appears to be one of the best choices for efficient, consistent performance,

and achieving solutions of high reliability. Sub-algorithm analysis was conducted in §7.1.11 for two variants of the basic AMALGAM, namely AMALGAM$_{\text{ndu}}$ and AMALGAM$_{\text{ndug}}$, in order to demonstrate a fundamental shortcoming of the basic formulation. The variant including GREEDY as a sub-algorithm exhibited a premature convergence phenomenon, caused by excess rewarding of GREEDY performance, which over-emphasizes finding local optima. The basic AMALGAM formulation is unable to discriminate between mild improvements, as delivered by GREEDY, and more significant ones generated by other metaheuristics. The results from the Phase 2 constraint technique comparison in §7.2 indicated that both the penalty term method and the constrained domination technique are viable options for general usage. Both alternatives are able to locate very similar Pareto-fronts along the region of highest benefit-cost, and the constrained domination technique may be the method of choice due to its generality and lack of requirement of user-parameters. The results of Chapter 7 were presented in fulfilment of Dissertation Objectives 6, 7 and 8. Finally, in order to conduct this analysis, the implementation of this optimisation model in a software library was completed, in partial fulfilment of Dissertation Objective 10.

In Phase 3 of the analysis, which appears as Chapter 8, the four WDS reliability surrogate measures Resilience Index, Network Resilience, Flow Entropy, and the novel Mixed Surrogate measure, were compared in terms of their ability to yield designs which are robust with respect to probabilistic reliability (obtained via simulation using uncertain nodal demands, as described in §8.1) and failure reliability (measured by simulating all single-pipe failures, as described in §8.2). An analysis was conducted for the eight WDS benchmarks TRP, TLN, HANOI, NYTUN, BLACK, FOSS, PESC and MOD. An iterative demand adaptation method was used in conjunction with DDA (employing EPANET) to calculate demand deficits (method outlined in §8.3). Thirty optimisation runs were used to produce attainment approximation sets for each RSM, and stochastic demand and failure reliability were quantified via simulation for these designs, using the demand satisfaction measures, ADS under uncertainty (ADSU) and ADS under pipe failure (ADSF). Regression analysis was also used to investigate the relationship between the RSMs and their ADS reliability equivalents. While they were often strongly correlated ($R^2 > 0.8$) this only occurred consistently enough for the Network Resilience RSM, since its average R^2 values for ADSU and ADSF were both larger than 0.8, at the very least indicating a very strong positive correlation, and potentially warranting the general hypothesis of a linear relationship between the NR and the ADS measures. Part of the investigation was to determine whether there was any physical effect of the different RSMs on the resulting designs. For example, it is desirable to search for WDS solutions with fewer and less severe discontinuities of size between adjacent pipes, so this was tested. Pipe size discontinuity was quantified by summing the pipe diameter differences at all the nodes of a design (sum of diameter differences — SDD), as well as the sum of the squared diameter differences (SQDD). On average across the eight benchmarks, Resilience Index, Network Resilience, Flow Entropy and the Mixed Reliability measures yielded average SDD values which were 93.27%, 73.38%, 79.38% and 96.84% of the maximum obtained for that benchmark, respectively. Therefore, Network Resilience produced the designs with the least extreme discontinuities, providing significantly smoother pipe gradients on average, which is desirable in practise. This accords with the objectives of reliable loops and flow uniformity set out in the derivations of these RSMs. Although it is difficult to make general conclusions from this analysis involving such a small number of benchmarks, the following general observations could be made:

- The Resilience Index seems to dominate the other RSMs in terms of yielding solutions reliable under pure stochastic demand variation, which agrees intuitively with its primary goal of maximizing excess system power, and its freedom from other constraining criteria.

- However, the Resilience Index performed relatively poorly in comparison to both Network Resilience and the Mixed Surrogate when it came to pipe failure analysis, which accords with their goal of obtaining redundancy through reliable loops.
- The Flow Entropy measure performed the worst overall. It is advised that Flow Entropy be used for WDS design only in combination with other RSMs.
- Although no clear 'best measure' was identified, Network Resilience demonstrated excellent performance with respect to both ADSU and ADSF, and in most cases dominated the other RSMs in ADSF-Cost space.
- The solutions generated using Network Resilience have the desirable property of being less costly on average and have superior pipe adjacency characteristics, whereby large size discontinuities are avoided. This suggests that Network Resilience may be the most practical RSM of the four tested for use in general WDS design.

Although the use of RSMs may be able to save an order of magnitude in terms of processing speed compared to traditional stochastic probabilistic reliability quantification, which requires Monte Carlo simulation or analytical gymnastics, it is not yet clear whether they will provide sufficient quality of results compared to actually incorporating stochastic demands and failure analysis into the design optimisation process. The contents of Chapter 8 was in fulfilment of Dissertation Objective 9. In order to conduct this analysis, the implementation of this optimisation model in a software library was completed in final fulfilment of Dissertation Objective 10.

In Chapter 9, a the design of real South African case study (the R21 Corridor WDS) was optimised using AMALGAMS$_{\text{ndp}}$ and Network Resilience, and it was found that it is able to improve rapidly and significantly upon a preliminary engineered design (a significant saving of R11 877 193 (16.47% of the project cost) was achieved by Alternative Design 1). AMALGAMS$_{\text{ndp}}$ was rapidly able to locate feasible solutions, and the technique of design with redundant layouts proved fruitful in providing alternative layouts for the system. The duration of optimisation is sufficiently short for such a medium-sized WDS, showing that AMALGAMS$_{\text{ndp}}$ may easily be employed in practice. In conclusion, AMALGAMS$_{\text{ndp}}$ has been demonstrated to be a powerful optimisation paradigm in the solution of a real-world engineering design problem within the domain of discrete optimisation problems.

10.2 Contributions of this Dissertation

An attempt is made in this section to clarify the contributions made in this dissertation towards the field of WDSDO.

10.2.1 Major Contributions

Major contributions are listed in this section, judged in terms of their potential positive impact on the WDSDO community.

Contribution 1 *The fair MOO algorithm comparison methodology of §5.5.3.*

The fair MOO algorithm comparison methodology of §5.5.3 was developed for this dissertation as a suggested improvement on the traditional algorithmic comparative strategies for MOO.

One is interested in comparing algorithmic performances in a fair manner, but there are two major components of performance, namely speed and solution quality. The philosophy of fair time trials was adopted, whereby algorithms should be given an equivalent time period in which to find the best results possible. However, this time period should not be assigned arbitrarily (as is often done in the literature), and this method also fails to address the relative efficiencies of the algorithms being compared. One solution to both of these problems is the hypervolume convergence trial technique of §5.5.1, which defines convergence as the event that the percentage improvement in the approximation set falls below a specified threshold for a required number of consecutive generations (0.05% change in hypervolume for 200 consecutive generations was used in this dissertation). Numerous convergence trials were conducted for each algorithm-benchmark pair in order to derive average convergence times for each (which could then be used to compare relative algorithmic efficiencies). The 90^{th} percentile of average convergence times was then used as the time limit in full time-trials with equivalent time allotments, such that the Pareto-approximation solution quality could be compared.

Contribution 2 *The GREEDY algorithm in §5.7.1.*

A greedy WDS design algorithm was developed by the author for specific use in this dissertation, and it was called the *WDS Greedy algorithm* (abbreviated as GREEDY). It was adapted from four prior WDS design heuristics, namely those of Keedwell and Khu [142], Afshar *et al.* [4], Todini [227], and Saldarriga *et al.* [205]. It employs these heuristics as well as several practical adjustment steps to improve solution performance based on engineering judgement, similar to those strategies that might be used by an engineer during the course of a manual design procedure. It is greedy in the sense that it conducts a neighborhood search in which the best improvement step is followed for each of the different heuristic rules. Owing to its greedy nature, there is a danger that the search may become trapped at local optima, which was demonstrated to be the case in the comparative analysis, with GREEDY coming in last position of the twenty-three algorithms analyzed across eight WDS benchmarks. Thus, it was demonstrated that an algorithm which mimics engineering judgement cannot compete with state-of-the-art metaheuristics.

Contribution 3 *The first application of the AMALGAM hyperheuristic towards WDS design, and the development and testing of improved variants of AMALGAM.*

The AMALGAM evolutionary hyperheuristic was described in §5.8. This generic evolutionary meta-algorithmic framework, which incorporates a number of sub-algorithms in the solution of a MOO problem, was developed by Vrugt and Robinson [244]. AMALGAM attempts to produce a faster, more reliable search by dividing the creation of offspring amongst the sub-algorithms in proportion to the success of these sub-algorithms during previous generations. The philosophy behind this approach is that the strengths of different metaheuristics may be exploited dynamically. This dissertation constitutes the first successful application of AMALGAM to the multi-objective WDSDO problem. AMALGAM and its variants fared very well in the comparative algorithmic analysis of Chapter 7.

Various shortcomings were identified in the original AMALGAM formulation, particularly in situations where a sub-algorithm locates many successful offspring of only marginally improved fitness (as was demonstrated for $\text{AMALGAM}_{\text{ndug}}$ in §7.1.11). Alternative formulations for AMALGAM were devised for this dissertation. The first variant was AMALGAMS, which uses the SPEA-II environmental selection strategy instead of that of NSGA-II. This variant demon-

strated excellent performance, particularly in the implementation $\text{AMALGAMS}_{\text{ndp}}$, which was identified as the best algorithm for obtaining a broad selection of solutions for very large / difficult WDSDO problems. The TAMALGAM variant was developed in order to reward sub-algorithm efficiency, by including the sub-algorithm running time in the offspring partitioning formula. This variant proved extremely successful, particularly in the $\text{TAMALGAM}_{\text{ndu}}$ implementation, which achieved excellent efficiency and dominance. The AMALGAMI and AMALGAMJ variants were developed in an attempt to find more intelligent offspring partitioning scheme using solution dominance rankings, and discounting the importance of the number of solutions generated, respectively. The $\text{TAMALGAMJ}_{\text{ndu}}$ variant appeared as one of the top four performing algorithms in the comparative analysis. The intended effect of preventing sub-algorithms from generating many low-quality successes was demonstrated clearly, since all variants outperformed the original $\text{AMALGAM}_{\text{ndug}}$ implementation during the full-length time trials as reported on in Chapter 7.

Contribution 4 *The largest comparative study of metaheuristics for multi-objective WDSDO in Chapter 7.*

As far as the author is aware, the content of Chapter 7 in this dissertation constitutes the largest single study of its kind, comparing the largest number of multi-objective metaheuristics towards the bi-objective design optimization of the largest number of WDS benchmarks. The twenty-three alternative algorithms — including fourteen different implementations of AMALGAM, four novel algorithms developed for this study (ADMOEA, ANIMA, GREEDY and PUMDA), and five metaheuristics from the literature (NSGA-II, SPEA-II, PSO, DE and UMDA) — were compared with respect to the design of nine WDS benchmarks (TRP, TLN, NYTUN, HANOI, BLACK, FOSS, PESC and MOD). The best four algorithms in the full-length fair time-trials were identified as NSGA-II, $\text{TAMALGAMJ}_{\text{ndu}}$, $\text{TAMALGAM}_{\text{ndu}}$ and $\text{AMALGAMS}_{\text{ndp}}$, which were mutually non-dominated in terms of average dominance rank, standard deviation of hypervolume, average rank and standard deviation of dominance rank, and average hypervolume, respectively.

These four algorithms were then compared with respect to the very large EXNET benchmark, requiring over a month of computing time. It was found that $\text{AMALGAMS}_{\text{ndp}}$ convincingly outperformed all other algorithms in terms of averages and standard deviations of dominance rank and hypervolume. In terms of attainment, $\text{TAMALGAM}_{\text{ndu}}$ was able to locate better solutions in the high reliability region of the Pareto-front. It was suggested that $\text{AMALGAMS}_{\text{ndp}}$ is the algorithm of choice amongst those tested for large / difficult WDSDO problems.

Contribution 5 *The first systematic comparison of four RSMs using reliability and failure analysis in Chapter 8.*

As far as the author is aware, this dissertation contains the first study comparing more than two RSMs in terms of their ability to generate designs that are robust under uncertain demands and pipe failure reliability, using regression analysis to determine the correlation between the RSMs and ADS measures of reliability. The four RSMs compared were the Resilience Index, the Network Resilience, Flow Entropy, and the Mixed Surrogate, with respect to eight WDS benchmarks. It was found that the Resilience Index exhibited the highest correlation with respect to ADSU on average, and that Network Resilience achieved the highest correlation with respect to ADSF on average. Flow Entropy was deemed to be an unreliable indicator of reliability, except in conjunction with other RSMs, such as in the Mixed Surrogate, which

produced expensive solutions of the highest ADS, but having the worst pipe size discontinuities. The Resilience Index RSM produced solutions with fewer pipes (less redundancy) and larger pipe size discontinuities. The Network Resilience RSM produced cost-effective solutions with the smallest pipe discontinuities, sufficient pipes for redundancy, and the best overall correlation with ADS measures; and was therefore recommended as the most practical RSM of those considered here for WDSDO.

Contribution 6 *A real-world South African developmental case study comparing a computer-assisted human-engineered design to AMALGAMS generated solutions.*

A real-world developmental case study, called the R21 Corridor WDS, was conducted using $\text{AMALGAMS}_{\text{ndp}}$ for bi-objective design with objectives of cost minimisation and Network Resilience maximisation. $\text{AMALGAMS}_{\text{ndp}}$ was shown to improve substantially upon a preliminary engineered design for the R21 Corridor WDS, both in terms of cost and reliability (a significant saving of 16.47% of the project cost was achieved by Alternative Design 1). Furthermore, $\text{AMALGAMS}_{\text{ndp}}$ was rapidly able to find feasible solutions, and exhibited an acceptable running time, which warrants its use in practice. The technique of design with redundant layouts was also demonstrated to be effective.

10.2.2 Secondary Contributions

Miscellaneous secondary contributions are listed in this section.

1) *An in-depth literature survey of WDSDO.*

Given the immense body of literature regarding WDSDO, a comprehensive survey of the field would fill an entire book on its own. However, an in-depth survey was made of single- and multi-objective WDS design optimisation, summarising what the author believes are the most critical contributions by the research community towards this complex and rapidly developing field. This survey spans Chapters 3, 4 and 5, providing a history of WDSDO, covering numerous attempts at single- and multi-objective problem formulation, the relevant associated optimisation algorithms, and numerous advanced topics in WDS design. This literature survey provides a much needed update to such resources as Mays [169] and Walski *et al.* [251].

2) *The suggestion of an automated fireflow analysis procedure combining deterministic and stochastic analysis in §4.2.3.*

A procedure was suggested for incorporating fireflow analysis at some of the most critical sections of a WDS, also including a stochastic component to uncover unexpected weaknesses in the system. Such a fireflow analysis scheme stands in contrast to industry practice, where typically only a few critical scenarios are considered, or the system is massively over-designed at great expense to account for all possible fire scenarios (an especially notable phenomenon in the USA). This procedure has yet to be tested.

3) *The Mixed RSM described in §4.3.2.*

A Mixed RSM combining the Resilience Index of Todini [227] with the Flow Entropy measure of Tanyimboh and Templeman [225] was suggested as potentially being a more practical repre-

sentation of real-world reliability requirements. This measure has the advantage of addressing uniformity of flow and pathway redundancy by means of Flow Entropy, and excess nodal power by means of the Resilience Index. While the Mixed Surrogate yielded solutions of higher ADS, these solutions were substantially more expensive than those produced using the other RSMs, and the correlation between the Mixed Surrogate and ADS metrics was also lower.

4) *A new tank design optimization model in §4.5.2.*

A detailed model for incorporating tank design into WDSDO using the penalty term method for constraint handling was suggested, based on similar models by Vamvakeridou-Lyroudia *et al.* [238] and Prasad and Tanyimboh [195]. This model incorporates the goals of cyclic recharge, sufficient emergency storage, and stagnation / overflow avoidance. The model also allows for easy integration into the existing MOO WDSDO framework described in §5.3.

5) *A generic mathematical formulation of the multi-objective WDSDO problem in §5.3.*

A very general formulation of the multi-objective WDSDO problem was provided in (5.1), catering for both discrete and continuous design variables corresponding to the heterogenous components of a WDS, the objective functions for cost minimisation, reliability maximisation and penalty function minimisation (which caters for soft constraints on hydraulic performance), as well as numerous alternative objectives and constraints. This formulation is perfectly geared towards MOEAs, and combines similar formulations from several different sources [16, 87, 141].

6) *The formulation of a normalised penalty function for the accommodation of hydraulic performance goals in §5.4.4.*

A normalised penalty function (5.2) was developed to penalise hydraulic performance constraint violations (maximum pipe velocity violations, as well as maximum and minimum head violations at the nodes), normalised by the size of the feasible range of each constraint. This formulation makes specifications of weighting coefficients easier by removing order disparities between different constraints and it also enables more meaningful aggregation of minimum and maximum constraint violations. Such a penalty term may be used directly as an indication of feasibility, since it is zero when all velocity and head values are in their feasible ranges.

7) *A novel mutation operator for evolutionary optimisation algorithms based on the triangular distribution in §5.4.5.*

A novel mutation operator based on the triangular distribution was developed for real and integer coded genes in order to address the need for general purpose variational operators within the WDSDO context. Triangular mutation was found to be superior to polynomial mutation for pipe size allocation, and was consequently used throughout this dissertation.

8) *The metaheuristic population sizing methodology of §5.4.6.*

A population sizing methodology was developed primarily to ensure that populations are large enough to contain sufficient solution building blocks (schemata). This method uses the technique of Harik and Lobo [119] in the parameterless GA, whereby a race of populations is conducted using populations whose sizes are doubled, until such time as the larger population fails to

outperform the smaller one over an equivalent time period. This smaller size is then taken as the preferred size. Because this size varied from trial to trial, stochastic sizing was repeated ten times for each algorithm-benchmark pair in order to find the most prominent size, which was then indicated as the 'optimal' size for that algorithm-benchmark pair. Once this had been completed for all algorithms, the largest 'optimal' size within every algorithm group was applied to the entire group. Having MOO algorithms produce similarly sized Pareto-approximation sets has many advantages (including leveling the playing field in terms of algorithmic complexity and allowing for fair comparison of solution set quality metrics). Relative algorithmic efficiencies may still be compared by means of conducting *hypervolume convergence trials*.

9) *The ϵ-archive size quality metric in §5.5.4.*

The ϵ-archive size of final approximation sets was suggested as a non-Pareto-compliant measure of solution spread / diversity. Each output approximation set is inserted into an ϵ-archive of identical precision. This archive size is equivalent to the number of evenly spaced ϵ-dominance hypergrid cells containing solutions. If two algorithms consistently converge to a common front (*e.g.* they both have similar dominance ranks), but one algorithm has a larger ϵ-archive, then its solutions cover more of the Pareto-front and are consequently better distributed. The ϵ-archive size provides a dimensionless comparison of relative spread, but should only be considered in conjunction with Pareto-compliant performance measures (*e.g.* dominance rank or hypervolume).

10) *The Partitioned UMD algorithm in §5.7.3.*

A variant of the UMDA, called the *Partitioned UMDA* (and abbreviated as PUMDA), was developed by the author in §5.7.3, whereby the objective space is re-partitioned along the reliability axis during each generation. The size of each partition is generated independently using half of the absolute value of a normal distribution sample with a mean of zero and a standard deviation equal to one third of the reliability range ($r_{\min}$, $r_{\max}$), sampled iteratively until the full range is partitioned. For each sub-range of the reliability range, all the solutions that fall within that partition are then employed to generate a UMD probability model for that partition. In order to generate new solutions, a partition is selected at random and its UMD model sampled to produce offspring. This allows the algorithm to focus on different parts of the solution space, corresponding to different design paradigms, but by redefining the partitions, also allows for the mixing of these paradigms. PUMDA was found to outperform the UMDA overall. PUMDA also formed one of the sub-algorithms in the most successful AMALGAM implementation, namely AMALGAMS$_{\text{ndp}}$.

11) *The ADMOEA algorithm in §5.7.4.*

An algorithm was developed for this dissertation based on the original DMOEA [271], using an identical cellular grid model for the cellular rank and density information, but differing in terms of the population growth and decline strategies. This algorithm, called *Another DMOEA* (and abbreviated as ADMOEA) incorporates a number of advanced features, including a growth strategy whereby a number of new solutions are generated as a function of the grid-size and the current Pareto-set size, three different search mechanisms and a probability vector to control mechanism selection (updated during each generation depending on each mechanism's success rate), a solution age that is incremented during each generation (under the condition that solutions may not be removed from the population before they reach a certain age, which

allows them sufficient time to propagate their genes), a population decline strategy that selects solutions for removal on the basis of their age, cellular rank and density, a regeneration strategy that recreates the entire population using PUMDA once limited improvement has occurred for fifty generations, an external epsilon archive to hold Pareto-optimal solutions (updated before each regeneration), and compression and growth mechanisms to alter the dimensions of the hypergrid in order to zoom in on the important region of the objective space, or to accommodate new solutions outside of the current grid dimensions. Although the ADMOEA proved to be the fastest algorithm in terms of convergence, it was significantly less stable than the other MOEAs (exhibiting the worst average standard deviation of hypervolume) and was outperformed by ten of the twenty-three metaheuristics during the full time-trials.

l2) *The self-adaptive ANIMA MOEA in §5.7.5.*

The novel algorithm ANIMA, a self-adaptive MOEA based on the NSGA-II framework, was developed for this dissertation. It employs two different search mechanisms, namely the SBX crossover operator with triangular mutation, and a differential evolution operator. ANIMA encodes the variation operator parameters along with the solution genes, effectively making each solution an agent carrying both the search instructions and the solution information. These parameter values are generated randomly initially, but are passed on to newly created offspring by their parent solutions. ANIMA proved more successful on average than ADMOEA, but was still outperformed by eight of the twenty-three metaheuristics during the full time-trials, showing less stable performance than several non-adaptive methods.

13) *A comparison of two constraint handling techniques for WDSDO in §7.2.*

Two constraint handling techniques were compared towards WDSDO, namely the penalty function method and the constrained domination method. Neither method consistently outperformed the other, with the constrained domination method performing poorly for the HANOI benchmark, and outperforming the penalty function method for the FOSS benchmark. It is suggested that the penalty function method may be more flexible, since it is able to allow for the mixing of feasible and infeasible solutions for greater variation, while the constrained domination method is more generally applicable since it has the advantage of not requiring user-specified parameters.

14) *The development of a demand adaptation method towards PDA using a DDA model in §8.1.*

In order to address the need for PDA, a demand adaptation method was developed using a traditional DDA model, whereby nodal demands are lowered or increased in accordance with an iterative procedure, depending on whether the pressure at the particular node is below the minimum required or not. This method was used to quantify the demand deficit for a network under extreme demands or pipe failure conditions, in order to calculate the ADS during reliability analysis.

15) *The definition of several novel metrics to describe WDS composition.*

In addition to the ADS metrics, and basic metadata such as average cost and number of pipes, several metrics were developed to describe WDS composition, in order to compare solutions

produced by the RSMs. The first such metric was the *summed diameter differences* (SDD), whereby all pipes adjacent to each node are checked and their diameter differences added to the total. A similar metric was defined as the *sum of the squared diameter differences* (SQDD), whereby the squared differences of adjacent pipes are added, in order to further penalize large discontinuities. These metrics provide an indication of the smoothness of physical pipe size gradients, since fewer discontinuities would indicate a more rational, balanced system, consisting of more redundant pathways of high reliability. The third metric was the *source share deviation from the mean* (SSDM), which considers the proportion of water supply provided by each reservoir, calculates the mean share, and sums the share deviations from the mean, in order to quantify how balanced the supply from the sources is, so as to determine whether or not a system is overly dependant on a single source.

10.3 An Appraisal of the Contributions of this Dissertation

This dissertation constitutes an investigation into the effectiveness of numerous metaheuristics and design strategies with respect to water distribution systems design optimisation. A large number of metaheuristics and hyperheuristics, including both novel algorithms and variations on well-known existing algorithms, were considered in this dissertation, in order to further the knowledge in the field of WDSDO. The common design paradigm of bi-objective WDSDO (focussing on pipe sizing and network layout), with the objectives of cost minimisation and reliability maximisation, was adopted to maximise the general applicability of this study. As far as this author is aware, a WDSDO study of this magnitude and carefully considered experimental strategy has not been conducted before. This dissertation may serve the purpose of providing a novel WDSDO framework for the rational investigation of algorithmic efficiency and solution quality that future algorithmic comparison studies might consider adopting or improving upon.

The algorithms to be compared were selected on the basis of representing several different metaheuristic design paradigms (including traditional MOEAs, auto-adapting MOEAs, local-search heuristics, EDAs, heuristics derived from nature, and hyperheuristics). The top ten performing algorithms applied in the full time-trials with respect to the first eight benchmarks included NSGA-II, seven variants of AMALGAM [244] (an evolutionary hyperheuristic which employs multiple sub-algorithms in a dynamic fashion, applied for the first time to WDSDO in this study), SPEA-II and ANIMA. Attempts to develop state-of-the-art algorithms which consistently outperform conventional multi-objective evolutionary algorithms (such as NSGA-II and SPEA-II) were, however, unsuccessful. Compelling evidence was nevertheless provided that advanced algorithms may be superior for the design of very large or complex WDSs. In particular, one variant of the AMALGAM hyperheuristic design paradigm employing multiple metaheuristics simultaneously (namely, AMALGAMS$_{\text{ndp}}$) convincingly outperformed the common "industry standard" NSGA-II algorithm on the design of a large WDS benchmark. This impressive performance boost provides the research community with a new design strategy for difficult problems, and contributes to state-of-the-art in design theory for WDSDO. Apart from numerous AMALGAM variants, many of which improved upon the original AMALGAM formulation, several novel algorithms were developed and investigated in this study, including ADMOEA, ANIMA, GREEDY and PUMDA. This research provides fresh insights into some alternative design paradigms, and reveals the difficulty of developing stable adaptive algorithms for WDSDO.

Furthermore, several metaheuristics have been excluded as serious contenders for use in general WDSDO, including naive multi-objective particle swarm optimization, the UMDA and

PUMDA algorithms (at least under the proposed population assignment strategy — alternative automated strategies should be investigated), as well as algorithms based solely on greedy search heuristic techniques. Whereas the EDAs showed much promise as sub-components within a broader optimisation framework, it was demonstrated that greedy search techniques may actually impede performance when used as part of a broader strategy (*e.g.* as a local search component, or within a hyperheuristic). It is recommended that greedy searches for WDSDO only be used in combination with a tabu search method or a machine learning algorithm. This is an important finding for the WDSDO community, since there is considerable interest in the literature in improving algorithmic efficiency by means of local-search methods, but not enough emphasis on the dangers of premature convergence, particularly for larger networks. It is hoped that these investigations may serve as a discouragement to other researchers with respect to following the same routes of investigation, or that it may encourage them to develop algorithms that keep these limitations in mind. A technical note on the results of this comparative analysis was well received by the reviewers, and published as

- Raad DN, Sinske A & Van Vuuren JH, 2010. *Multiobjective optimisation for water distribution system design using a hyperheuristic*, Journal of Water Resources Planning and Management, **136**(5), 592–596.

A novel reliability study was conducted on four WDS reliability surrogate measures with the aim of comparing their effectiveness at designing networks reliable under uncertain demands and pipe failure scenarios. This is the first time that a practical, multifaceted framework for testing the effectiveness of reliability surrogate measures has been developed and implemented. This constitutes an important contribution to the field, where such investigations have hitherto only been applied in a very narrow, limited fashion. Investigating the correlation between RSMs and stochastic estimates of reliability has not been carried out before. It is essential that RSMs be subjected to such rigorous scientific analysis in order to evaluate their real-world practicality. The Network Resilience measure [194] was identified as the best candidate for real-world design, since it demonstrates high correlation with both stochastic and failure reliability, and yields solutions of superior robustness under failure, having lower average cost, and exhibiting the smallest adjacent pipe-size discontinuities. This is a very useful result for researchers and WDS engineers alike, but further investigation may be required into potential dangers of using Network Resilience instead of a more intensive form of analysis, such as Monte Carlo simulation. The results of this analysis were used to produce an article, published as

- Raad DN, Sinske A & Van Vuuren JH, 2010. *Comparison of four reliability surrogate measures for water distribution systems design*, Water Resources Research, **46**, Paper W05524 (no page numbers).

Another important contribution was the comparison of two constraint handling techniques, which revealed that neither consistently outperformed the other, but that the penalty method is more flexible, while the constrained domination method [190] is more general. Since these two techniques are interchangeable, the engineer might easily opt for either depending on preference, or consider trying both during the course of a WDSDO exercise.

An in-depth literature survey was conducted with respect to WDSDO, addressing the need to update some of the older surveys in the literature, particularly with respect to MOO in a WDSDO context. A generic mathematical model was formulated for multi-objective WDS design, and a normalized penalty function for hydraulic performance goals was also developed, which may be useful for researchers.

During the course of work towards this dissertation, several novel techniques and metrics were developed, including a novel mutation operator based on the triangular distribution, the suggestion of the ϵ-archive size quality metric, the development of a population sizing methodology for comparative analysis based on the parameterless GA [119], the design of a fair algorithmic comparison technique using convergence trials and equal time-trials, the definition of the Mixed Surrogate RSM, and the development of novel metrics for describing performance (ADSU and ADSF) and desirable WDS structure (SDD, SQDD,SSDM). These techniques expand the design and analysis toolkit of WDS design engineers. Evaluating results using such metrics will empower the WDS community to make better decisions with regards to their choice of reliability surrogate measures and final design selection. A particularly useful development for the WDS engineering community was the demand-adaptation procedure for conducting PDA with a DDA hydraulic engine. While this is not ideal in terms of efficiency, it provides a pragmatic work-around in the interim for reliability analysis, while PDA is not yet an industry standard.

Some WDS design models were developed but not tested, including an automated fireflow analysis procedure combining deterministic and stochastic analysis, and the development of a new tank design optimization model with a normalised penalty function formulation addressing the need for tank performance constraints. Despite not being tested here, these are deemed practical enhancements to a WDSDO strategy.

Finally, a real-world South African WDSDO case study, called the R21 Corridor, was conducted. In this study the results from an AMALGAMS design procedure were compared to a computer-assisted human-engineered design. Significant cost savings (R11 877 193, equating to 16.47% of the project cost) and reliability improvements were obtained using the AMALGAMS method within a realistic time-frame. This further demonstrates the power and practicality of hyperheuristics in a WDSDO setting and was published as

- Raad DN, Sinske A & Van Vuuren JH, 2009. *Robust multi-objective optimization for water distribution system design using a meta-meta-heuristic.* International Transactions in Operational Research, **16**(5), 595–626,

for which the Operations Research Society of South Africa awarded the prestigious Tom Rozwadowski medal in 2010.

Chapter 11

Future Work

During the process of compiling this dissertation, several opportunities for further research were identified by the author. This chapter contains several brief proposals for further research in WDSDO. These proposals include expanding the WDSDO model, investigating alternative metaheuristics towards WDSDO, the further investigation of RSMs and robust design methodologies, the standardisation of certain WDSDO objectives and performance metrics in order to facilitate a database of WDS case studies and associated MOO results, enhancements to the AMALGAM hyperheuristic and the novel algorithms developed in this dissertation (ANIMA, ADMOEA), and a study of specific measures designed to accelerate WDSDO.

Proposal 1 *Expanding the WDSDO model: Additional objectives, design variables, features and analysis types.*

There are numerous ways in which the WDSDO model may be extended to match the requirements of real-world WDS design more closely. The WDSDO model may be extended to *include additional objectives* such as 1) water quality maximisation, 2) water age minimisation, 3) leakage minimisation, 4) pumping cost minimisation, 5) pipe replacement cost minimisation, and 6) the minimisation of nodal pressure deviations from goal pressures. *New soft constraints may also be incorporated*, such as minimum peak-flow cleaning velocities for pipes and tank performance constraints. The hydraulic simulation model may be adapted to incorporate some form of *critical transient analysis*. The MOO WDSDO formulation may additionally be extended to *incorporate design over time*. All algorithms would have to be re-evaluated in order to determine whether they are capable of successfully handling design in higher-dimensional spaces. The challenge lies in capturing the most essential design functionality without unnecessarily complicating the optimisation. An large-scale industry survey would help with this decision.

The WDS benchmarks analysed in this dissertation did not include *the design of tanks, valves or pumps*, a common requirement in the industry. The software library may be extended to include these design variables relatively easily using the models provided in Chapter 4 and the modelling capabilities of EPANET. Most of the algorithms analysed are capable of handling both discrete and continuous variables, with the exception of GREEDY.

Another enhancement may be the *development of a decision support system* with a graphic user interface to allow for ease of use, or the incorporation of the best optimisation routines into an existing commercial software package. The facility of GIS analysis to determine maximal practical layouts and excavation costs for WDSDO would also be extremely useful.

The *automated fire-flow analysis procedure* of §4.2.3 has yet to be tested, particularly in respect of industry regulations and specifications. The inclusion of economic costs due to estimated fire damage in the objective function may also be considered.

In order to be more representative of reality, *failure analysis* may be conducted on the basis of actual pipe failure probabilities as a function of pipe diameters (derived from empirical data) [112]. The estimated *cost of pipe replacement* over the system lifespan may also be included in the cost objective function.

Although WDSDO analysis carries a heavy computational burden, many more WDS benchmarks should be used in a large-scale study of algorithmic performance, in order to enable more general conclusions. This is particularly true for additional large systems such as EXNET. It would be extremely useful if there were an initiative from the WDS engineering community to make such designs readily available in a common format. On a similar note, the use of pressure-driven analysis yields many benefits over DDA, particularly in terms of quantifying the level of demand satisfaction. There should be a determined drive from the industry to standardise on PDA models.

Proposal 2 *The investigation of alternative metaheuristics for WDSDO.*

Several additional algorithms may be investigated for use in WDSDO. The following represents a sampling of modern metaheuristics across a broad range of design paradigms:

1. State-of-the-art MOEAs, such as GDE 3 [152], the *Multi-objective Fast Messy Genetic Algorithm*[1] (MOFMGA) [55], the *Non-dominated Sorting Cooperative Coevolutionary Genetic Algorithm* (NSCCGA) [130], the *Improved Self-Adaptive Chaotic Genetic Algorithm* (ISACGA) [272], the *Multi-objective Cellular Algorithm* (MOCell) [180], the *Random Directions Multiple Objective Genetic Local Searcher* (RD-MOGLS) algorithm [137], and the *Fast Pareto Genetic Algorithm* (FastPGA) [82],

2. modern EDAs, such as the *Multi-objective Real-coded Bayesian Optimisation Algorithm* (MORBOA) [7], and the *Eigenspace Estimation of Distribution Algorithm* (EEDA) [245],

3. other nature inspired meta-heuristics, such as the *Efficient Multi-objective Particle Swarm Optimisation Algorithm* (EMOPSO) [36], multi-objective ant colony optimisation algorithms [275], the *Memetic Algorithm with Population Management* (MA—PM) [196], the *Immune Forgetting Multiobjective Optimization Algorithm* (IFMOA) [273], and a multi-dimensional version of the *Honey Bee Mating Optimisation Algorithm* [174],

4. hybrid algorithms which combine evolutionary optimisation with machine learning techniques, such as the *Pareto Efficient Global Optimisation Algorithm*[1] (ParEGO) [147], the *Learnable Evolution Model for Multi-objective Optimisation*[1] (LEMMO) [68], and the *Multi-objective Tchebycheff-based Genetic Algorithm* (MOTGA) [10], and finally,

5. miscellaneous advanced metaheuristics such as the *Multiple Trajectory Search* (MTS) [232] the *MOSS-II Tabu/Scatter Search* [19], and parallel MOEAs [26].

The GREEDY algorithm of §5.7.1 may readily form the basis of a multi-objective tabu search. The method of tabu search requires a neighbourhood function in order to move between solutions, which GREEDY provides by means of its various heuristic search steps. Tabu search

[1]These algorithms have already been investigated in limited WDSDO studies and are included here because they show promise and should be used in larger studies.

combats the tendency of local searches to become trapped in local optima by storing a history of recent solutions, thereby preventing the search from revisiting a previous state. Tabu search might prove particularly useful as a refinement optimisation technique after other algorithms have converged.

Proposal 3 *Further investigation of reliability surrogate measures and robust optimisation strategies.*

Alternatives to the RSMs analysed in this dissertation may be investigated. For variants on the known RSMs, one might create a new measure combining Flow Entropy with Network Resilience, or assign different weights to the Flow Entropy factor in the existing Mixed Surrogate measure. Furthermore, one may consider multi-objective optimisation using several RSMs simultaneously, such as having Network Resilience and Flow Entropy as separate objectives to be maximized.

Alternative RSMs might be based on *graph theoretic network descriptors*, such as measures which explicitly take into account the *number of independent pathways* between critical vertices, the size of the smallest edge cut set whose removal separates critical vertices (*local edge connectivity*), the *number of primary loops* in the network, or approximations to the minimum cut sets (*graph edge connectivity*) which would render the network disjoint. Obviously, the higher these descriptor values, the more reliable a network would be in a purely topological sense. Determining the number of independent pathways between two vertices can be solved efficiently using a breadth-first search. Determining local edge connectivity may be determined efficiently using the *max-flow min-cut algorithm*, and the graph edge connectivity as a whole determined by considering the minimum local edge connectivity for all pairs of vertices in the network (which is not necessarily practical to calculate within a reasonable time-frame, excluding this as a useful measure for very large networks) [113]. Such descriptors might be useful for rapid reliability approximation if there is a means of identifying critical nodes in the network. This would reasonably include all the water sources, and might additionally include the node(s) of highest demand, the node(s) at the centroid of the network, and/or the node(s) furthest from the sources. Descriptors such as these may be used in tandem with other RSMs, rather than forming primary reliability indicators.

In 2002, Ostfield *et al.* [188] developed three water distribution reliability measures, namely the *Fraction of Delivered Volume* (FDV), the *Fraction of Delivered Demand* (FDD) and the *Fraction of Delivered Quality* (FDQ) for use in MCS-based reliability studies. The FDV is the sum of the total volumes delivered to a consumer node divided by the sum of the total volumes requested by the consumer over all the simulation runs (*i.e.* it is a macroscopic form of ADS). The FDD is the sum of all time periods in all simulation runs for which the demand supplied at a consumer node is above the necessary demand factor, divided by the total number of simulation runs and multiplied by a demand cycle (*e.g.* 24 hours). Finally, the FDQ is the sum of all time periods in all simulation runs for which the concentration supplied at a consumer node is below a threshold concentration factor divided by the total number of simulation runs multiplied by a demand cycle. Naturally FDV and FDD must be maximised, while FDQ should be minimised. These reliability measures are very general, but require Monte Carlo simulation to calculate them, and are closer to being actual measures of reliability than surrogate measures. However, one may consider adapting them for a reduced set of critical scenarios.

Robust design strategies might include topological feature enforcement, such as the assurance of at least 2-connectedness for a network, the minimisation of the number of pipes that do not

form part of at least one primary loop, or the incorporation of pipe failure analysis during the design process.

Future studies should compare the performance of RSMs to the best stochastic algorithms, such as the RNSGA-II (Kapelan *et al.* [141]), *i.e.* those algorithms and their implementations which explicitly take probabilistic and failure reliability into account. These algorithms will typically require an order of magnitude increased processing time since numerous hydraulic simulations must be conducted. This is applicable even when a sampling reduction technique such as LHS is used.

Proposal 4 *WDSDO metric standardisation and a database of WDS case studies and associated MOO results.*

One of the most significant problems with a MOO WDSDO study of this sort is the lack of readily available data for validation of optimisation model results. The main stumbling block to this problem is the lack of consensus on optimisation objectives in addition to cost minimisation. The author suggests that the WDS research community agree on several standard quantifiers of reliability (such as Network Resilience, ADS or FDV), water quality (chlorine concentrations, water age, FDQ), pressure health measures (percentage deviation from ideal goal pressures), and so forth. Such an agreement will enable future design studies to employ similar objectives, which will allow for the creation of a public domain database of WDSDO results organised per benchmark. This approach may also require a shift away from using DDA as standard, towards the more physically accurate PDA models. Furthermore, there must be some consensus on algorithmic comparison studies, since the current experimental methods employed are often designed arbitrarily, may be statistically unsound, or use scientifically questionable principles (such as comparing evolutionary algorithms for an equivalent number of generations). A guideline should be developed which includes specifications of standard experimental procedures, such as convergence analysis (with associated standard parameter values), and the criteria for scientifically meaningful results. This might even include the use of standard benchmark algorithms, such as NSGA-II and SPEA-II.

This analysis of standardised metrics and experimental procedures for WDSDO may be conducted by an extensive literature review, and by means of polling the leading researchers in the field. Although there are already a few online repositories of WDS benchmarks (for instance [65] and [84]), it would be ideal if far more WDS benchmarks were made available, particularly more large, real-world systems such as EXNET, to improve the generality of results. Such a repository may then double as a database of optimisation results for the relevant WDS benchmarks.

Proposal 5 *Further investigation of the AMALGAM, ADMOEA and ANIMA paradigms.*

The AMALGAM framework might be improved in several ways, particularly in terms of the reward function which partitions offspring amongst the sub-algorithms. Currently, the use of a partitioning formula acts as an instantaneous assessment of sub-algorithm performance, with a limited memory based on the historical allocations. Instead, the use of a machine learning component (such as an artificial neural network or rule induction algorithm) for the assignment of offspring to different sub-algorithms may yield better global success. Statistical models such as the Hierarchical Bayesian Optimisation Algorithm [191] may also potentially be used for the same purpose. Sub-algorithm performance may be tracked over short, medium and long term periods, for varied offspring assignment strategies. A round-robin offspring generation

scheme may also be considered for sub-algorithms — so that each sub-algorithm will not have to update its internal model during each generation, which should save numerical processing time. A dynamic population sizing methodology may also potentially be incorporated into the AMALGAM framework. Furthermore, better formalism may be warranted in terms of AMALGAM sub-algorithm model building.

The ADMOEA and ANIMA paradigms may be developed further, or even merged, with a primary focus on stablising performance and finding more rational evolution strategies. It would be particularly worthwhile to investigate whether one could create ADMOEA variants where the speed benefits can be maintained without a reduction in solution quality.

Some ideas for future variants of AMALGAM, ADMOEA and ANIMA include:

1. A library of basic search operators, with variable search parameters, used to build search mechanisms dynamically in true hyperheuristic form. This could potentially involve statistical models or a rule induction algorithm, such as the C4.5 algorithm of Quinlan [198], both for operator selection and parameter assignment / adaptation.
2. The incorporation of search landscape approximation techniques, such as a Kriging (Gaussian process) model [147], or a Tchebycheff scalarising function [10].
3. Allowing for competitive / cooperative evolution with sub-populations [69].
4. The incorporation of search concepts found in *Simulated Annealing* (to allow moves resulting in worse solutions) and *Tabu search* (to prevent revisiting known solutions).
5. The incorporation of explicit solution building-block filtering and recombination using non-dominated templates [55].
6. Alternative environmental selection mechanisms, such as the use of an ϵ-dominance scheme [125], external archiving schemes, and explicit removal of the best solutions (solution death) to force new search trajectories.
7. A multi-layer evolutionary scheme for search parameter evolution (a meta-MOEA).

Proposal 6 *Conducting a WDSDO acceleration study*

One of the most significant challenges of WDSDO is the high computational cost associated with such an exercise, which grows exponentially in the size of the problem. For very large WDS systems found in the real-world, current generation optimisation algorithms may be insufficient for solving WDSDO within a realistic time-frame. There are several avenues of research available towards speeding up the WDSDO process.

On a physical computation level, it is well-known that current generation technology is exceedingly under-utilised. It is expected that significant speed enhancements might be achieved by employing multi-threading with numerous processors, through distributed computing (involving multiple computers communicating across networks), or using the parallel processing capabilities of modern graphics cards (especially with respect to solving the large matrix equations required in hydraulic simulation), also known as GPU programming [150]. Employing these methods alone or in tandem could potentially achieve several orders of magnitude improvement in processing speed.

Considering the optimisation-simulation framework of §3.2, there are numerous mathematical techniques that might be employed to accelerate all levels of the WDSDO process — both with

respect to the hydraulic simulation model and the stochastic demand simulator, as well as the actual optimisation algorithms.

Acceleration at a hydraulic simulation level may include:

1. the use of metamodels, such as an *Artificial Neural Network* (ANN) that replaces the traditional hydraulic simulator [22],
2. practical heuristics such as *intelligent pipe grouping* whereby groups of pipes are sized similarly to greatly reduce dimensionality, or the *Explicit Integration Method* [241] whereby subsections of the network are simulated independently and results combined for a global solution, or
3. employing specialised numerical approximation techniques for solving sparse nonlinear systems of equations (*e.g.* the method using Gaus-Seidel-iterations developed by Abou El-Seoud [3]).

Care should be taken when using any approximation techniques, since a global optimum is typically concealed and there may be a high probability of converging to a local optimum. This suggests the use of multi-resolution analysis, whereby approximation models are used predominantly to accelerate the optimisation, but in conjunction with regular accurate functional evaluation.

In an attempt at speeding up the stochastic demand simulator, one may consider the use of LHS or Descriptive Sampling [121]. Such an approach will require fewer samples than traditional Monte Carlo Simulation. Alternatively one may attempt to approximate performance by spreading samples across multiple generations (as per the RNSGA-II developed by Kapelan *et al.* [141]).

There are several methods for accelerating optimisation algorithms, including:

1. modern *initialisation strategies* for population-based algorithms (as opposed to purely random initialisations) such as Latin Hypercube or Orthogonal sampling of decision variable values and the use of heuristic initialisation,
2. the use of *evolutionary approximation techniques* such as *fitness inheritance* (whereby offspring inherit fitness from their parents without the need for simulation) and *fitness imitation* (whereby solutions are clustered according to some rule and only the fitness of a single solution per cluster is evaluated),
3. the use of multiple sub-populations with different search mechanisms or levels of fitness approximation, and the migration of solutions between sub-populations, and,
4. the use of hybrid MOEAs which incorporate machine learning techniques (*e.g.* LEMMO [140]).

A study could be undertaken to investigate the use of the acceleration techniques mentioned above towards the speed enhancement of WDSDO. The advantages and shortcomings of the various techniques may be identified, and an attempt may be made to find the most suitable / practical combination of acceleration methods for real-world WDSDO.

References

[1] Aarts EHL & Van Laarhoven PJM, 1985, *Statistical cooling: A general approach to combinatorial optimisation problems*, Philips Journal of Research, **40**, pp. 193–226.

[2] Abebe AJ & Solomatine DP, 1998, *Application of global optimisation to the design of pipe networks*, Proceedings of the 3rd International Conference on Hydroinformatics, Copenhagen, Denmark, pp. 989–996.

[3] Abou El-Seoud, 1996, *A fast approximate solution of a large sparse nolinear system by monotone convergent iterations*, International Journal of Computer Mathematics, **61(1)**, pp. 119–136.

[4] Afshar MH, Akbari M & Mario MA, 2005, *Simultaneous layout and size optimisation of water distribution networks: Engineering approach*, Journal of Infrastructure Systems, **11(4)**, pp. 221–230.

[5] Afshar MH, Akbari M & Mario MA, 2007, *A parameter-free self-adapting boundary genetic search for pipe network optimisation*, Computer Optimisation Applications, **37**, pp. 83–102.

[6] Ahn CW, 2006, *Advances in evolutionary algorithms: Theory, design and practice*, Studies in Computational Intelligence, **18**, Springer-Verlag, Berlin.

[7] Ahn CW & Ramakrishna RS, 2007, *Multiobjective real-coded Bayesian optimisation algorithm revisited: diversity preservation*, Proceedings of the Genetic and Evolutionary Computation Conference (GECCO 2007), London, pp. 593–600.

[8] Anton H & Rorres C, 2000, *Elementary linear algebra (8th Edition)*, John Wiley & Sons, Inc., New York (NY).

[9] Alperovits E & Shamir U, 1977, *Design of optimal water distribution systems*, Water Resources Research, **13(6)**, pp. 885–900.

[10] Alves MJ & Almeida M, 2007, *MOTGA: A multiobjective Tchebycheff based genetic algorithm for the multidimensional knapsack problem*, Computers & Operations Research, **34**, pp. 3458–3470.

[11] Aoki Y, 1998, *Flow analysis considering pressures* (Japanese), 49th National Meeting on Waterworks, pp. 262–263.

[12] Aporntewan C & Chongstitvatana P, 2004, *Simultaneity matrix for solving hierarchically decomposable functions*, Proceedings of the Genetic and Evolutionary Computation Conference (GECCO 2004), Seattle (WA), pp. 877–888.

[13] Atiquzzaman M, Liong S & Yu X, 2006, *Alternative decision making in water distribution network with NSGA-II*, Journal of Water Resources Planning and Management, **132(2)**, pp. 122–126.

[14] American Water Works Association, 1998, *Distribution system requirements for fire protection*, AWWA Manual M-31, Denver (CO).

[15] Awumah K, Goulter I & Bhatt S, 1990, *Assessment of reliability in water distribution networks using entropy based measures*, Stochastic Hydrology and Hydraulics, **4**, pp. 325–326.

[16] Babayan AV, Kapelan Z, Savic DA & Walters GA, 2005, *Least cost design of water distribution network under demand uncertainty*, Journal of Water Resources Planning and Management, **131(5)**, pp. 375–382.

[17] Babayan AV, Kapelan Z, Savic DA & Walters GA, 2006, *Comparison of two methods for the stochastic least cost design of water distribution systems*, Engineering Optimisation, **38(3)**, pp. 281–297.

[18] Basile N, Fuamba M & Barbeau B, 2008, *Optimisation of water tank design and location in water distributions systems*, Proceedings of the 10th Annual Water Distribution Systems Analysis Conference (WDSA 2008), Kruger National Park, South Africa.

[19] Beausoleil RP, 2008, *MOSS-II: Tabu/Scatter search for nonlinear multiobjective optimization*, Advances in Metaheuristics for Hard Optimization, Springer, Berlin, pp. 39–67.

[20] Bentley Systems, 2008, *Water distribution modeling and management — WaterCAD for Bentley's Haestad Methods*, [Online], [Cited July 15th 2008], Available from `http://www.bentley.com/en-US/Products/WaterCAD/`

[21] Bragalli C, D'Ambrosio C, Lee J, Lodi A & Toth P, 2008, *Water network design by MINLP*, IBM Research Report, RC24495 (W0802-056) (Mathematics).

[22] Broad DR, Dandy GC & Maier HR, 2005, *Water distribution system optimisation using metamodels*, Journal of Water Resources Planning and Management, **131(3)**, pp. 172–180.

[23] Bullnheimer B, Hartl RF & Strauss C, 1999, *A new rank based version of the Ant System: A computational study*, Central European Journal for Operations Research and Economics, **1(7)**, pp. 25–38.

[24] Burden RL & Faires JD, 2001, *Numerical analysis*, Brooks/Cole, Pacific Grove (CA).

[25] Burke E, Kendall G, Newall J, Hart E, Ross P, & Schulenburg S, 2003, *Hyper-heuristics: an emerging direction in modern search technology*, Handbook of metaheuristics, Kluwer Academic Publishers, Norwell (MA), pp. 457–474.

[26] Cancino W, Jourdan L, Talbi E-G & Delbem ACB, 2010, *A Parallel Multi-Objective Evolutionary Algorithm for Phylogenetic Inference*, Learning and Intelligent Optimisation, Lecture Notes in Computer Science, **6073**, pp. 196–199.

[27] Chan TM, Man KF, Kwong S & Tang KS, 2007, *A jumping gene paradigm for evolutionary multiobjective optimisation*, IEEE Transactions on Evolutionary Computation, **12(2)**, pp. 143–159.

[28] Chang CS, Xu DY & Quek HB, 1999, *Pareto-optimal set based multi-objective tuning of fuzzy automatic train operation for mass transit system*, IEE Proceedings on Electric Power Applications, **146(5)**, pp. 577–583.

[29] Charles COA & Woods DJ, 1972, *Hydraulic network analysis using linear theory*, Journal of the Hydraulics Division, ASCE, **98(7)**, pp. 1157–1170.

[30] Chase DV, Walski TM & Savic DA, 2001, *Water distribution modeling*, Haestad Press, Waterbury (CT).

[31] Cheung PB, Van Zyl JE, Reis LFR, 2004, *Extension of EPANET for Pressure Driven Demand Modeling in Water Distribution System*, (Unpublished) Technical Report, Department of Civil and Urban Engineering Science, University of Johannesburg, South Africa.

[32] Cheung PB, Propato M, Bremond B & Piller O, 2006 *Water quality parameter estimation in a water distribution system*, (Unpublished) Research paper, AD-Re-A Project, French Research Ministry.

[33] Chiplunkar AV, Mehndiratta SL & Khanna P, 1986, *Looped water distribution system optimisation for single loading*, Journal of Environmental Engineering, **112(2)**, pp. 265–279.

[34] Clark R, Grayman WM & Males RM, 1988, *Contaminant propagation in distribution systems*, Journal of Environmental Engineering, **114(4)**, pp. 929–943.

[35] Coello CAC & Lamont GB, 2004, *Applications of multi-objective evolutionary algorithms*, World Scientific, London.

[36] Coello CAC, Santana-Quintero LV & Toscano-Pulido G, 2006, *EMOPSO: A multi-objective particle swarm optimizer with emphasis on effiency*, (Unpublished) Research paper, CINVESTAV-IPN (Evolutionary Computation Group), University of San Nicolas de los Garza, Mexico City, Mexico.

[37] Cohon JL, 1987, *Multi-objective programming and planning*, Academic Press, New York (NY).

[38] Conover WJ, 1999, *Practical non-parametric statistics*, Wiley, New York (NY).

[39] Corne DW, Jerram N, Knowles JD, Oates MJ, 2001, *PESA-II: region-based selection in multiobjective optimisation*, Procceedings of the Genetic and Evolutionary Computation Conference (GECCO 2001), 1, pp. 283–290.

[40] Costa ALH, De Medeiros JL & Pessoa FLP, 2000, *Optimisation of pipe networks including pumps by simulated annealing*, Brazillian Journal of Chemical Engineering, **17**, pp. 887–896.

[41] Cross H, 1936, *Analysis of flow in networks of conduits or conductors*, (Unpublished) Bulletin No. 286, University of Illinois Engineering Experimental Station, Urbana (IL).

[42] Crowe CT, Elger D & Roberson JA, 2001, *Engineering fluid mechanics*, John Wiley & Sons, Inc., New York (NY).

[43] Crowe & Roberson, 1990, *Engineering fluid mechanics*, Houghton Mifflin, New York (NY).

[44] CSIR, 2003, *Guidelines for human settlement planning and design*, The Red Book (2nd edn.), A report compiled under the patronage of the Department of Housing, South Africa.

[45] Cullinane M, Lansey K & Mays L, 1992, *Optimisation-availability based design of water distribution networks*, Journal of Hydraulic Engineering, **118(3)**, pp. 420–441.

[46] Cunha MD & Sousa J, 1999, *Water distribution network design optimisation: Simulated annealing approach*, Journal of Water Resources Planning and Management, **125(4)**, pp. 215–221.

[47] Da Conceicao CM & Ribeiro L, 2004, *Tabu search algorithms for water network optimisation*, European Journal of Operational Research, **157**, pp. 746–758.

[48] Dandy GC & Engelhardt MO, 2001, *Optimum rehabilitation of water distribution system considering cost and reliability*, Proceedings of the World Water and Environmental Resources Congress, World Water Congress 2001, Orlando, Florida, p. 399.

[49] Dandy GC & Engelhardt MO, 2001, *Optimal scheduling of water pipe replacement using genetic algorithms*, Journal of Water Resources Planning and Management, **127(4)**, pp. 214–223.

[50] Dandy GC, Simpson AR & Murphy LJ, 1993, *Review of pipe network optimisation techniques*, Proceedings of the 2nd Australasian Conference on Computing for the Water Industry Today Tomorrow, Melbourne, Australia, pp. 373–383.

[51] Dandy GC, Simpson AR & Murphy LJ, 1996, *An improved genetic algorithm for pipe network optimisation*, Water Resources Research, **32(2)**, pp. 449–458.

[52] Davidson JW, 1999, *Evolution program for layout geometry of rectilinear looped networks*, Journal of Computing in Civil Engineering, **13(4)**, pp. 246–253.

[53] Davidson JW & Goulter IC, 1995, *Evolution program for design of rectilinear branched networks*, Journal of Computing in Civil Engineering, **9(2)**, pp. 112–121.

[54] Dawkins R, 1976, *The selfish gene*, Oxford University Press, New York (NY).

[55] Day RO & Lamont GB, 2005, *Extended multi-objective fast messy genetic algorithm solving deception problems*, Proceedings of the 5th International Conference on Evolutionary Multi-Criterion Optimisation (EMO 2005), Springer-Verlag, Berlin, pp. 296-310.

[56] Deb K, 1998, *An efficient constraint handling method for genetic algorithms*, (Unpublished) Technical Report, Kanpur Genetic Algorithms Laboratory (KanGAL), Department of Mechanical Engineering, Indian Institute of Technology Kanpur, Kanpur.

[57] Deb K, 1999, *Multi-objective genetic algorithms: Problem difficulties and construction of test problems*, Evolutionary Computation, **7(3)**, pp. 205–230.

[58] Deb K & Agrawal RB, 1994, *Simulated binary crossover for continuous search space*, (Unpublished) Technical Report, Department of Mechanical Engineering, Indian Institute of Technology, Kanpur.

[59] Deb K & Goyal M, 1996, *A combined genetic adaptive search (geneas) for engineering design*, Computer Science and Informatics **26(4)**, pp. 30–45.

[60] Deb K & Goel T, 2001, *Controlled elitist non-dominated sorting genetic algorithms for better convergence*, Proceedings of the 1st International Conference on Evolutionary Multi-Criterion Optimisation (EMO 2001), Zurich, Switzerland, pp. 67-81.

[61] Deb K, Pratap A, Agarwal S & Meyarivan T, 2002, *A fast and elitist multi-objective genetic algorithm - NSGA-II*, IEEE Transactions on Evolutionary Computation, **6(2)**, pp. 182–197.

[62] Deb K, Mohan M & Mishra S, 2003, *A fast multi-objective evolutionary algorithm for finding well-spread pareto-optimal solutions*, (Unpublished) KanGAL Report No. 2003002, Kanpur Genetic Algorithms Laboratory (KanGAL), Indian Institute of Technology, Kanpur.

[63] Deb K, Karthic S & Okabe T, 2007, *Self-adaptive simulated binary crossover for real-parameter optimisation*, Procceedings of the Genetic and Evolutionary Computation Conference (GECCO 2007), London, England.

[64] Deb K & Tiwari S, 2008, *Omni-optimizer: A generic evolutionary algorithm for single and multi-objective optimisation*, European Journal of Operational Research, 185, pp. 1062–1087.

[65] DEIS Operations Research Group, 2010, *LabOR - DEIS Operations Research Group - Library of Instances*, [Online], [Cited 06 July 2010], Available from `http://www.or.deis.unibo.it/research_pages/ORinstances/ORinstances.htm`.

[66] Department of Water Affairs and Forestry, 2003, *South African strategic framework for water services*, [Government Report], Government Publications, Pretoria, South Africa.

[67] De Schaetzen WBF, Walters GA & Savic DA, 2000, *Optimal sampling design for model calibration using shortest path, genetic and entropy algorithms*, Urban Water, **2(2000)**, pp. 141–152.

[68] di Pierro F, Khu S-T, Savic D & Berardi L, 2009, *Efficient multi-objective optimal design of water distribution networks on a budget of simulations using hybrid algorithms*, Environmental Modelling & Software, 24, pp. 202–213.

[69] Dimou CK & Koumousis VK, 2003, *Genetic Algorithms in Competitive Environments*, Journal of Computing in Civil Engineering, **17(3)**, pp. 142–149.

[70] Dorigo M & Di Caro G, 1999, *The ant colony optimisation meta-heuristic*, pp. 11–32 in Corne D, Dorigo M, Glover F, Dasgupta D, Moscato P, Poli R & Price K (Eds), *New ideas in optimisation*, McGraw-Hill, New York (NY).

[71] Dorigoa M & Blumb C, 2005, *Ant colony optimisation theory: A survey*, Theoretical Computer Science **344**, pp. 243-278.

[72] Dorigo M & Gambardella LM, 1997, *Ant colony system: A cooperative learning approach to TSP*, IEEE Transactions in Evolutionary Computation, **11**, pp. 53–66.

[73] Dreo J, Petrowski A, Siarry P & Talliard E, 2006, *Metaheuristics for hard optimisation: Methods and case studies*, Springer-Verlag, Berlin.

[74] Duan N, Mays LW & Lansey KE, 1990, *Optimal reliability based design of pumping and distribution systems*, Journal of Hydraulic Engineering, **116(2)**, pp. 249–268.

[75] Duan Q, Sorooshian S & Gupta V, 1992, *Effective and efficient global optimisation for conceptual rainfall-runoff models*, Water Resources Research, **28(4)**, pp. 1015–1031.

[76] Duan QY, Gupta VK & Sorooshian S, 1993, *Shuffled complex evolution approach for effective and efficient minimization*, Journal of Optimisation Theory and Applications, **76(3)**, pp. 501–521.

[77] Duan QY, Sorooshian S & Gupta VK, 1994, *Optimal use of the SCE-UA global optimisation method for calibrating watershed models*, Journal of Hydraulic Engineering, **158(1)**, pp. 265–284.

[78] Eiger G, Shamir U & Ben-Tal A, 1994, *Optimal design of water distribution networks*, Water Resources Research, **30(9)**, pp. 2637–2646.

[79] El-Bahrawy A & Smith A, 1987, *A methodology for optimal design of pipe distribution networks*, Canadian Journal of Civil Engineering, **14**, pp. 207–215.

[80] Elbeltagia E, Hegazyb T & Griersonb D, 2005, *Comparison among five evolutionary-based optimisation algorithms*, Advanced Engineering Informatics, **19**, pp. 43-53.

[81] Environmental Protection Agency (US), 2008, *EPANET — Drinking water research — US EPA*, [Online], [Cited August 20th, 2008], Available at `http://www.epa.gov/NRMRL/wswrd/dw/epanet.html`

[82] Eskandari H, Geiger CD, & Lamont GB, 2007, *FastPGA: A dynamic population sizing approach for solving expensive multiobjective optimisation problems*, Proceedings of Evolutionary Multi-Criterion Optimisation, 4th International Conference (EMO 2007), Springer-Verlag, Berlin, pp. 141–155.

[83] Eusuff MM & Lansey KE, 2003, *Optimisation of water distribution network design using the shuffled frog leaping algorithm*, Journal of Water Resources Planning and Management, **129(3)**, pp. 210–225.

[84] Exeter Center for Water Systems, 2007, *Center for Water Systems, University of Exeter — Projects Page*, [Online], [Cited 02 March 2007], Available from `http://www.projects.ex.ac.uk/cws/index.php`.

[85] Fanni A, Liberatore S, Sechi GM, Soro M & Zuddas P, 2000, *Optimisation of water distribution systems by a tabu search metaheuristic*, Proceedings of the 7th INFORMS Computing Society Conference, Cancun.

[86] Farmani R, Savic DA & Walters GA, 2003, *Multi-objective optimisation of water system: A comparative study*, Paper presented at conference on Pumps, Electromechanical Devices and Systems Applied to Urban Water Management, 2003, Institute for Water Technology, Valencia, Spain.

[87] Farmani R, Walters GA & Savic DA, 2005, *Trade-off between total cost and reliability for Anytown water distribution network*, Journal of Water Resources Planning and Management, **131(3)**, pp. 161–171.

[88] Farmani R, Walters GA & Savic DA, 2005, *Evolutionary multi-objective optimisation in water distribution network design*, Engineering Optimisation, **37(2)**, pp. 167-183.

[89] Featherstone R & El-Jumaily K, 1983, *Optimal diameter selection for pipe networks*, Journal of Hydraulic Engineering, **109(2)**, pp. 221–233.

[90] Filion YR & Jung BS, 2008, *Particle swarm optimisation of water distribution networks with economic damages*, Proceedings of the 10th Annual Water Distribution Systems Analysis Conference (WDSA 2008), Kruger National Park, South Africa.

[91] Fonseca CM & Fleming PJ, 1993, *Genetic algorithms for multiobjective optimisation: formulation, discussion and generalization*, Proceedings of the 5th International Conference on Genetic Algorithms, San Mateo California, Morgan Kauffman Publishers, pp. 416–423.

[92] Fonseca CM & Fleming PJ, 1997, *Multi-objective optimisation*, in Back IT, Fogel DB, Michalewicz Z (Eds), *Handbook of Evolutionary Computation*, Institute of Physics Publishing and Oxford University Press, Oxford, United Kingdom.

[93] Freedman RA & Young HD, 2003, *University Physics (*11th *Edition)*, Addison Wesley, San Francisco (CA).

[94] Fujiwara O, Jenchaimahakoon B & Edirisinghe NCP, 1987, *A modified linear programming gradient method for optimal design of looped water distribution networks*, Water Resources Research, **23(6)**, pp. 977–982.

[95] Fujiwara O & Khang D, 1990, *A two-phase decomposition method for optimal design of looped water distribution networks*, Water Resources Research, **26(4)**, pp. 539–549.

[96] Geem ZW, Kim JH & Yoon YN, 2000, *Parameter calibration of the nonlinear muskingum model using harmony search*, Proceedings of the Korea Water Resources Association Coference, YongIn, Korea.

[97] Geem ZW, Kim JH & Loganathan GV, 2002, *Harmony search optimisation: Application to pipe network design*, International Journal of Modelling and Simulation, **22(2)**, pp. 125–133.

[98] Geem ZW, 2009, *Harmony search optimisation to the pump-included water distribution network design*, Civil Engineering and Environmental Systems, **26(3)**, pp. 211–221.

[99] Gessler J, 1985, *Pipe network optimisation by enumeration*, Proceedings of the Speciality Conference on Computer Applications in Water Resources, American Society of Civil Engineers, New York (NY).

[100] Giustolisi O, Savic D & Kapelan Z, 2008, *Pressure-driven demand and leakage simulation for water distribution networks*, Journal of Hydraulic Engineering, **134(5)**, pp. 626–635.

[101] Glover F & Laguna M, 1998, *Tabu Search: A manual*, Springer, Berlin.

[102] Glover F, 2001, *Future paths for integer programming and links to artificial intelligence*, Computers and Operations Research, **13**, p. 533.

[103] GLS Software (Pty) Ltd, 2007, *Wadiso 5.3*, Water Distribution System Optimisation Software, Stellenbosch, South Africa.

[104] Goldberg DE, 1983, *Computer-aided gas pipeline operation using genetic algorithms and rule learning*, PhD Disseration, University of Michigan, Ann Arbor (MI).

[105] Goldberg DE & Kuo CH, 1987, *Genetic algorithms in pipeline optimisation*, Journal of Computing in Civil Engineering, **1(2)**, pp. 128–141.

[106] Goldberg DE, 1989, *Genetic algorithms in search, optimisation, and machine learning*, Addison Wesley, San Francisco (CA).

[107] Goldberg DE, Deb K, Kargupta H & Harik G, 1993, *Rapid, accurate optimisation of difficult problems using fast messy genetic algorithms*, (Unpublished) Report No. 93004, Illinois Genetic Algorithms Laboratory, University of Illinois (IL).

[108] Goldman FE & Mays LW, 2005, *Water distribution system operation: Application of simulated annealing approach*, in Mays LW (Ed), *Water resources systems management tools*, Mcgraw-Hill Professional, New York (NY).

[109] Goulter IC, 1988, *Measures of internal redundancy in water distribution network layouts*, Journal of Information and Optimisation Science, **9**, pp. 363–390.

[110] Goulter IC, 1992, *Systems analysis in water-distribution network design: From theory to practice*, Journal of Water Resources Planning and Management, **118(3)**, pp. 238–248.

[111] Goulter IC & Morgan DR, 1985, *An integrated approach to the layout and design of water distribution networks*, Civil Engineering Systems, **2(2)**, pp. 104–113.

[112] Goulter IC, Walski TM, Mays LW, Sekarya ABA, Bouchart R & Tung YK, 2000, *Reliability analysis for design*, in Mays LW (Ed), *Water Distribution Systems Handbook*, McGraw-Hill, New York (NY).

[113] Gross JL & Yellen J, 2005, *Graph theory and its applications*, Chapman and Hall, London.

[114] Gupta I, Gupta A & Khanna P, 1999, *Genetic algorithm for the optimisation of water distribution systems*, Environmental Modelling & Software, **4**, pp. 437–446.

[115] Hadji G & Murphy LJ, 1990, *Genetic algorithms for pipe network optimisation*, Final Year Student Civil Engineering Research Report, University of Adelaide, Adelaide.

[116] Halhal D, Walters GA, Ouazar D & Savic DA, 1997, *Water network rehabilitation with structured messy genetic algorithms*, Journal of Water Resources Planning and Management, **123(3)**, pp. 137–146.

[117] Halhal D, Walters GA, Savic DA & Ouazar D, 1999, *Scheduling of water distribution system rehabilitation using structured messy genetic algorithms*, Evolutionary Computation, **7(3)**, pp. 311–329.

[118] Hansen CT, Madsen K & Nielsen HB, 1991, *Optimisation of pipe networks*, Mathematical Programming, **52**, pp. 45–58.

[119] Harik GR & Lobo FG, 1999, *A parameter-less genetic algorithm*, in Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-99), Morgan Kaufmann, pp. 258–267.

[120] Hasover AM & Lind NC, 1974, *Exact and invariant second moment code format*, Journal of the Engineering Mechanics Division, ASCE, **100(1)**, pp. 111–121.

[121] Hess S, Train KE & Polak JW, 2004, *On the use of a modified latin hypercube sampling (MLHS) method in the estimation of a mixed logit model for vehicle choice*, (Unpublished) Manuscript, Imperial College, London.

[122] Hoag LN & Weinberg G, 1957, *Pipeline network analysis by digital computers*, Journal of the American Water Works Association, **49**, pp. 517–524.

[123] Holland JH, 1975, *Adaptation in natural and artificial systems*, University of Michigan Press, Ann Arbor (MI).

[124] Holloway MB, 1985, *Dynamic pipe network computer model*, PhD disertation, Washington State Univesity, Pullman (WA).

[125] Horoba C & Neumann F, 2008, *Benefits and drawbacks for the use of epsilon-dominance in evolutionary multi-objective optimisation*, Proceedings of the 10th Genetic and Evolutionary Computation Conference (GECCO-08), ACM, pp. 641–648.

[126] Huddleston DH, Alarcon VJ & Chen W, 2004, *Water distribution network analysis using excel*, Journal of Hydraulic Engineering, **130(10)**, pp. 1033–1035.

[127] Hytran Systems, 2008, *Water hammer software for Windows from Hytran Solutions*, [Online], [Cited July 15th 2008], Available from `http://www.hytran.net/`

[128] Iman RL & Conover WJ, 1982, *A distribution-free approach to inducing rank correlation among input variables*, Communications in Statistics - Simulation and Computation, **11(3)**, pp. 311–334.

[129] Insurance Services Office (ISO), 1998, *Fire suppression rating schedule*, New York (NY).

[130] Iorio AW & Li X, 2004, *A Cooperative Coevolutionary Multiobjective Algorithm Using Non-dominated Sorting*, in Proceedings of the Genetic and Evolutionary Computation Conference (GECCO 2004), Springer-Verlag, Berlin, pp. 537–548.

[131] International Water Association, 2008, *Reference paper archive*, [Online], [Cited October 7th, 2008], Available from `http://www.iwahq.org`

[132] Izquierdo J, Montalvo I, Peacuterez R & Iglesias PL, 2008, *A diversity-enriched variant of discrete PSO applied to the design of water distribution networks*, Engineering Optimisation, **40(7)**, pp. 655–668.

[133] Jacobs P & Goulter IC, 1988, *Evaluation of methods for decomposition of water distribution networks for reliability analysis*, Civil Engineering Systems, **5**, pp. 58–64.

[134] Jacobsen LB, Dishari MA, Murphy LJ & Frey J, 1998, *Las Vegas valley water district plans for expansion improvements using genetic algorithm optimisation*, Proceedings of the AWWA Information Management and Technology Conference, American Water Works Association, Reno (NV).

[135] Jacoby SLS, 1968, *Design of optimal hydraulic networks*, Journal of the Hydraulics Division, ASCE, **94(3)**, p. 641.

[136] Janna S, 1998, *Design of thermal systems (2nd Edition)*, PWS Publishing, Boston (MA).

[137] Jaszkiewicz A, 1998, *Genetic local search for multiple objective combinatorial optimization*, Technical Report RA-014/98, Institute of Computing Science, Poznan University of Technology.

[138] Jensen MT, 2003, *Reducing the run-time complexity of multiobjective EAs: The NSGA-II and other algorithms*, IEEE Transactions on Evolutionary Computation, **7(5)**, pp. 503–515.

[139] Jeppson RW, 1976, *Analysis of flow in pipe networks*, Ann Arbor Science Publishers, Ann Arbor (MI).

[140] Jourdan L, Corne DW, Savic DA & Walters G, 2006, *LEMMO: hybridising rule induction and NSGA II for multi-objective water systems design*, Procceedings of the Eighth International Conference on Computing and Control for the Water Industry, 2, pp. 45–50.

[141] Kapelan ZS, Savic DA & Walters GA, 2005, *Multiobjective design of water distribution systems under uncertainty*, Water Resources Research, **41(1)**, pp. 1–15.

[142] Keedwell E & Khu ST, 2006, *Novel cellular automata approach to optimal water distribution network design*, Journal of Computing in Civil Engineering, **20(1)**, pp. 49–56.

[143] Keedwell E & Khu ST, 2006, *A novel evolutionary metaheuristic for the multiobjective optimisation of real-world water distribution networks*, Engineering Optimisation, **38(3)**, pp. 1–18.

[144] Kennedy J & Eberhart RC, 1995, *Particle swarm optimisation*, Proceedings of the 1995 IEEE International Conference on Neural Networks, Piscataway (NJ), pp. 1942–1948.

[145] Kim JH & Mays LW, 1994, *Optimal rehabilitation model for water-distribution systems*, Journal of Water Resources Planning and Management, **120(5)**, pp. 674–692.

[146] Kirkpatrick S, Gelatt CD (Jr) & Vecchi MP, 1983, *Optimisation by simulated annealing*, Science, **220**, pp. 671–680.

[147] Knowles J, 2006, *ParEGO: a hybrid algorithm with on-line landscape approximation for expensive multiobjective optimisation problems*, IEEE Transactions on Evolutionary Computation, **10(1)**, pp. 50–66.

[148] Knowles JD, Thiele L & Zitzler E, 2006, *A tutorial on the performance assessment of stochastic multiobjective optimizers*, (Unpublished) TIK-Report No. 214, Computer Engineering and Network Laboratory (TIK), Swiss Federal Institute of Technology (ETH), Zurich.

[149] Kollat JB & Reed PM, 2005, *The value of online adaptive search: A performance comparison of NSGAII, ϵ-NSGAII and ϵ-MOEA*, Proceedings of the 5$^{\text{th}}$ International Conference on Evolutionary Multi-Criterion Optimisation (EMO 2005), Springer-Verlag, Berlin, pp. 386–398.

[150] Krüger J, Schiwietz T, Kipfer P & Westermann R, 2004, *Numerical simulations on PC graphics hardware*, ParSim 2004 (Special Session of EuroPVM/MPI 2004), Budapest, Hungary.

[151] Kukkonen S & Lampinen J, 2004, *An extension of generalized differential evolution for multiobjective optimisation with constraints*, Proceedings of the 8th International Conference on Parallel Problem Solving from Nature (PPSN 2004), Birmingham, pp. 752–761.

[152] Kukkonen S & Lampinen J, 2005, *GDE3: The third evolution step of generalized differential evolution*, (Unpublished) KanGAL Report No. 2005013, Kanpur Genetic Algorithms Laboratory, Indian Institute of Technology, Kanpur.

[153] Kulturel-Konak S, Konak A & Coit DW, 2007, *Multi-objective metaheuristic approaches to reliability optimisation*, Computational Intelligence in Reliability Engineering, **39**, pp. 37–62.

[154] KY Pipe Inc., 2008, *Welcome to Kypipe*, [Online], [Cited July 15th 2008], Available from `http://www.kypipe.com`

[155] Lam CF, 1973, *Discrete gradient optimisation of water systems*, Journal of the Hydraulics Division, ASCE, **99(6)**, pp. 863–872.

[156] Lampinen J, 2001, *DE's selection rule for multiobjective optimisation*, (Unpublished) Technical Report, Department of Information Technology, Lappeenranta University of Technology, Lappeenranta.

[157] Lansey KE, 2000, *Optimal design of water distribution systems*, in Mays LW (Ed), *Water distribution systems handbook*, McGraw-Hill, New York (NY).

[158] Lansey KE, Duan N, Mays LW & Tung YK, 1989, *Water distribution system design under uncertainties*, Journal of Hydraulic Engineering, **115(10)**, pp. 1401–1419.

[159] Lansey KE & Mays LW, 1989, *Optimisation model for water distribution system design*, Journal of Hydraulic Engineering, **115(10)**, pp. 1401–1419.

[160] Lasdon LS & Waren AD, 1982, *GRG2 user's guide*, Department of General Business, University of Texas, Austin (TX).

[161] Laumanns M, Thiele L, Deb K & Zitzler E, 2002, *Combining convergence and diversity in evolutionary multiobjective optimisation*, Evolutionary Computation, **10(3)**, pp. 263–282.

[162] Liong S & Atiquzzaman MD, 2004, *Optimal design of water distribution network using shuffled complex evolution*, Journal of the Institution of Engineers, Singapore, **44(1)**, pp. 93–107.

[163] Lippai I, Heaney JP & Laguna M, 1999, *Robust water system design with commercial intelligent search optimizers*, Journal of Computation in Civil Engineering, **13(3)**, pp. 135–143.

[164] Lobo FG & Goldberg DE, 2004, *The parameter-less genetic algorithm in practice*, Information Sciences **167**, pp. 217–232.

[165] Lobo FG & Lima CF, 2005, *A review of adaptive population sizing schemes in genetic algorithms*, Proceedings of the Genetic and Evolutionary Computation Conference (GECCO 2005), ACM, Washington (DC).

[166] Loganathan G, Greene J & Ahn T, 1995, *Design heuristic for globally minimum cost water distribution systems*, Journal of Water Resources Planning and Management, **121(2)**, pp. 182–192.

[167] Madavan NK, 2002, *Multiobjective optimisation using a pareto differential evolution approach*, Proceedings of the 2002 Congress on Evolutionary Computation (CEC 2002), Honolulu, pp. 1145–1150.

[168] Maier HR, Simpson AR, Zecchin AC, Foong WK, Phang KY, Seah HY & Tan CL, 2003, *Ant colony optimisation for design of water distribution systems*, Journal of Water Resources Planning and Managemment, **129(3)**, pp. 200–209.

[169] Mays LW (Ed), 2000, *Water distribution systems handbook*, McGraw-Hill, New York (NY).

[170] Mays LW, 2005, *Water resources engineering*, John Wiley & Sons, Inc., New York (NY).

[171] McClintock B, 1950, *The origin and behavior of mutable loci in maize*, Proceedings of the National Academy of Science, **36**, pp. 344–355.

[172] McClintock B, 1951, *Chromosome organization and genic expression*, Cold Spring Harbor Symposia on Quantitative Biology, **16**, pp. 13–47.

[173] McKay MD, Conover WJ & Beckman RJ, 1979, *A comparison of three methods for selecting values of input variables in the analysis of output from a computer code*, Technometrics, **211**, pp. 239–245.

[174] Mohan S & Babu KSJ, 2010, *Optimal water distribution network design with honey-bee mating optimisation*, Journal of Computing in Civil Engineering, **24(1)**, pp. 117–126.

[175] Monbaliu J, Jo JH, Fraisse CW & Vadas RG, 1990, *Computer aided design of pipe networks*, in Simonovic SP, Goulter IC, Lence BJ (Eds), *Water resources systems application*, Friesen Printers, Winnipeg.

[176] Metaheuristics Network, *Metaheuristics network — Algorithms*, [Online], [Cited July 10th 2007], Available at `http://www.metaheuristics.net`

[177] Mhlenbein H, 1997, *The equation for response to selection and its use for prediction*, Evolutionary Computation, **5(3)**, pp. 303–346.

[178] Murphy LJ & Simpson AR, 1992, *Genetic algorithms in pipe network optimisation*, (Unpublished) Research Report No. R93, Department of Civil and Environmental Engineering, University of Adelaide, Adelaide.

[179] MWH Soft, 2008, *Products — Infowater*, [Online], [Cited July 15th 2008], Available from `http://www.mwhsoft.com`

[180] Nebro AJ, Durillo JJ, Luna F, Dorronsoro B & Alba E, 2007, *MOCell: A cellular genetic algorithm for multiobjective optimisation*, International Journal of Intelligent Systems, pp. 25–36.

[181] Nelder JA & Mead R, 1965, *A simplex method for function minimization*, Computer Journal, **7**, pp. 308–313.

[182] Nicolini M, 2004, *Evaluating performance of multi-objective genetic algorithms for water distribution system optimisation*, Phoon S-Y, Liong K-K & Babovic V (eds.), Sixth International Conference on Hydroinformatics, World Scientific Publishing Company, **1**, pp. 850–857.

[183] Nikuradse J, 1933, *Stromungsgesetze in rauhen Rohren ("Laws of flow in rough pipes")*, VDI-Forschungsheft, **361**.

[184] Nunoo C & Mrawira D, 2004, *Shuffled complex evolution algorithms in infrastructure works programming*, Journal of Computing in Civil Engineering, **18(3)**, pp. 256–266.

[185] Olsson RJ, Kapelan Z & Savic DA, 2009, *Probabilistic building block identification for optimal design and rehabilitation of water distribution systems*, Journal of Hydroinformatics, **11(2)**, pp. 89–105.

[186] Optiz EM, Langowski JF, Hanna-Somers NA, Willett JS & Hauer RJ, 1998, *Forecasting urban water use: Models and application*, in Baumann DD, Boland J & Hanemann WM (Eds), *Urban water demand management and planning*, McGraw-Hill Professional, New York (NY).

[187] Ormsbee L & Contractor D, 1981, *Optimisation of hydraulic networks*, Proceedings of the International Symposium on Urban Hydrology, Hydraulics and Sediment Control, Lexington (KY), pp. 255–261.

[188] Ostfeld A, Kogan D & Shamir U, 2002, *Reliability simulation of water distribution systemssingle and multiquality*, Urban Water, **4(1)**, pp. 53–61.

[189] Oxford English Dictionary, [Online], [Cited October 30th, 2008], Available at `http://www.oed.com`

[190] Oyama A, Shimoyama K & Fujii K, 2005, *New constraint-handling method for multi-objective multi-constraint evolutionary optimisation and its application to space plane design*, in Schilling R, Haase W, Periaux J, Baier H & Bugeda G (eds), *Evolutionary and deterministic methods for design, optimisation and control with applications to industrial and societal problems*, EUROGEN 2005, FLM, Munich.

[191] Pelikan M & Goldberg DE, 2000, *Hierarchical problem solving by the Bayesian optimisation algorithm*, Proceedings of the Genetic and Evolutionary Computation Conference (GECCO 2000), Morgan Kaufmann, Las Vegas, (NV), pp. 267–274.

[192] Todini E & Pilati S, 1987, *A gradient method for the analysis of pipe networks*, Paper presented at the International Conference on Computer Applications for Water Supply and Distribution 1987, Leicester Polytechnic, Leicester.

[193] Powell MJD, 1978, *Algorithms for nonlinear constraints that use Lagrangian functions*, Mathematical Programming, **14**, pp. 224–248.

[194] Prasad TD & Park N-S, 2004, *Multiobjective genetic algorithms for design of water distribution networks*, Journal of Water Resources and Planning Management, **130(1)**, pp. 73–82.

[195] Prasad TD & Tanyimboh TT,2008, *Entropy based design of 'Anytown' water distribution network*, Proceedings of the 10th Annual Water Distribution Systems Analysis Conference (WDSA 2008), Kruger National Park, South Africa.

[196] Prodhon C & Prins C, 2008, *A Memetic Algorithm with Population Management (MA—PM) for the Periodic Location-Routing Problem*, in Proceedings of the 5th International Workshop on Hybrid Metaheuristics (HM 2008), Springer, pp. 43–57.

[197] Quindry G, Brill E & Liebman J, 1981, *Optimisation of looped water distribution systems*, Journal of Environmental Engineering, **107(4)**, pp. 665–679.

[198] Quinlan JR, 1993, *C4.5: Programs for machine learning*, Morgan Kaufmann Publishers, San Francisco (CA).

[199] Raad DN, 2008, *R21 Corridor water distribution system benchmark*, [Online], [Cited November 30th, 2008], Available at `http://www.vuuren.co.za/main.php` → `benchmarks`

[200] Raghuwanshi MM & Kakde OG, 2001, *Survey on multiobjective evolutionary and real coded genetic algorithms*, Complexity International, **11**, pp. 150–161.

[201] Reddy MJ & Kumar DN, 2007, *Multiobjective differential evolution with application to reservoir system optimisation*, Journal of Computing in Civil Engineering, **21(2)**, pp. 136–146.

[202] Robič T & Filipič B, 2005, *DEMO: Differential evolution for multiobjective optimisation*, Proceedings of the 3rd International Conference on Evolutionary Multi-Criterion Optimisation (EMO 2005), Guanajuato, pp. 520–533.

[203] Rossman LA, 2000, *Computer models/EPANET*, in Mays LW (Ed), *Water distribution systems handbook*, McGraw-Hill, New York (NY).

[204] Rossman LA, 2000, *EPANET 2 users manual*, Water Supply and Water Resources Division, National Risk Management Research Laboratory, U.S. Environmental Protection Agency, Cincinnati (OH).

[205] Saldarriaga JG, Bernal A & Ochoa S, 2008, *Optimized design of water distribution network enlargements using resilience and dissipated power concepts*, Proceedings of the 10th Annual Water Distribution Systems Analysis Conference (WDSA 2008), Kruger National Park, South Africa, pp. 298–312.

[206] Salgado R, Todini E & O'Connell PE, 1987, *Comparison of the gradient method with some traditional methods for the analysis of water supply distribution networks*, Proceedings of the International Conference on Computer Applications for Water Supply and Distribution, Leicester Polytechnic, Leicester.

[207] Ministry of Water Affairs and Forestry, 2003, *South African strategic framework for water services*, [Government Report], Government Publications, Pretoria.

[208] Savic DA, 2008, Co-director of Centre for Water Systems in Exeter, UK, [Personal Communication], contactable at `d.savic@exeter.ac.uk`

[209] Savic DA & Walters GA, 1997, *Genetic algorithms for least-cost design of water distribution networks*, Journal of Water Resources Planning and Management, **123(2)**, pp. 67–77.

[210] Savic DA & Walters GA, 1995, *Genetic operators and constraint handling for pipe network optimisation*, Evolutionary Computing, **2**, pp. 154–165.

[211] Savic DA, Walters GA, Randall-Smith M & Atkinson RM, 2000, *Large water distribution systems design through genetic algorithm optimisation*, Proceedings of the ASCE Joint Conference on Water Resources Engineering and Water Resources Planning and Management, Minneapolis (MI).

[212] Schaake JC & Lai D, 1969, *Linear programming and dynamic programming applications to water distribution network design*, (Unpublished) Technical Report 116, Hydrodynamic Laboratory, Department of Civil Engineering, Massachusetts Institute of Technology, Cambridge (MA).

[213] Schaffer D, 1985, *Multiple objective optimisation with vector evaluated genetic algorithms*, Genetic Algorithms and their Applications: Proceedings of the 1st International Conference on Genetic Algorithms, Lawrence Erlbaum, Hillsdale (NJ), pp. 93–100.

[214] Shamir U, 1974, *Optimal design and operation of water distribution systems*, Water Resources Research, **10(1)**, pp. 27–35.

[215] Shannon CE, 1948, *A mathematical theory of communication*, The Bell System Technical Journal, **27**, pp. 623–656.

[216] Sherali HD, Totlani R & Loganathan GV, 2001, *Effective relaxation and partitioning schemes for solving water distribution network design problems to global optimality*, Journal of Global Optimisation, **19**, pp. 1–26.

[217] Schlierkamp-Voosen D & Muhlenbein H, 1994, *Strategy adaption by competing subpopulations*, Parallel Problem Solving from Nature (PPSN III), Springer-Verlag, pp. 199208.

[218] Simpson AR, Dandy GC & Murphy LJ, 1994, *Genetic algorithms compared to other techniques for pipe optimisation*, Journal of Water Resources Planning and Management, **120(4)**, pp. 423–443.

[219] Simpson AR, Maier HR, Foong WK, Phang KY, Seah HY & Tan CL, 2001, *Selection of parameters for ant colony optimisation applied to the optimal design of water distribution systems*, Proceedings of the International Congress on Modelling and Simulation (MODSIM 2001), Canberra.

[220] Sinske AN, 1993, *Development and inclusion of matrix solving techniques for reservoir storage sizing in the WADISO water distribution system optimisation program*, B Eng Thesis (Civil Engineering), University of Stellenbosch, Stellenbosch.

[221] Storn R & Price K, 1997, *Differential evolution: A simple and efficient heuristic for global optimisation over continuous spaces*, Journal of Global Optimisation, **11**, pp. 341–359.

[222] Stützle T & Hoos HH, 2000, *MAX-MIN ant systems*, Future Generation Computer Systems, **16**, pp. 889–914.

[223] Su YC, Mays LW, Duan N & Lansey KE, 1987, *Reliability based optimisation model for water distribution systems*, Journal of Hydraulic Engineering, **113(12)**, pp. 1539–1556.

[224] Tabesh M, Soltani J, Farmani R & Savic D, 2009, *Assessing pipe failure rate and mechanical reliability of water distribution networks using data driven modelling*, Journal of Hydroinformatics, **11(1)**, pp. 1–17.

[225] Tanyimboh TT & Templeman AB, 1993, *Calculating maximum entropy flows in networks*, Journal of the Operational Research Society, **44(4)**, pp. 383–396.

[226] Tanyimboh TT & Templeman AB, 2005, *Modelling errors, entropy and the hydraulic reliability of water distribution systems*, Advances in Engineering Software, **36**, pp. 780–788.

[227] Todini E, 2000, *Looped water distribution networks design using a resilience index based heuristic approach*, Urban Water, **2**, pp. 115–122.

[228] Todini E, 2003, *A more realistic approach to the "extended period simulation" of water distribution networks*, Advances in Water Supply Management, Maksimovic C, Butler D, Memon FA (Eds), Swets and Zellinger, Lisse, 1, pp. 173–183.

[229] Todini E, 2008, *Design, expansion and rehabilitation of water distribution networks aimed at reducing water losses: Where are we?*, Proceedings of the 10th Annual Water Distribution Systems Analysis Conference (WDSA 2008), Kruger National Park, South Africa.

[230] Tolson BA, Maier HR, Simpson AR & Lence BJ, 2004, *Genetic algorithms for reliability based optimisation of water distribution systems*, Journal of Water Resources Planning and Management, **130(1)**, pp. 63–72.

[231] Toscano-Pulido G, Santana-Quintero LV & Coello Coello CA, 2007, *EMOPSO: A multi-objective particle swarm optimizer with emphasis on efficiency*, (Unpublished) manuscript, CINVESTAV-IPN (Evolutionary Computation Group), University of San Nicolas de los Garza, Mexico City.

[232] Tseng L-Y & Chen C, 2007, *Multiple trajectory search for multi-objective optimization*, Proceedings of the IEEE Congress on Evolutionary Computation (CEC2007), Singapore, IEEE, pp. 3609–3616.

[233] United Nations, 2008, *Millenium Development Goals*, [Online], [Cited December 20th 2007], Available at `http://www.un.org`

[234] United Nations World Water Assessment Programme, 2003, *The 1st UN world water development report: Water for people, water for life*, UNESCO (United Nations Educational, Scientific and Cultural Organization) and Berghahn Books, Paris.

[235] United Nations World Water Assessment Programme, 2006, *The 2nd UN world water development report: Water, a shared responsibility*, UNESCO (United Nations Educational, Scientific and Cultural Organization) and Berghahn Books, Paris.

[236] Vairavamoorthy K & Shen Y, 2004, *Least cost design of water distribution network using particle swarm optimisation*, Phoon K-K & Liong S-Y (eds.), Sixth International Conference on Hydroinformatics, World Scientific Publishing Company, 1, pp. 834–841.

[237] Vamvakeridou-Lyroudia LS, Walters GA & Savic DA, 2004, *Fuzzy multiobjective design optimisation of water distribution networks*, (Unpublished) Report No. 2004/01, Centre for Water Systems, School of Engineering, Computer Science and Mathematics, University of Exeter, Exeter.

[238] Vamvakeridou-Lyroudia LS, Savic DA & Walters GA, 2007, *Tank simulation for the optimisation of water distribution networks*, Journal of Hydraulic Engineering, **133(6)**, pp. 625–636.

[239] Van Zyl JE, 2004, *The effect of pressure on leaks in water distribution systems*, Proceedings of the 2004 Water Institute of Southern Africa (WISA) Biennial Conference, pp. 1104–1109.

[240] Van Zyl JE, 2007, *OOTEN — Object-oriented Toolkit for EPANET* [Online], [Cited October 7th 2009], Available from `http://epanet.de/en/ooten/index.html`

[241] Van Zyl JE, Savic DA & Walters GA, 2006, *Explicit integration method for extended-period simulation of water distribution systems*, Journal of Hydraulic Engineering, **132(4)**, pp. 385–392.

[242] van Zyl HJ, Ilemobade AA & van Zyl JE, 2008, *An improved area-based guideline for domestic water demand estimation in South Africa*, Water SA, **34(3)**, pp. 381–392.

[243] Vasan A & Simonovic SP, 2010, *Optimisation of water distribution network design using differential evolution*, Journal of Water Resources Planning and Management, **136(2)**, pp. 279–287.

[244] Vrugt JA & Robinson BA, 2007, *Improved evolutionary optimisation from genetically adaptive multimethod search*, Proceedings of the National Academy of Sciences, **104(3)**, pp. 708–711.

[245] Wagner M, Auger A & Schoenauer M, 2004, *EEDA: A new robust estimation of distribution algorithm*, Technical Report No. 5190, Institut National de Recherche en Informatique et en Automatique (INRIA), Rocquencourt, France.

[246] Walski TM, Brill ED, Gessler J, Goulter IC, Jeppson RM, Lansey K, Lee H, Liebman JC, Mays LW, Morgan DR & Ormsbee LE, 1987, *Battle of the network models: Epilogue*, Journal of Water Resources Planning and Mangement, **113(2)**, pp. 191–203.

[247] Walski TM, Sjostrom JW & Gessler J, 1990, *Water distribution systems: Simulation and sizing*, Lewis Publishers, Inc., Chelsea (MI).

[248] Walski TM, Youshock M & Rhee H, 2000, *Use of modeling in decision making for water distribution master planning*, Proceedings of the 2000 ASCE EWRI Conference, Minneapolis (MN).

[249] Walski TM, 2000, *Hydraulic design of water distribution storage tanks*, in Mays L (Ed), 2000, *Water distribution systems handbook*, McGraw-Hill, New York (NY).

[250] Walski TM, 2001, *The wrong paradigm — Why water distribution optimisation doesn't work*, Journal of Water Resources Planning and Management, **127(2)**, pp. 203–205.

[251] Walski TM (Ed), 2003, *Advanced water distribution modeling and management*, Haestad Press, Waterbury (CT).

[252] Walters GA & Cembrowicz RG, 1993, *Optimal design of water distribution networks*, in Cabrera E & Martinez F, (Eds), *Water supply systems: State-of-the-art and future trends*, Computational Mechanics Inc., Southampton, pp. 91–117.

[253] Walters GA & Lohbeck T, 1993, *Optimal layout of tree networks using genetic algorithms*, Engineering Optimisation, **22(1)**, pp. 27–48.

[254] Walters GA, Halhal D, Savic DA & Ouzar D, 1999, *Improved design of Anytown distribution network using structured messy genetic algorithms*, Urban Water, **1(1)**, pp. 23–38.

[255] Walters GA & Smith DK, 1995, *Evolutionary design algorithm for optimal layout of tree networks*, Engineering Optimisation, **24(4)**, pp. 261–281.

[256] Water for People, 2007, *Water for people: Helping people improve their quality of life*, [Online], [Cited October 27th, 2007], Available from `http://www.waterforpeople.org`

[257] White FM, 1994, *Fluid mechanics (3rd Edition)*, McGraw-Hill, New York (NY), pp. 307–320.

[258] World Health Organization / UNICEF, 2010, *Progress on Sanitation and Drinking Water: 2010 Update*, WHO/UNICEF Joint Monitoring Programme for Water Supply and Sanitation, WHO Press, Geneva.

[259] Winston WL, 2004, *Operations research: Applications and algorithms (4th Edition)*, Thomson-Brooks/Cole, Belmont (CA).

[260] Wolpert DH and Macready WG, 1997, *No free lunch theorems for optimisation*, IEEE Transactions on Evolutionary Computation, **1(1)**, pp. 67–82.

[261] Wood D, 1980, *User's Manual — Computer analysis of flow in pipe networks including extended period simulations*, (Unpublished) Technical Report, Department of Civil Engineering, University of Kentucky, Lexington (KY).

[262] Wu ZY & Simpson AR, 2001, *Competent genetic-evolutionary optimisation of water distribution systems*, Journal of Computing in Civil Engineering, **15(2)**, pp. 90–101.

[263] Wu ZY & Simpson AR, 2002, *A self-adaptive boundary search genetic algorithm and its application to water distribution systems*, Journal Hydraulic Research, **40**, pp. 191–203.

[264] Wu ZY, Boulos PF, Orr CH & Ro JJ, 2001, *Using genetic algorithms to rehabilitate distribution systems*, Journal of the American Water Works Association, **93(11)**, pp. 74–85.

[265] Wu ZY, Walski TM, Mankowski R, Tryby H, Herrin G & Hartell W, 2002, *Optimal capacity of water distribution systems*, Proceedings of the 1st Annual Environmental and Water Resources Systems Analysis Symposium, Roanoke (VA).

[266] Wu ZY & Walski T, 2004, *Self-adaptive penalty cost for optimal design of water distribution systems*, Critical Transitions in Water and Environmental Resources Management, American Society of Civil Engineers.

[267] Xu C & Goulter IC, 1998, *Probabilistic model for water distribution reliability*, Journal of Water Resources Planning and Management, **124(4)**, pp. 218–228.

[268] Xu C & Goulter IC, 1999, *Reliability-based optimal design of water distribution networks*, Journal of Water Resources Planning and Management, **125(6)**, pp. 352–362.

[269] Yang S, Hsu N-S, Louie PWF & Yeh WW-G, 1996, *Water distribution network reliability: Connectivity analysis*, Journal of Infrastructure Systems, **2(2)**, pp. 54–64.

[270] Yen BC, Cheng ST & Melching CS, 1986, *First-order reliability analysis*, in Yen BC (Ed), *Stochastic and risk analysis in hydraulic engineering*, Water Resources Publications, Littleton (CO), pp. 1–36.

[271] Yen GG & Lu H, 2003, *Dynamic multiobjective evolutionary algorithm: Adaptive cell-based rank and density estimation*, IEEE Transactions on Evolutionary Computation, **7(3)**, pp. 253–274.

[272] Yuan X, Zhang Y & Yuan Y, 2008, *Improved Self-Adaptive Chaotic Genetic Algorithm for Hydrogeneration Scheduling*, Journal of Water Resources Planning and Management, **134(4)**, pp. 319–325.

[273] Zhang X, Lu B, Gou S & Jiao L, 2006, *Immune multiobjective optimization algorithm for unsupervised feature selection*, Applications of Evolutionary Computing (EvoWorkshops 2006), Budapest, Hungary, Lecture Notes in Computer Science 3907, pp. 484–494.

[274] Zecchin AC, Simpson AR, Maier HR & Nixon JB, 2005, *Parametric study for an ant algorithm applied to water distribution system optimisation*, IEEE Transactions in Evolutionary Computation, **9(2)**, pp. 175–191.

[275] Zecchin AC, Maier HR, Simpson AR, Leonard M & Nixon JB, 2007, *Ant colony optimisation applied to water distribution system design: Comparative study of five algorithms*, Journal of Water Resources Planning and Management, **133(1)**, pp. 87–92.

[276] Zitzler E, Deb K & Thiele L, 2000, *Comparison of multiobjective evolutionary algorithms: Empirical results*, Evolutionary Computation **8(2)**, pp. 173–195.

[277] Zitzler E, Laumans M & Thiele L, 2002, *SPEA2: Improving the Strength Pareto Evolutionary Algorithm for multiobjective optimisation*, in Giannakoglou K, Tsahalis D, Periaux J, Papailiou K, Fogarty T (Eds), *Evolutionary methods for design, optimisation and control*, CIMNE, Barcelona.

[278] Zitzler E, Laumanns M & Bleuler S, 2003, *A tutorial on evolutionary multiobjective optimisation*, (Unpublished) Technical Report, Computer Engineering and Network Laboratory (TIK), Swiss Federal Institute of Technology (ETH), Zurich.

[279] Zitzler E & Thiele L, 1999, *Multiobjective evolutionary algorithms: A comparative case study and the Strength Pareto Evolutionary Algorithm*, IEEE Transactions on Evolutionary Computation, **3(4)**, pp. 257–271.

Appendix A

Hydraulic Theory

This section contains a brief review of basic fluid mechanics definitions and theory with examples and hydraulic equation derivations. It may be considered a supplement to Chapter 2.

A.1 Density

The *mass density* of a fluid is its mass per unit volume, measured in SI units of kilograms per cubic meter (kg/m^3). The Greek symbol ρ is used to denote density. The density of water at 4^oC is 1000 kg/m^3 [170, 42].

A.2 Specific Weight

The *specific weight* of a fluid is its weight per unit volume, measured in SI units of Newtons per cubic meter (N/m^3). Specific weight is denoted by the Greek symbol γ. The specific weight of water at 4^oC is 9810 N/m^3 [170, 42]. The relationship between density and specific weight is $\gamma = \rho g$, where $g \approx 9.81$ denotes the gravitational acceleration constant of the earth near its surface.

A.3 Density variations

Fluids for which density changes occur as a result of pressure are said to be *compressible*. *Incompressible* fluids should therefore have a constant density, although density changes may still occur as a result of chemical impurities (such as salt in water) and temperature changes. Many fluids (such as water) are treated as incompressible under normal operating conditions. A fluid for which density is not constant in its flow field is said to be *nonhomogeneous* [170, 42].

A.4 Specific Gravity

The *specific gravity* of a fluid is the ratio of its specific weight to the specific weight of water at a standard reference temperature of 4^o C [170, 42]. The specific gravity, S, of a fluid is therefore

the ratio

$$S = \frac{\gamma_{\text{fluid}}}{\gamma_{\text{water}}} = \frac{\rho_{\text{fluid}}}{\rho_{\text{water}}}.$$

A.5 Viscosity

In the flow of fluids, *shear stress* (the force tangential and opposite to the flow direction) is present, causing fluid friction. *Viscosity* is a measure of a fluid's resistance to shear or angular deformation. In a viscous fluid, shear stress τ is proportional to the velocity gradient, $\mathrm{d}v/\mathrm{d}y$, also known as the *time rate of strain* [170, 42].

For example, a fluid flowing next to a wall undergoes a friction force which manifests itself less as the distance from the wall increases (as shown in Figure A.1). The time rate of strain is $\mathrm{d}v/\mathrm{d}y$, where v is the horizontal velocity of the fluid and y is the distance from the wall. The concept of shear stress applies equally between two thin sheets of fluid. Shear stress is expressed as

$$\tau = \mu \frac{\mathrm{d}v}{\mathrm{d}y}, \tag{A.1}$$

where the proportionality factor μ is called the *dynamic viscosity* of the fluid. Shear stress has units of $\mathrm{N/m^2}$, and dynamic viscosity has units of $\mathrm{N.s/m^2}$. The SI units of dynamic viscosity is centipose (cP), where 1 cP = 1 $\mathrm{N.s/m^2} \times 10^{-3}$.

The flow profile in Figure A.1 is typical of that for a *laminar* (*non-turbulent*) flow close to a solid boundary. The velocity gradient at the boundary is finite. The curve of velocity variation cannot be tangent to the boundary since this would imply an infinite velocity gradient and, in turn, an infinite shear stress, which is impossible. A velocity gradient, which becomes less steep ($\mathrm{d}v/\mathrm{d}y$ becomes smaller) with distance from the boundary, has a maximum sheer stress at the boundary, and the shear stress decreases with distance from the boundary. Note that the velocity of the fluid at the boundary is zero (the fluid assumes the velocity of the boundary and no slip occurs). The viscosity of water at 20^oC is 10^{-3} $\mathrm{N.s/m^2}$.

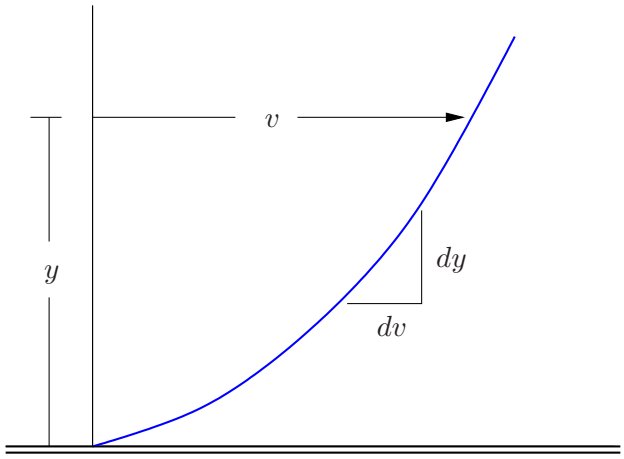


Figure A.1: *Velocity distribution next to a boundary.*

The *kinematic viscosity*, ν, of a fluid is the ratio of its dynamic viscosity to its density. Therefore $\nu = \mu/\rho$, which has SI units of m^2/s. The notion of kinematic viscosity has been defined because many equations in hydraulics include the ratio μ/ρ.

When a shear stress is applied to a fluid, motion occurs. Solids can resist shear stress in a static condition, but fluids deform continuously under shear stress. The viscous resistance of

fluids is independent of the normal force (pressure) acting within the fluid. For liquids, shear stress is involved with cohesive forces between molecules. These forces decrease with increasing temperature which results in a decrease in viscosity. An equation for the variation of liquid viscosity with temperature is

$$\mu = ce^{bT}, \tag{A.2}$$

where c and b are empirical constants which require viscosity data at two known temperatures for evaluation. Equation (A.2) should therefore be used for interpolation purposes [42].

Ideal fluids are defined as ones in which viscosity is zero. There are no such fluids [170, 42].

A.6 Elasticity

The *elasticity* (or *compressibility*) of a fluid is related to the amount of deformation (expansion or contraction) for a given pressure change. Elasticity is characterised by the *bulk modulus of elasticity*, E_v, which is defined as the ratio of relative change in volume, $\mathrm{d}V/V$, due to a differential change in pressure, $\mathrm{d}p$, expressed as

$$E_v = -\frac{\mathrm{d}p}{\mathrm{d}V/V}.$$

Also, it is true that $\mathrm{d}V/V = \mathrm{d}\rho/\rho$, so that $E_v = -\frac{\mathrm{d}p}{\mathrm{d}\rho/\rho}$. The bulk modulus of elasticity of water is approximately 2.2 GN/m^2, which corresponds to a 0.05% change in volume for a 1 MN/m^2 change in pressure. This justifies its treatment as an incompressible fluid [170, 42].

A.7 Surface Tension

The molecules of water below the surface exert forces on each other which are equal in all directions. Molecules near the surface have greater attraction for each other. These molecules are not able to bond in all directions and consequently form stronger bonds with adjacent molecules. The surface water therefore acts like a stretched membrane seeking a minimum possible area by exerting a tension on the adjacent portion of the surface or an object in contact with the water surface. This surface tension acts in the plane of the surface for capillary action and is a function of temperature. Surface tension may usually be ignored in macro-scale applications, such as the subject matter of this dissertation [42, 93, 170].

A.8 Velocity and Flow Visualization

The two ways to visualise the motion of fluids are the *Lagrangian* and *Eulerian* viewpoints. The Lagrangian viewpoint considers the motion over time of individual particles. It is therefore necessary to consider the motion of all the particles simultaneously to model macroscopic fluid behaviour. If $\mathbf{F}$ is the resultant force acting on a particle of mass m and $\mathbf{a}$ is its resulting acceleration, then the motion in the flow field is obtained by solving the basic equation of motion, $\mathbf{F} = m\mathbf{a}$ (Newton's second law), for every fluid particle in the field of flow [42].

It is common in fluid mechanics to consider fluids as a continuum. The *Eulerian* viewpoint focuses on a particular point or control volume in space, and considers the motion of fluid that

passes through it as a function of time. The velocity of the particles passing through a point depends on the location of the point in space and time. This may be expressed as

$$\mathbf{v} = f_1(x, y, z, t)\boldsymbol{i} + f_2(x, y, z, t)\boldsymbol{j} + f_3(x, y, z, t)\boldsymbol{k}. \tag{A.3}$$

In order to describe the flow field, the Eulerian approach requires that one knows the fluid motion at all points in the field. Owing to the difficulty of tracking the motion of individual particles, the Eulerian approach is used in the vast majority of fluid dynamic analyses.

Streamlines are lines drawn in the flow field so that the velocity vectors of the fluid at any point on a streamline are tangent to the streamline at any instant in time (see Figure A.2). The total velocity can be expressed as a function of distance along a streamline, s, and time t,

$$\mathbf{v} = \mathbf{v}(s, t). \tag{A.4}$$

Streamlines together form a *flow pattern* which clearly shows the geometry of the flow field. It

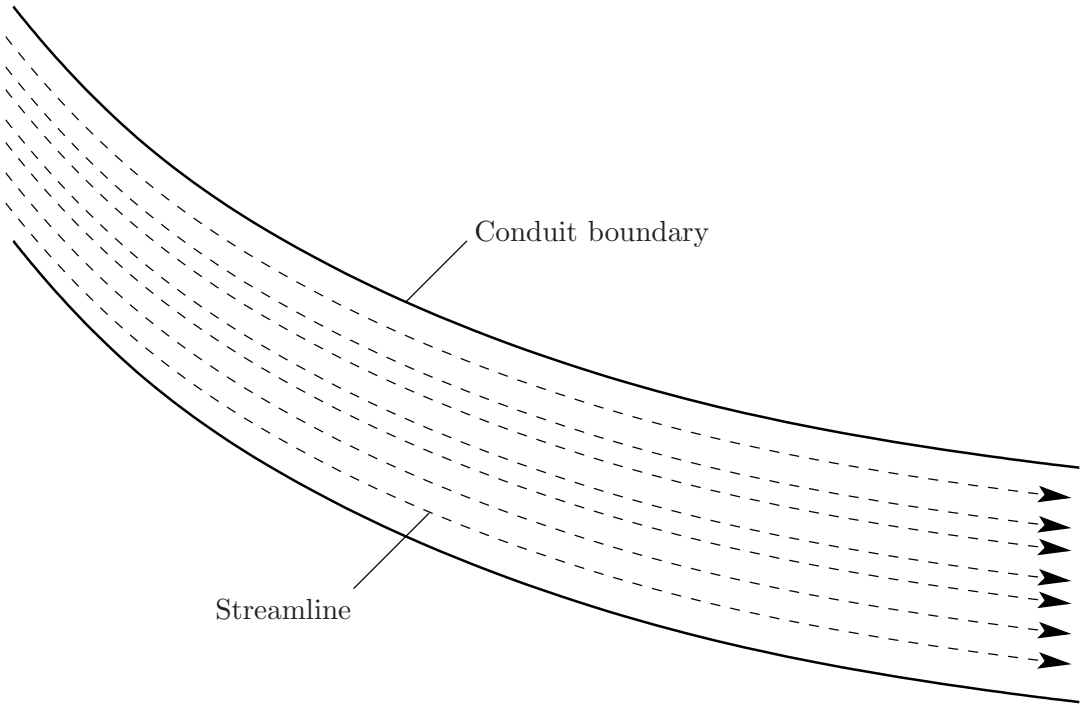


Figure A.2*: Streamline representation in fluid flow.*

should be noted that streamlines are an instantaneous representation of a flow field. A *pathline* is the true path followed by a fluid particle.

Uniform flow is fluid motion characterised by a constant velocity which does not change from point to point along any of the streamlines ($\frac{\partial \mathbf{V}}{\partial \mathbf{S}} = 0$), which are both straight and parallel. The variation in velocity with respect to time at a given point in a flow field is used to classify flow. *Steady flow* occurs when the velocity does not vary in magnitude or direction with respect to time ($\frac{\partial \mathbf{V}}{\partial t} = 0$). This means that the overall flow pattern does not change with time, and therefore that the overall mass in a control volume is constant over time. Pathlines coincide with streamlines if flow is *steady* The counterparts of uniform flow and steady flow are *nonuniform flow* ($\frac{\partial \mathbf{V}}{\partial \mathbf{S}} \neq 0$) and *unsteady flow* ($\frac{\partial \mathbf{v}}{\partial t} \neq 0$) respectively. A flow pattern for unsteady flow can only be regarded as an instantaneous representation of the flow geometry [170, 42].

A.9 Laminar and Turbulent Flow

Turbulent flow is irregular with no definite flow patterns. It is caused by eddies of varying size within the flow that create a mixing action. The fluid particles follow irregular and erratic paths, with no two particles having similar motion. The index relating to turbulence is the *Reynolds number*,

$$R_{\mathrm{e}} = \frac{vD\rho}{\mu} = \frac{vD}{\nu},$$

where D is a characteristic length such as the internal diameter of a pipe. If the average velocity with respect to time (a *temporal mean* taken over a relatively long period of time) is constant at a given point, turbulent flow may be simplified to a steady flow process in analysis. Turbulent flow can sometimes be caused by increasing the flow velocity, or by boundary surfaces causing abrupt changes in velocity.

Laminar flow does not exhibit eddies which cause intense mixing. Flow is therefore very smooth and may be represented by means of streamlines. The fluid particles move in definite paths and the fluid appears to move by the sliding of laminations of infinitesimal thickness relative to the adjacent fluid layers. The viscous shear of the fluid particles produces resistence to the flow. Resistence to flow varies with the first power of the velocity.

Flow may be *one-*, *two-*, or *three-dimensional*, for which one, two or three coordinate directions respectively, are required to describe velocity and property changes in a flow field. An example of one-dimensional flow could be average flow velocity in a conduit (a simplified analysis technique).

Pipe flow is generally turbulent for $R_{\mathrm{e}} > 2000$ and laminar for $R_{\mathrm{e}} < 2000$. This is also dependent on the initial conditions of the fluid (before it enters the pipe), and whether the pipe undergoes vibration or not. In pipe flow, it is common to use $D = 4R$, where R is the *hydraulic radius*, defined as the cross-sectional area A divided by the wetted perimeter P. Therefore for full pipe flow $R = A/P = (\pi D^2/4)/\pi D = D/4$ [42, 170].

A.10 Control Volume Approach

Since it is difficult to identify individual particles, problems in fluid mechanics are solved by focusing on a given space through which fluids flow (the Eulerian viewpoint), rather than solely considering a given mass of fluid. The actual method is called the *Reynolds transport theorem* or *control-volume approach*.

A *fluid system* is defined as a given mass of fluid which is continuous and always contains the same fluid particles. The system has a *system boundary* or *control surface* (as illustrated at time t by the dashed line in Figure A.3). Mass does not cross the system boundary.

A *control volume* is a selected volumetric region in space, surrounded by a control surface, CS. This surface may coincide with physical boundaries, such as the wall of a pipe. A portion of a control surface may be a hypothetical surface through which the fluid flows. Therefore the fluid mass within the control volume may change with time and can cross the control volume boundary. A control volume may deform with time as well as move and rotate in space.

Extensive and *intensive* properties are used in the control volume approach to apply physical properties for discrete masses to a fluid flowing continuously through a control volume. Extensive properties are related to the total mass of the system, whereas intensive properties are independent of the mass of fluid. Extensive properties include mass m, momentum mv, and

energy E. Intensive properties include unit mass, momentum per unit mass (velocity v), and energy per unit mass (represented by e). Both intensive and extensive properties may be scalar or vector properties.

The relationship between extensive (B) and intensive (b) properties for a given system S is defined by

$$B = \int_S b \, \mathrm{d}m = \int_S b\rho \, \mathrm{d}V, \tag{A.5}$$

where $\mathrm{d}m$ and $\mathrm{d}V$ denote differential mass and differential volume respectively.

For the control volume indicated by the dotted line in Figure A.3 the net flowrate (or net outflow rate) is

$$\begin{aligned} \dot{Q} &= Q_{\text{out}} - Q_{\text{in}} \\ &= \mathbf{v}_2 \cdot \mathbf{A}_2 - \mathbf{v}_1 \cdot \mathbf{A}_1 \\ &= \sum_{\text{CS}} \mathbf{v} \cdot \mathbf{A}. \end{aligned}$$

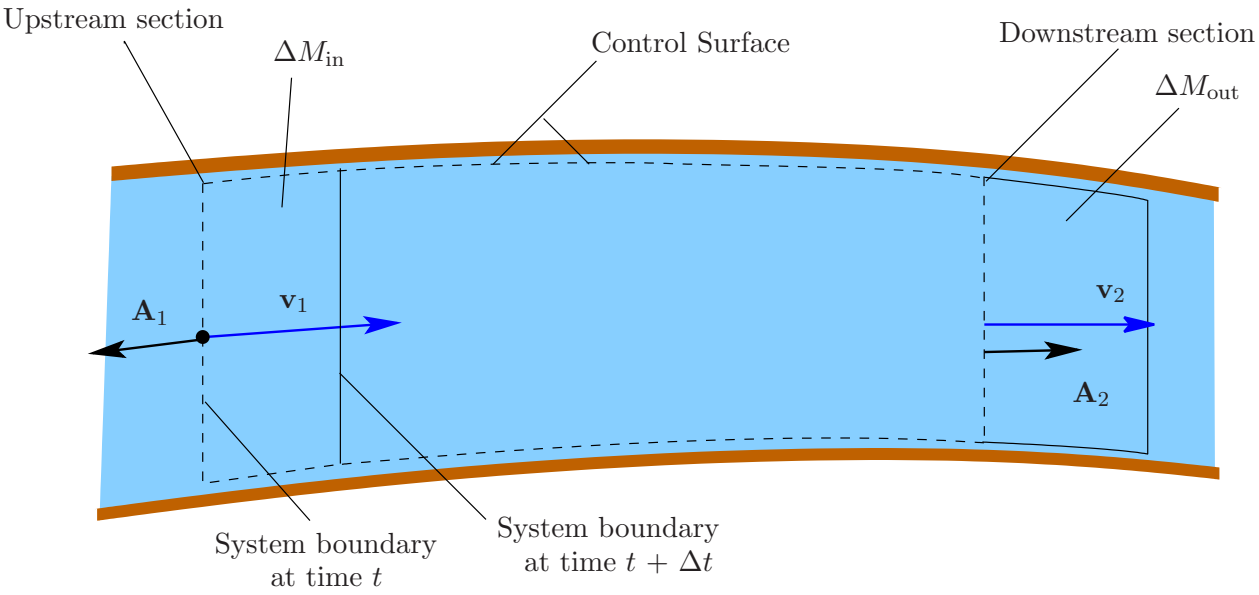


Figure A.3: *Control-volume in pipe flow.*

In other words, the dot product, $\mathbf{v} \cdot \mathbf{A}$, for all flows in and out of a control volume is the net rate of outflow. If the flow is steady in a control volume, $\dot{Q} = 0$, and

$$\mathbf{v}_2 \cdot \mathbf{A}_2 = \mathbf{v}_1 \cdot \mathbf{A}_1.$$

This is known as the one-dimensional *continuity equation* for incompressible fluids undergoing steady flow. The mass rate of flow out of the control volume is

$$\frac{\mathrm{d}m}{\mathrm{d}t} = \dot{m} = \sum_{\text{CS}} \rho \mathbf{v} \cdot \mathbf{A}.$$

Therefore the rate of flow of extensive property B is the product of the mass rate and the intensive property,

$$\frac{\mathrm{d}B}{\mathrm{d}t} = \dot{B} = \sum_{\text{CS}} b\rho \mathbf{v} \cdot \mathbf{A}.$$

If the velocity varies across the flow section, then one must integrate across the section, so that the previous equation becomes

$$\dot{B} = \int_{\mathrm{CS}} b\rho \mathbf{v} \cdot \mathrm{d}\mathbf{A}. \tag{A.6}$$

The basic equation for the control-volume approach is derived by considering the rate of change of an extensive property B of the system of fluid that is flowing through a control volume. Consider Figure A.3 again. The dashed lines form the control surface of a system at time t. The solid lines form the control surface of the same system at time $t + \Delta t$ (the system has flowed along the pipe). Note that the vertical lines divide the flow space into three regions, the first and last of which have been labeled ΔM_{in} and ΔM_{out} respectively. Let the dashed lines represent a control volume CV. The rate of change of extensive property B of the system, $\mathrm{d}B_{\mathrm{sys}}/\mathrm{d}t$, may be stated as

$$\frac{\mathrm{d}B_{\mathrm{sys}}}{\mathrm{d}t} = \lim_{\Delta t \to 0} \left[\frac{B_{t+\Delta t} - B_t}{\Delta t} \right].$$

The mass of the system at time $t+\Delta t$ is equal to the mass of the fluid within the control volume at time $t + \Delta t$ ($M_{\mathrm{CV,t+\Delta t}}$), together with the mass of fluid that has moved out of the control volume during time Δt (ΔM_{out}), less the mass of fluid that has moved into the control volume during time Δt (ΔM_{in}). Let ΔB_{out} and ΔB_{in} satisfy a similar definition, except pertaining to extensive property B of the system. Thus, the property B of the system at time $t+\Delta t$ may be written as $B_{\mathrm{CV,t+\Delta t}} + \Delta B_{\mathrm{out}} - \Delta B_{\mathrm{in}}$, and the rate of change of B may hence be expressed as

$$\begin{aligned} \frac{\mathrm{d}B_{\mathrm{sys}}}{\mathrm{d}t} &= \lim_{\Delta t \to 0} \left[\frac{(B_{\mathrm{CV,t+\Delta t}} + \Delta B_{\mathrm{out}} - \Delta B_{\mathrm{in}}) - (B_{\mathrm{CV,t}})}{\Delta t} \right] \\ &= \lim_{\Delta t \to 0} \left[\frac{B_{\mathrm{CV,t+\Delta t}} - B_{\mathrm{CV,t}}}{\Delta t} \right] + \lim_{\Delta t \to 0} \left[\frac{\Delta B_{\mathrm{out}} - \Delta B_{\mathrm{in}}}{\Delta t} \right] \\ &= \frac{\mathrm{d}B_{\mathrm{CV}}}{\mathrm{d}t} + \dot{B}_{\mathrm{CV}}. \end{aligned} \tag{A.7}$$

The first term on the right-hand side of (A.7) is the rate of change with respect to time of extensive property B of the fluid inside the control volume, and the second term is the net rate of flow of extensive property B from the control volume. Substituting (A.5) and (A.6) into (A.7) yields

$$\frac{\mathrm{d}B_{\mathrm{sys}}}{\mathrm{d}t} = \frac{\mathrm{d}}{\mathrm{d}t} \int_{\mathrm{CV}} b\rho \, \mathrm{d}V + \int_{\mathrm{CS}} b\rho \mathbf{v} \cdot \mathrm{d}\mathbf{A}. \tag{A.8}$$

This is known as the *general control volume equation* (also referred to as *Reynolds transport theorem*). This equation is applied to develop continuity, energy and momentum equations for hydraulic systems [42, 170].

A.11 Continuity

The extensive property in the continuity equation is mass ($B = m$) and the intensive property is $b = \mathrm{d}B/\mathrm{d}m = 1$. The mass of the system is constant owing to the law of conservation of mass, therefore $\mathrm{d}B/\mathrm{d}t = \mathrm{d}m/\mathrm{d}t = 0$. From (A.8), the general form of the continuity equation is then

$$0 = \frac{\mathrm{d}}{\mathrm{d}t} \int_{\mathrm{CV}} \rho \, \mathrm{d}V + \int_{\mathrm{CS}} \rho \mathbf{v} \cdot \mathrm{d}\mathbf{A}, \tag{A.9}$$

also called the *integral equation of continuity for unsteady, variable-density flow.* Equation (A.9) may be rewritten as

$$\int_{\mathrm{CS}} \rho \mathbf{v} \cdot \mathrm{d}\mathbf{A} = -\frac{\mathrm{d}}{\mathrm{d}t} \int_{\mathrm{CV}} \rho \, \mathrm{d}V, \tag{A.10}$$

which states that the net rate of outflow of mass from the control volume is equal to the rate of decrease of mass within the control volume. If the density is constant, (A.10) may be divided on both sides by ρ. The continuity equation for flow with a uniform velocity across the flow section and constant density then becomes

$$\sum_{\mathrm{CS}} \mathbf{v} \cdot \mathbf{A} = -\frac{\mathrm{d}}{\mathrm{d}t} \int_{\mathrm{CV}} \mathrm{d}V.$$

For constant density, steady one-dimensional flow, such as water flowing in a conduit, the equation becomes

$$\sum_{\mathrm{CS}} vA = 0.$$

Considering a control volume between locations 1 and 2 in a pipe conduit, the continuity equation then delivers

$$Q_1 = v_1 A_1 = v_2 A_2 = Q_2.$$

For constant-density, unsteady flow, consider that $\int_{\mathrm{CV}} \mathrm{d}V$ is the volume of fluid stored in a control volume denoted by S. This means that

$$\frac{\mathrm{d}}{\mathrm{d}t} \int_{\mathrm{CV}} \mathrm{d}V = \frac{\mathrm{d}S}{\mathrm{d}t}. \tag{A.11}$$

The net outflow is defined as

$$\int_{\mathrm{CS}} \mathbf{v} \cdot \mathrm{d}\mathbf{A} = \int_{\mathrm{outlet}} \mathbf{v} \cdot \mathrm{d}\mathbf{A} + \int_{\mathrm{inlet}} \mathbf{v} \cdot \mathrm{d}\mathbf{A} = Q(t) - I(t). \tag{A.12}$$

Equations (A.11) and (A.12) may be substituted into (A.10) to arrive at

$$\frac{\mathrm{d}S}{\mathrm{d}t} = I(t) - Q(t), \tag{A.13}$$

an equation frequently used in hydraulic analyses [42].

A.12 Energy

The control volume approach may be combined with the first law of thermodynamics to develop the energy equation for fluid flow in hydro processes. This energy balance must form an accounting of the energy inputs and outputs to and from a system [170, 42, 93]. The first law of thermodynamics states that the rate of change of energy with time is the rate at which heat is transferred into the fluid, $\mathrm{d}H/\mathrm{d}t$, less the rate at which the fluid performs work on its surroundings, $\mathrm{d}W/\mathrm{d}t$, expressed as

$$\frac{\mathrm{d}E}{\mathrm{d}t} = \frac{\mathrm{d}H}{\mathrm{d}t} - \frac{\mathrm{d}W}{\mathrm{d}t}.$$

The total energy of a fluid system is the sum of the internal energy E_{u}, the kinetic energy E_{k}, and the potential energy E_{p}. The extensive property of the system is then $B = E = E_{\mathrm{u}} + E_{\mathrm{k}} + E_{\mathrm{p}}$, and the intensive property $b = \frac{dB}{dm} = e = e_{\mathrm{u}} + e_{\mathrm{k}} + e_{\mathrm{p}}$, where e represents the energy per unit mass. Substituting B and b into a one-dimensional version of the control volume equation (A.8) yields

$$\begin{aligned} \frac{\mathrm{d}E}{\mathrm{d}t} = \frac{\mathrm{d}H}{\mathrm{d}t} - \frac{\mathrm{d}W}{\mathrm{d}t} &= \frac{\mathrm{d}}{\mathrm{d}t} \int e\rho \, \mathrm{d}V + \sum_{\mathrm{CS}} e\rho \mathbf{v} \cdot \mathrm{d}\mathbf{A} \\ &= \frac{\mathrm{d}}{\mathrm{d}t} \int (e_{\mathrm{u}} + e_{\mathrm{k}} + e_{\mathrm{p}})\rho \, \mathrm{d}V + \sum_{\mathrm{CS}} (e_{\mathrm{u}} + e_{\mathrm{k}} + e_{\mathrm{p}})\rho \mathbf{v} \cdot \mathrm{d}\mathbf{A}. \end{aligned}$$

The kinetic energy per unit mass is the total kinetic energy of the mass with velocity v divided by the mass m,

$$e_{\mathrm{k}} = \frac{mv^2/2}{m} = \frac{v^2}{2}. \tag{A.14}$$

The potential energy per unit mass is the weight of the fluid, γV, multiplied by the centroid elevation z divided by the mass m, producing

$$e_{\mathrm{p}} = \frac{\gamma V z}{m} = \frac{\gamma V z}{\rho V} = gz. \tag{A.15}$$

The *general energy equation for unsteady variable-density flow* may therefore be written as

$$\frac{\mathrm{d}H}{\mathrm{d}t} - \frac{\mathrm{d}W}{\mathrm{d}t} = \frac{\mathrm{d}}{\mathrm{d}t}\int (e_{\mathrm{u}} + \frac{v^2}{2} + gz)\rho \,\mathrm{d}V + \sum_{\mathrm{CS}} (e_{\mathrm{u}} + \frac{v^2}{2} + gz)\rho \mathbf{v}\cdot \mathrm{d}\mathbf{A}. \tag{A.16}$$

For steady flow, this reduces to

$$\frac{\mathrm{d}H}{\mathrm{d}t} - \frac{\mathrm{d}W}{\mathrm{d}t} = \sum_{\mathrm{CS}} (e_{\mathrm{u}} + \frac{v^2}{2} + gz)\rho \mathbf{v}\cdot \mathrm{d}\mathbf{A}. \tag{A.17}$$

The work done by a system on its surroundings may be divided into shaft work, W_{s}, and flow work, W_{f}. Flow work is the result of the pressure force as the system moves through space, and shaft work is any other work. Considering Figure A.3 again, let the force at the upstream end of the control volume be $p_1 A_1$, and the distance travelled over time Δt be $l_1 = v_1 \Delta t$. Recalling that work is the product of force and the distance over which the force acts, one arrives at the conclusion that the work done on the upstream end is

$$W_{f_1} = v_1 p_1 A_1 \Delta t,$$

and the downstream end

$$W_{f_2} = -v_2 p_2 A_2 \Delta t.$$

The minus sign for the upstream work is because the pressure on the surrounding fluid acts in the opposite direction to the motion of the system boundary. The rate of work at the upstream and downstream ends respectively is $\frac{\mathrm{d}W_{f_1}}{\mathrm{d}t} = v_1 p_1 A_1$ and $\frac{\mathrm{d}W_{f_2}}{\mathrm{d}t} = -v_2 p_2 A_2$.

This demonstrates that the rate of flow work may be expressed generally as

$$\frac{\mathrm{d}W_{\mathrm{f}}}{\mathrm{d}t} = \sum_{\mathrm{CS}} p\mathbf{v}\cdot\mathbf{A} = \sum_{\mathrm{CS}} \frac{p}{\rho}\rho\mathbf{v}\cdot\mathbf{A},$$

which allows the net rate of work of the system to be written as

$$\frac{\mathrm{d}W}{\mathrm{d}t} = \frac{\mathrm{d}W_{\mathrm{s}}}{\mathrm{d}t} + \sum_{\mathrm{CS}} \frac{p}{\rho}\rho\mathbf{v}\cdot\mathbf{A}. \tag{A.18}$$

Employing (A.18) and (A.16), the general energy equation for unsteady, variable-density flow may therefore be expressed as

$$\frac{\mathrm{d}H}{\mathrm{d}t} - \frac{\mathrm{d}W_{\mathrm{s}}}{\mathrm{d}t} + \sum_{\mathrm{CS}} \frac{p}{\rho}\rho\mathbf{v}\cdot\mathbf{A} = \frac{\mathrm{d}}{\mathrm{d}t}\int (e_{\mathrm{u}} + \frac{v^2}{2} + gz)\rho \,\mathrm{d}V + \sum_{\mathrm{CS}} (e_{\mathrm{u}} + \frac{v^2}{2} + gz)\rho\mathbf{v}\cdot\mathrm{d}\mathbf{A},$$

which may be rearranged to yield

$$\frac{\mathrm{d}H}{\mathrm{d}t} - \frac{\mathrm{d}W_{\mathrm{s}}}{\mathrm{d}t} = \frac{\mathrm{d}}{\mathrm{d}t}\int (e_{\mathrm{u}} + \frac{v^2}{2} + gz)\rho \,\mathrm{d}V + \sum_{\mathrm{CS}} (\frac{p}{\rho} + e_{\mathrm{u}} + \frac{v^2}{2} + gz)\rho\mathbf{v}\cdot\mathrm{d}\mathbf{A}.$$

For steady flow, this reduces to

$$\frac{\mathrm{d}H}{\mathrm{d}t} - \frac{\mathrm{d}W_\mathrm{s}}{\mathrm{d}t} = \sum_{\mathrm{CS}} (\frac{p}{\rho} + e_\mathrm{u} + \frac{v^2}{2} + gz)\rho \mathbf{v} \cdot \mathrm{d}\mathbf{A}. \tag{A.19}$$

A.13 Momentum

The control volume approach may be combined with Newton's second law of motion to derive the *general momentum equation for fluid flow* in hydraulic systems. Newton's second law states that the resultant force, $\sum \mathbf{F}$, on a body of mass m is equal to the rate of change of momentum, $\mathbf{p}$, of the body, that is

$$\sum F = \frac{\mathrm{d}\mathbf{p}}{\mathrm{d}t} = m\mathbf{a},$$

where $\mathbf{a}$ denotes the acceleration of the body. Although Newton's law applies to a single particle, the law may also be formulated for a fluid system. If the extensive property of the system is the momentum of the entire mass forming the system, $B = \mathbf{p} = m\mathbf{v}$, the equation becomes

$$\sum \mathbf{F} = \frac{\mathrm{d}B_\mathrm{sys}}{\mathrm{d}t}.$$

The intensive property is $b = \frac{\mathrm{d}B}{\mathrm{d}m} = \mathbf{v}_e$. This velocity is represented with the symbol $\mathbf{v}_e$ because it is the velocity of a unit mass fluid element referenced with respect to an inertial reference frame (a reference frame that is not itself accelerating).

If B is momentum, then substituting b into the general control volume equation (A.8) yields

$$\frac{\mathrm{d}B_\mathrm{sys}}{\mathrm{d}t} = \sum \mathbf{F} = \frac{\mathrm{d}}{\mathrm{d}t} \int_{\mathrm{CV}} \mathbf{v}_e \rho \, \mathrm{d}V + \int_{\mathrm{CS}} \mathbf{v}_e \rho \mathbf{v} \cdot \mathrm{d}\mathbf{A}, \tag{A.20}$$

known as the *integral momentum equation for fluid flow*. For steady flow, (A.20) reduces to

$$\sum \mathbf{F} = \int_{\mathrm{CS}} \mathbf{v}_e \rho \mathbf{v} \cdot \mathrm{d}\mathbf{A},$$

and for steady flow where velocity across the control surface is uniform, one obtains

$$\sum \mathbf{F} = \sum_{\mathrm{CS}} \mathbf{v}_e \rho \mathbf{v} \cdot \mathrm{d}\mathbf{A}.$$

Under true uniform flow, $\sum_{\mathrm{CS}} \mathbf{v}_e \rho \mathbf{v} \cdot \mathrm{d}\mathbf{A} = 0$ and $\sum \mathbf{F} = 0$ [170, 42, 93].

A.14 Velocity Distribution Correction Factor

Owing to the fact that velocity is not actually uniform over a cross section, an *energy correction factor* is typically introduced in the pipe flow energy equation. Consider the velocity distribution shown in Figure 2.2. The mass of fluid flowing through an area $\mathrm{d}A$ per unit time is $\rho v \, \mathrm{d}A$, where v is the velocity through $\mathrm{d}A$. The flow of kinetic energy per unit time through this area is $\rho v \, \mathrm{d}A(v^2/2) = (\gamma/2g)v^3 \, \mathrm{d}A$. The total kinetic energy flowing through the section per unit time is

$$\frac{\gamma}{2g} \int_A v^3 \, \mathrm{d}A. \tag{A.21}$$

Using the mean velocity $\overline{v}$ and the energy coefficient α, the total energy per unit weight is $\alpha\overline{v}^2/2g$, because the flow across the entire section is $\gamma A\overline{v}$. Therefore the total kinetic energy is

$$(\gamma A\overline{v})\left(\alpha\frac{\overline{v}^2}{2g}\right) = \gamma\alpha A\frac{\overline{v}^3}{2g}. \tag{A.22}$$

From (A.21) and (A.22) one obtains

$$\frac{\gamma}{2g}\int_A v^3\,\mathrm{d}A = \gamma\alpha A\frac{v^3}{2g}.$$

Solving for the correction factor yields

$$\alpha = \frac{1}{A\overline{v}^3}\int_A v^3\,\mathrm{d}A.$$

The new energy equation for pipe flow is

$$\frac{p1}{\gamma} + z_1 + \alpha_1\frac{v_1^2}{2g} + h_\mathrm{p} = \frac{p2}{\gamma} + z_2 + \alpha_2\frac{v_2^2}{2g} + h_\mathrm{t} + h_\mathrm{L}.$$

The value of α for a full parabolic velocity distribution is equal to 2 for laminar flow. However, α normally ranges from 1.03 to 1.06 for turbulent flow. In practice α is typically a value close to 1 and is therefore often ignored (as is the case in this dissertation). [42, 93].

A.15 Moody diagram

The Moody diagram is presented in Figure A.4 showing the Darcy Williams frictional factors for the various combinations of Reynolds number and relative roughness values.

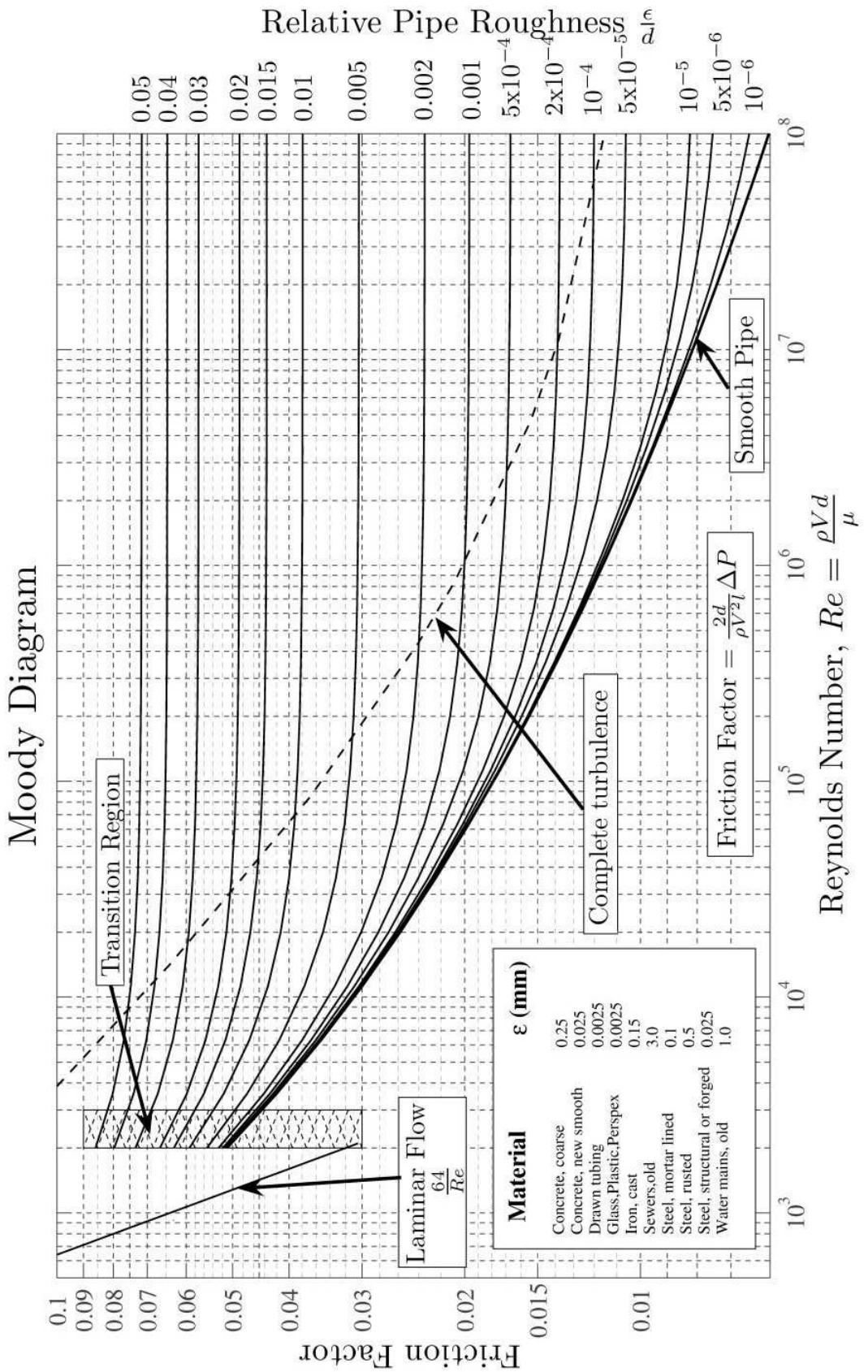


Figure A.4: *Moody diagram for Darcy Williams friction factors derived from R_e and relative roughness (source: Wikimedia Commons, S Beck and R Collins, University of Sheffield).*

Appendix B

Mathematical Supplement

This appendix contains some miscellaneous prerequisite mathematical theory. Topics discussed include Newton's method, regression analysis, numerical integration and the normal and uniform probability distributions.

B.1 Taylor Series Expansion

In mathematics, the Taylor series is a representation of a function as an infinite sum of terms calculated from the values of its derivatives at a single point. It may be regarded as the limit of the Taylor polynomials. The Taylor series of a function $f(x)$ that is infinitely differentiable in the region x_0 is expressed as the following power series:

$$f(x_0) + \frac{f'(x_0)}{1!}(x - x_0) + \frac{f''(x_0)}{2!}(x - x_0)^2 + \frac{f^{(3)}(x_0)}{3!}(x - x_0)^3 + \dots.$$

Since $f(x)$ is often equal to its Taylor expansion evaluated close to x_0, it is considered a very important function. Taylor series need not be convergent in general, but they often are. The limit of a convergent Taylor series of a function f need not in general be equal to the function value $f(x)$, but this is frequently the case. If $f(x)$ is equal to its Taylor series in a neighbourhood of a, it is said to be analytic in this neighborhood. If $f(x)$ is equal to its Taylor series everywhere it is called entire. Taylor series expansion is at a point a is often conducted using only the most significant terms (including the first and often the second derivative), in order to simplify analysis [24].

B.2 Newton's Method

Newtons method is an iterative procedure towards the numerical solution of a root problem, which frequently approaches the exact solution rapidly. Newtons method may be derived by performing a Taylor series expansion and discarding all derivatives higher than the first derivative term. For instance, in solving $f(p) = 0$, with a first order Taylor series expansion and a current approximation to p as $\hat{x}$, one obtains

$$f(\hat{x}) + (p - \hat{x})f'(\hat{x}) \approx 0,$$

which allows one to use

$$p \approx \hat{x} - \frac{f(\hat{x})}{f'(\hat{x})}.$$

This allows one to start with an initial approximation, and iteratively attempt to find the correct p. Newton's algorithm cannot, however, guarantee convergence. The algorithm is presented as Algorithm 31 [8, 24].

Algorithm 31 Newton's Method

Input: An initial approximation p_0, a tolerance T, and a maximum number of iterations N.

1: Set $k \leftarrow 1$.
2: **while** $k \leq N$ **do**
3: Set $p = p_0 - f(p_0)/f'(p_0)$
4: **if** $|p - p_0| < T$ **then**
5: Output p; STOP.
6: **end if**
7: $k \leftarrow k + 1$
8: $p_0 \leftarrow p$
9: **end while**
10: Output "Method failed after N iterations"

B.3 Regression Analysis

In statistics, regression analysis is a collective name for techniques for the modeling and analysis of numerical data consisting of values of a dependent variable (also called response variable or measurement) and of one or more independent variables (also known as explanatory variables or predictors). The dependent variable in the regression equation is modelled as a function of the independent variables, corresponding parameters, and an error term. The error term is treated as a random variable. It represents unexplained variation in the dependent variable. The parameters are estimated so as to give a best fit of the data. Most commonly, the best fit is evaluated by using the least squares method, but other criteria have also been used.

An equation of the form $y = ax + c$ is used for linear regression. Expanding this to include an error term yields $y = ax + c + u$, where u varies according to some probability distribution. The test of significance approach is frequently used to determine how well a fitted regression line matches the data. This involves the calculation of a test statistic according to a known probability distribution (typically the t distribution), which allows the null hypothesis of a lack of a relationship between the estimator (regression line) and the actual measurement to be evaluated probabilistically [38].

B.4 Numerical Integration

The need for numerical integration (or *quadrature*) arises when a function has no antiderivative, or the integration is too complex. A sum of is used to approximate the function. In the simplest case, a trapezium is constructed between the points $(x_0, f(x_0))$, $(x_1, f(x_1))$, and $[x_0, x_1]$. Thus $\sum_{i=0}^{n} a_i f(x_i)$ approximates $f(x)$.

If the points are kept close enough to each other, the area of the trapezium approaches the integral over the region $[x_0, x_1]$. This concept can exploited to n points over the region, hopefully gaining a more exact approximation. The Trapezoidal rule, Simpson's rule, and Simpson's $\frac{3}{8}$'s rule are the approximation formulas for n values of 1, 2 and 3 respectively. *Newton Cotes*

formulas are the class of formulas using n degree polynomials to approximate curves. Due to the high oscillatory nature of polynomials, this method is inapplicable to most functions over a large interval.

It is useful to exploit the aforementioned formulas piece-wise, *i.e.* defining new polynomials to approximate parts of the curve, keeping the intervals small enough to minimize the oscillation. This method greatly reduces the error of approximation, and for a system of n intervals the error is given by the formula

$$\frac{b-a}{180}h_4 f^{(4)}(\mu),$$

where h is the size of the interval and μ is an element of the domain (when using Simpson's rule). Setting $\frac{b-a}{180}h_4 f^{(4)}(\mu) < \epsilon$ where ϵ is the maximum error or deviation allowed, solving for n will give the minimum number of intervals [24].

B.5 Normal Distribution

The normal distribution, also called the Gaussian distribution, is an important family of continuous probability distributions, applicable in many fields. Each member of the family may be defined by two parameters, location and scale: the mean (average, μ) and variance (standard deviation squared, σ^2) respectively. When μ is 0 and the standard deviation is 1 ($N(0,1)$), the distribution is known as the standard normal distribution. Measurements of physical phenomena can be approximated, to varying degrees, by the normal distribution [38].

The continuous probability density function (PDF) of the normal distribution is the Gaussian function

$$\varphi_{\mu,\sigma^2}(x) = \frac{1}{\sigma\sqrt{2\pi}}\epsilon^{-\frac{(x-\mu)^2}{2\sigma^2}} = \frac{1}{\sigma}\varphi\left(\frac{x-\mu}{\sigma}\right) \qquad x \in \mathbb{R}$$

and $\varphi(x) = \varphi_{0,1}(x) = \epsilon^{\frac{-x^2/2}{\sqrt{2\pi}}}$ $x \in \mathbb{R}$ for $\mu = 0$ and $\sigma = 1$.

The cumulative density function (CDF), which measures the probability of a randomly selected variable X being less than or equal to some point x on a probability distribution, may be expressed as

$$\begin{aligned}\Phi_{\mu,\sigma^2}(x) &= \int_{-\infty}^{x} \varphi_{\mu,\sigma^2}(u)\,du \\ &= \frac{1}{\sigma\sqrt{2\pi}}\int_{-\infty}^{x} \exp\Big(-\frac{(u-\mu)^2}{2\sigma^2}\Big)\,du \\ &= \Phi\Big(\frac{x-\mu}{\sigma}\Big), \quad x \in \mathbb{R}.\end{aligned}$$

In simulation studies, it is often necessary to generate normally distributed samples for model parameter values. The normal distribution has the useful feature that any random variable X which is distributed as $N(\mu,\sigma^2)$ is isomorphic to the standard normal distribution by $X = \mu + \sigma Y$ where $Y \in N(0,1)$. There are numerous techniques for generating stochastic samples from $N(0,1)$, such as the *Box-Muller method* or the *polar method*.

B.6 Uniform Distribution

For the continuous uniform distribution, the PDF is

$$f(x) = \begin{cases} \frac{1}{b-a} & \text{for } a \leq x \leq b, \\ 0 & \text{for } x < a \text{ or } x > b. \end{cases}$$

The values at the two boundaries a and b are usually unimportant because they do not alter the values of the integrals of f(x) dx over any interval, nor of x f(x) dx or the like [38].

The CDF of the uniform distribution is

$$F(x) = \begin{cases} 0 & \text{for} x < a, \\ \frac{x-a}{b-a} & \text{for } a \leq x < b, \\ 1 & \text{for } x \geq b. \end{cases}$$

Appendix C

Algorithmic Examples

Two simple examples are provided here for the GA and ACO, in order to provide the user with a firm understanding of the underlying strategies of these algorithms.

C.1 Genetic Algorithm Example

Consider a simple application of the genetic algorithm. Suppose that the fitness of an individual in a population is determined by the mathematical function $f(x) = -x^2+15x$ (higher is better — see Figure C.1), and each individual possesses a 4 bit binary string constituting its chromosome. Such a string may represent the values 0-15 in binary arithmetic. The objective function is

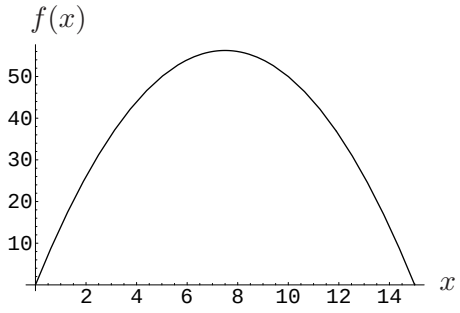


Figure C.1: *The fitness function* $f(x) = -x^2 + 15x$.

positive in the range 0-15, with a maximum occurring at $x = 7.5$. Since the individuals can only be discrete integers, no individual will be able to attain the maximum fitness provided by the function, but a value of $x^* = 7$ or $x^* = 8$ both result in the highest possible fitness of $f(x^*) = 56$. Now consider an initial population comprising 4 members. This is an unrealistically low figure used only for illustration purposes. Suppose a crossover probability of 70% and a mutation probability per bit of 2% are used. Table C.1 contains the randomly generated initial population. The combined (total) fitness of the first generation is 148. The method used to select individuals for reproduction is *fitness-proportionate selection*, whereby the number of times an individual is expected to reproduce is proportional to the ratio of its fitness to the total fitness of the population. The selection process uses a pseudo-random number generator to select two pairs of parents — A,A and A,D. Note that in this model solutions are allowed to 'reproduce' with themselves. Parents A,A can only generate identical copies of themselves, yielding E and F in Table C.2, although child E undergoes a mutation of its first bit. This

Individual	Chromosome String	Decimal Value (x)	Fitness ($f(x) = -x^2 + 15x$)
A	0 1 1 0	6	54
B	1 1 0 0	12	36
C	0 0 0 1	1	14
D	1 0 1 1	11	44
		Total Fitness	148

Table C.1*: First generation population.*

actually has the effect of reducing E's fitness from 54 to 14, but it adds desired variety to the population. The next reproduction pair A,D have offspring G and H, where G is the result of a genetic crossover occurring at bit position 3. G has the first two bits of A and the last two bits of D. Child H is an identical copy of D — containing none of A's chromosome. The total fitness of the second generation has increased to 168. For the next reproduction session, parent

Individual	Chromosome String	Decimal Value (x)	Fitness ($f(x) = -x^2 + 15x$)
E	**1** 1 1 0	14	14
F	0 1 1 0	6	54
G	0 1 1 1	7	56
H	1 0 1 1	11	44
		Total Fitness	168

Table C.2*: Second generation population*

pairs G,F and G,F are selected. These are the two fittest individuals and therefore have the highest probability of selection. Third generation children I, J, K, and L (Table C.1) are the result of genetic crossovers between G and F occurring at various it positions. No mutations occur and the total population fitness increases to 220. For the next generation, parent pairs

Individual	Chromosome String	Decimal Value (x)	Fitness ($f(x) = -x^2 + 15x$)
I	0 1 1 0	6	54
J	0 1 1 1	7	56
K	0 1 1 0	6	54
L	0 1 1 1	7	56
		Total Fitness	220

Table C.3*: Third generation population.*

L,I and L,J are selected. A crossover occurs between L and I at the second bit position to generate individual N. Although crossovers also occur between L and J, their chromosomes are identical so that their offspring are equivalent. The new individuals are now as fit as possible. The population fitness of 224 has been maximized in only four generations (Table C.1).

C.2 Ant Colony Search Example

This simplified example comes from Maier *et al.* [168]. Consider Figure C.2. The eight sub-figures show the progression over time of an ant colony finding the shortest path around an obstacle from their nest at position A to a food source at position B, illustrating their method of social collaboration. The distances are indicated in the first sub-figure (for example, distance AC is *d=1* length units). The time units are the time it takes for an ant to traverse a single

Individual	Chromosome String	Decimal Value (x)	Fitness ($f(x) = -x^2 + 15x$)
M	0 1 1 1	7	56
N	0 1 1 1	7	56
O	0 1 1 1	7	56
P	0 1 1 1	7	56
		Total Fitness	224

Table C.4*: Fourth generation population.*

distance unit. In the following sub-figures, the pheromone concentration on a path is indicated by means of a value p. This value is equivalent to the number of ants which have traveled along that path. Initially, all paths have a value of $p = 0$.

At time $T = 0$, sixteen ants depart from the nest in search of food (the bracketed figures indicate how many ants are headed down a path in the direction of the arrow). At time $T = 1$ the ants reach position C, and have deposited 16 units of pheromone into path AC. At this point the ants must decide which path to take. Since both paths are unexplored, there is an equal chance of taking either one. It is assumed therefore that eight ants will choose path CD and eight will choose path CE. At time $T = 2$, the eight ants which chose path CE have reached E, so that the pheromone concentration of CE is 8. The ants who chose CD are only halfway along its length, and the pheromone count of the path is only compounded once they reach D. Three time units later, at time $T = 5$, the ants who were at E have traversed EF and FB, and then back to F, depositing 16 units of pheromone on FB, whereas the ants who went the long way around have only just arrived at F. The pheromone counts on the paths FD and FE are both equal to 8, therefore the returning group will split into two groups of four each. At time $T = 7$, the group who were on their way to the food have travelled to B and back again, depositing an additional 16 units of pheromone on FB, totalling 32 units — this group also splits into two groups of four going to D and E. Meanwhile, the other groups of four have travelled FD and FEC respectively.

At time $T = 9$ the group which was at C has returned to the nest and has set out again, travelling along CA and then AC, depositing an additional 8 units of pheromone on path AC. In the same time, the other group of four which was heading down FE has reached C, depositing another 4 units of pheromone on path EC, so that the total is 16 units. Meanwhile, the group of four at D have traveled DC, raising its pheromone count to 12. The ants who have just reached C from A must now choose which path to take to the food. Since path CE has a higher pheromone count than CD, the ants are more likely to follow CE. For arguments sake, say three ants set off down CE and one down CD. As time continues, it is easy to see that the shorter path will become dominant. The principle to be learnt is simple. More ants will travel over a shorter (more efficient) path than a longer one in the same time span, reinforcing the pheromone concentration of that path and increasing its desirability.

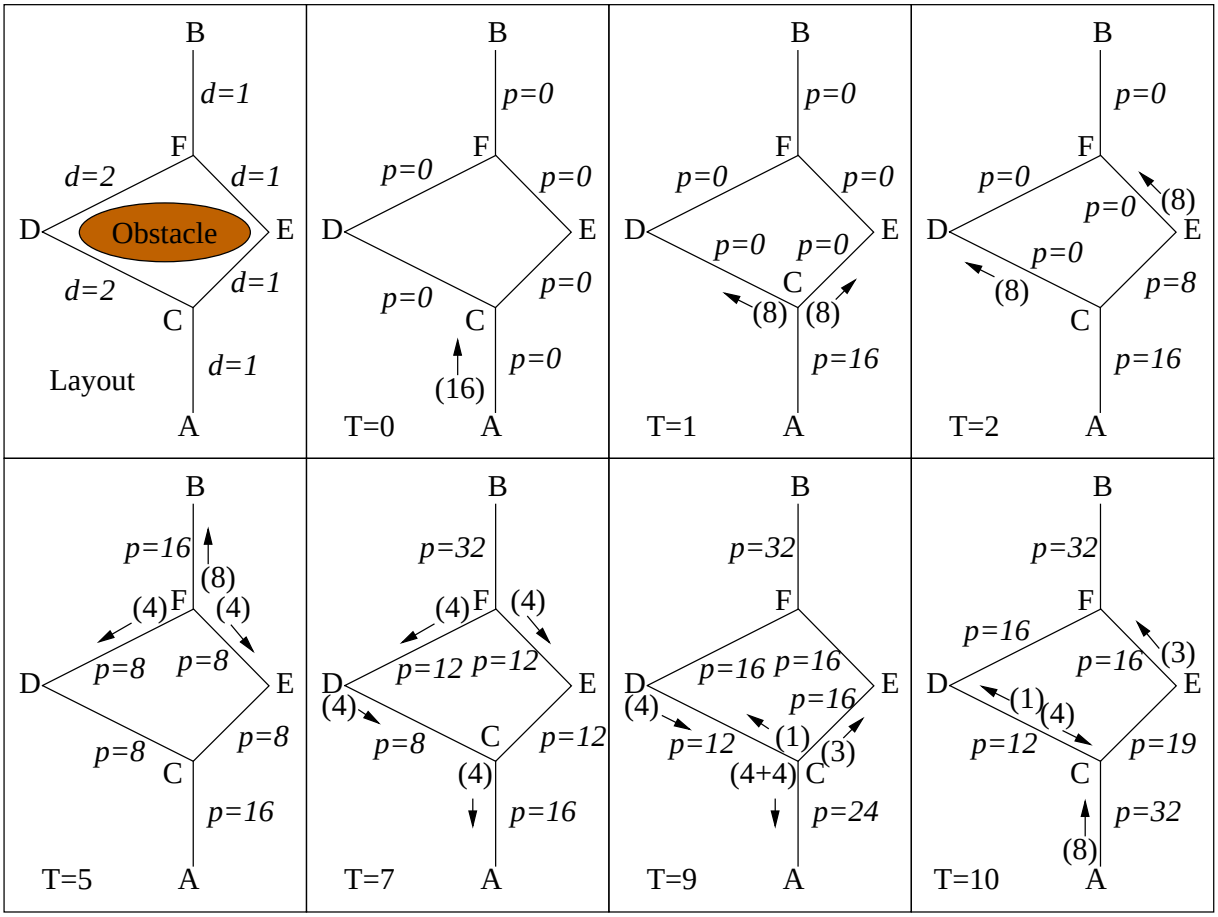


Figure C.2: *The shortest path finding mechanism of an ant colony.*

Appendix D

Optimisation Routine: Input Format

The optimisation routine named Waternet.exe was coded in C++ and uses as input an '.inp' file that contains the physical specification of a particular WDS and has the same input format as EPANET2 (EPANET2 is called as a subroutine). Additionally, it requires a '.opt' file with the same name containing the optimisation settings for that WDS. The format of these two files for the R21 Corridor system appears next. Note that any line preceded by a semicolon is a comment that is ignored. Furthermore, the identifiers of pipes and nodes must be numeric, but not necessarily in a particular order. However, the pipe diameter options should appear in increasing order of size, as many of the algorithms assume this is the case.

This '.inp' format illustration is not comprehensive. The full format specification appears in the EPANET2 Users Manual [204], made available on the accompanying CD.

Input file: "R21.inp"

[TITLE]
R21 Corridor WDS (Atteridgeville, Erkuhuleni)

[JUNCTIONS]

;ID	Elevation	Demand	Pattern
; –	m	l/s	–
86	1 649.808	0	1
88	1 575.698	10.3	1
90	1 650.27	0	1
91	1 630.131	0	1
92	1 570.321	10.3	1
96	1 628.97	0	1
97	1 561.797	10.3	1
101	1 604.409	0	1
102	1 565.795	4.5	1
106	1 590.173	0	1
107	1 579.097	4.5	1
111	1 593.238	7.9	1
116	1 590.001	7.9	1
117	1 585.106	7.9	1
121	1 602.184	7.9	1

122	1 566.384	7.9	1
126	1 601.9	7.9	1
131	1 612.68	7.9	1
136	1 603.938	7.9	1
141	1 613.226	7.9	1
146	1 602.761	7.9	1
150	1 614.388	7.9	1
151	1 593.916	7.9	1
155	1 603.896	7.9	1
157	1 599.065	7.9	1
158	1 635.887	0	1
162	1 632.199	0	1
163	1 629.394	0	1
167	1 622.4	0	1
172	1 612.801	0	1
177	1 602.288	0	1
182	1 600	0	1
187	1 593.533	2.2	1
192	1 587.578	2.2	1
197	1 589.137	2.2	1
202	1 589.137	2.2	1
207	1 607.182	2.2	1
212	1 573.055	0	1
217	1 580.293	4.5	1
222	1 593.084	4.4	1
227	1 606.625	4.4	1
232	1 590	2.2	1
237	1 595.048	2.2	1
242	1 596.694	2.2	1
247	1 612.636	2.2	1
252	1 600	2.2	1
257	1 599.052	3.7	1
262	1 606.592	3.7	1
267	1 604.163	3.7	1
277	1 607.272	0	1
282	1 605.377	9.5	2
287	1 603.346	9.5	2
292	1 602.083	9.5	2
297	1 597.741	9.5	2
302	1 592.063	9.5	2
307	1 580.563	9.5	2
312	1 587.149	9.5	2
317	1 592.238	3.9	1
322	1 588.636	3.9	1
332	1 583.609	3.9	1
337	1 564.546	4.5	1
342	1 577.904	3.9	1
347	1 590.505	3.9	1
357	1 572.943	0	1
362	1 587.472	8.7	2
367	1 597.906	8.7	2
372	1 598.054	8.7	2
377	1 607.223	8.7	2
382	1 614.492	8.7	2
387	1 613.147	8.7	2

397	1 594.951	10.3	1
402	1 596.012	10.3	1
407	1 590	10.3	1
412	1596.108	10.3	1
417	1593.98	10.3	1

[RESERVOIRS]

;ID	Head
; –	m
81	1652.500

[TANKS]

;ID	Elev.	Init.	Min.	Max.	Diam.	MinVol	VolCurve
;–	m	m	m	m	m	m^3	–

;no tanks present

[EMITTERS]

;ID	Flow Coeff

;no emitters present

[PIPES]

; ID	Node 1	Node 2	Length (m)	Diam. (mm)	RCoeff.	LCoeff (CV)
1	81	90	65.782	726	0.1	0
2	81	86	67.456	976	0.025	0
3	407	88	1 100.012	322	0.025	0
4	90	163	853.095	726	0.1	0
5	86	91	808.687	976	0.025	0
6	88	92	1 049.382	322	0.025	0
7	91	96	597.345	976	0.025	0
8	92	97	696.525	322	0.025	0
9	96	101	779.517	976	0.025	0
10	97	102	688.665	322	0.025	0
11	101	106	1 176.919	777	0.025	0
12	102	107	693.839	322	0.025	0
13	106	111	1 328.135	286	0.025	0
14	317	107	738.327	322	0.025	0
15	111	116	227.272	182	0.025	0
16	111	117	377.473	227	0.025	0
17	116	121	941.809	182	0.025	0
18	117	122	1 155.206	227	0.025	0
19	126	121	134.614	182	0.025	0
20	131	126	1166.876	182	0.025	0
21	136	131	1124.708	675	0.025	0
22	141	136	667.668	675	0.025	0
23	131	150	719.268	227	0.025	0
24	146	141	878.81	675	0.025	0
25	150	151	815.574	227	0.025	0
26	157	146	347.602	675	0.025	0
27	101	157	441.81	675	0.025	0
28	151	155	1 336.207	227	0.025	0
29	163	158	370.646	726	0.025	0
30	158	162	311.696	726	0.025	0

31	162	167	472.308	726	0.025	0
32	167	172	508.588	726	0.025	0
33	172	177	642.09	726	0.025	0
34	177	182	412.397	726	0.025	0
35	182	187	358.365	726	0.025	0
36	187	192	317.258	726	0.025	0
37	192	197	450.051	227	0.025	0
38	197	202	406.549	227	0.025	0
39	202	207	576.672	227	0.025	0
40	192	212	798.446	726	0.025	0
41	212	217	507.656	726	0.025	0
42	217	222	572.172	322	0.025	0
43	222	227	548.41	322	0.025	0
44	217	232	755.095	322	0.025	0
45	232	237	240.19	322	0.025	0
46	232	242	547.734	227	0.025	0
47	242	247	448.942	227	0.025	0
48	247	252	461.212	227	0.025	0
49	237	257	409.535	322	0.025	0
50	257	262	667.581	227	0.025	0
51	257	267	382.609	322	0.025	0
52	267	227	405.367	322	0.025	0
53	227	277	93.513	428	0.025	0
54	277	282	542.87	286	0.025	0
55	282	287	471.842	286	0.025	0
56	287	292	380.83	286	0.025	0
57	292	297	400.409	286	0.025	0
58	297	302	475.652	286	0.025	0
59	307	302	1 009.589	286	0.025	0
60	312	307	894.515	286	0.025	0
61	317	312	1 172.701	286	0.025	0
62	322	317	619.901	322	0.025	0
63	277	322	540.41	322	0.025	0
64	332	317	456.401	227	0.025	0
65	337	332	659.768	227	0.025	0
66	342	337	1047.515	227	0.025	0
67	342	347	485.255	322	0.025	0
68	347	277	597.906	322	0.025	0
69	357	342	728.84	322	0.025	0
70	362	357	816.463	322	0.025	0
71	367	362	597.304	675	0.025	0
72	372	367	211.854	675	0.025	0
73	377	372	470.718	675	0.025	0
74	382	377	537.272	675	0.025	0
75	387	382	437.266	675	0.025	0
76	101	387	602.973	675	0.025	0
77	106	397	376.871	675	0.025	0
78	397	402	465.392	675	0.025	0
79	402	407	448.446	675	0.025	0
80	407	412	447.379	675	0.025	0
81	412	417	537.311	675	0.025	0
82	362	417	611.11	675	0.025	0
83	141	155	2 955.915	0.0001	0.025	0
84	88	122	714.126	0.0001	0.025	0
85	207	252	1 504.713	0.0001	0.025	0
86	252	262	1 648.126	0.0001	0.025	0
87	262	267	688.391	0.0001	0.025	0
88	172	207	1 117.011	0.0001	0.025	0
89	337	97	1 070.074	0.0001	0.025	0

```
[PUMPS]
 ;Head                Tail    Properties
 ;ID                  Node    Node
 ;no pumps present

[CURVES]
 ;ID                  Flow    Head
 ;–                   l/s     m
 ;no curves present

[VALVES]
 ;ID                  Head       Tail       Diam   Type   Setting   (Losscoeff)
 ;–                   Node ID    Node ID    mm     –      l/s       m
 ;no valves present

[PATTERNS]
;ID     Multipliers...


[OPTIONS]
 UNITS                LPS
 HEADLOSS             D-W
 QUALITY              NONE
 VISCOSITY            0.1
 DIFFUSIVITY          1.0
 SPECIFIC GRAVITY     1.0
 TRIALS               40
 ACCURACY             0.01
 UNBALANCED           STOP
 EMITTER EXPONENT     1.0
 TOLERANCE            0.01
 MAP                  D:\ Masters \ Code \ WATERNET \ Waternet \ R21.map
;note: EPANET allows a coordinate map file in order to draw the network


[TIMES]
 Duration             21:00
 Hydraulic Timestep   1:00
 Quality Timestep     1:00
 Pattern Timestep     1:00
 Pattern Start        0:00
 Report Timestep      1:00
 Report Start         0:00
 Start ClockTime      12 pm
 Statistic            None

[REPORT]
 STATUS    NO
[END]
```

End of “R21.inp”

Optimisation file: "R21.opm"

```
[TITLE]
R21 Corridor WDS (Atteridgeville, Erkuhuleni)


[UNITS]
FLOW CMS
ROUGHNESS HW
COST R


[PIPE COST]
 SZ      127     145     182     227     286     322     363     428     479     530
 574     626     675     726
 SZ      777     828     878     929     976     1074    1176    1366    1568    1773
 1970    2174
 FN1     263     293     374     500     714     869     1058    1353    1472    1684
 1832    2228    2346    2557
 FN1     2687    3060    3062    3335    3539    4564    5078    7599    9551    10634
 13426   14688
 FN2     31000   36000   46000   59000   77000   89000   103000  126000  146000  166000
 185000  207000  230000  254000
 FN2     279000  306000  332000  361000  387000  446000  511000  643000  798000  971000
 1153000 1356000


[PIPE SIZE GROUPS]
;Id    Option / Size in (mm)
 SZ1   127   145   182   227   286   322   363   428   479   530   574   626   675   726
 SZ1   777   828   878   929   976   1074  1176  1366  1568  1773  1970  2174  E


[PIPE OPTION GROUP 1]
 PIPES                 All
 TYPE                  new
 PRICE FUNC            FN1
 SIZE GROUP            SZ1
 NEW ROUGHNESS         0.0
 SPECIAL COST          0.0


[PIPE OPTION GROUP 2]
 PIPES                 282
 TYPE                  new
 PRICE FUNC            FN1
 SIZE GROUP            SZ1
 NEW ROUGHNESS         0.0
 SPECIAL COST          100000


[PIPE CONSTRAINTS]
 ;Id   MinFlushVelocityAtPeak (m/s)   MaxVelocity (m/s)   MaxEnergyGradient
 ID    MNFLV                          MXV                 MXEG
 All   0                              2.68224             100
```

[LOADING CONDITION 1]

TYPE	Steady State
DESCRIPTION	Peak Condition 1
;FAILED PIPES	
;FAILED PUMPS	
DEMAND FACTOR	4
MIN HEAD	-1.E10
MAX HEAD	1.E10

NODE	FLOW	MINHEAD	MAXHEAD
86	0	-100	92
88	10.3	30	92
90	0	-100	92
91	0	-100	92
92	10.3	30	92
96	0	-100	92
97	10.3	30	92
101	0	-100	92
102	4.5	25	92
106	0	-100	92
107	4.5	25	92
111	7.9	25	92
116	7.9	25	92
117	7.9	25	92
121	7.9	25	92
122	7.9	25	92
126	7.9	25	92
131	7.9	25	92
136	7.9	25	92
141	7.9	25	92
146	7.9	25	92
150	7.9	25	92
151	7.9	25	92
155	7.9	25	92
157	7.9	25	92
158	0	-100	92
162	0	-100	92
163	0	-100	92
167	0	-100	92
172	0	-100	92
177	0	25	92
182	0	25	92
187	2.2	25	92
192	2.2	25	92
197	2.2	25	92
202	2.2	25	92
207	2.2	25	92
212	0	25	92
217	4.5	25	92
222	4.4	30	92
227	4.4	30	92
232	2.2	25	92
237	2.2	25	92
242	2.2	25	92
247	2.2	25	92
252	2.2	25	92
257	3.7	25	92

262	3.7	25	92
267	3.7	25	92
277	0	30	92
282	9.5	35	92
287	9.5	35	92
292	9.5	35	92
297	9.5	35	92
302	9.5	35	92
307	9.5	35	92
312	9.5	35	92
317	3.9	30	92
322	3.9	30	92
332	3.9	30	92
337	4.5	25	92
342	3.9	30	92
347	3.9	30	92
357	0	25	92
362	8.7	35	92
367	8.7	35	92
372	8.7	35	92
377	8.7	35	92
382	8.7	35	92
387	8.7	35	92
397	10.3	30	92
402	10.3	30	92
407	10.3	30	92
412	10.3	30	92
417	10.3	30	92

[LOADING CONDITION 2]

TYPE	Steady State
DESCRIPTION	Fire Condition 1
;FAILED PIPES	
;FAILED PUMPS	
DEMAND FACTOR	2
MIN HEAD	-1.E10
MAX HEAD	1.E10

NODE	FLOW	MINHEAD	MAXHEAD
86	0	-100	92
88	10.3	9	92
90	0	-100	92
91	0	-100	92
92	10.3	9	92
96	0	-100	92
97	10.3	9	92
101	0	-100	92
102	4.5	8	92
106	0	-100	92
107	4.5	8	92
111	7.9	8	92
116	7.9	8	92
117	7.9	8	92
121	7.9	8	92

122	7.9	8	92
126	7.9	8	92
131	7.9	8	92
136	7.9	8	92
141	7.9	8	92
146	7.9	8	92
150	7.9	8	92
151	7.9	8	92
155	7.9	8	92
157	7.9	8	92
158	0	-100	92
162	0	-100	92
163	0	-100	92
167	0	-100	92
172	0	-100	92
177	0	8	92
182	0	8	92
187	2.2	8	92
192	2.2	8	92
197	2.2	8	92
202	2.2	8	92
207	2.2	8	92
212	0	8	92
217	4.5	8	92
222	4.4	9	92
227	4.4	9	92
232	2.2	8	92
237	2.2	8	92
242	2.2	8	92
247	2.2	8	92
252	2.2	8	92
257	3.7	8	92
262	3.7	8	92
267	3.7	8	92
277	0	9	92
282	9.5	10	92
287	25	15	92
292	25	15	92
297	25	15	92
302	25	15	92
307	9.5	10	92
312	9.5	10	92
317	3.9	9	92
322	3.9	9	92
332	3.9	9	92
337	4.5	8	92
342	3.9	9	92
347	3.9	9	92
357	0	8	92
362	8.7	10	92
367	8.7	10	92
372	8.7	10	92
377	8.7	10	92
382	8.7	10	92
387	8.7	10	92
397	10.3	9	92

```
 402   10.3   9   92
 407   10.3   9   92
 412   10.3   9   92
 417   10.3   9   92

[ECONOMIC OPTIONS]
 PUMPING COSTS    no

[OTHER OPTIONS]
 OPTIMIZATION METHOD                 unspecified
 OPTIMIZATION TIME LIMIT             31536000
 STATIC FLOW                         yes
 INCLUDE BASE DEMAND SCENARIO        no
 INCLUDE INITIAL                     no
 OPTIMIZE PUMP SCHEDULE              no
 PENALTY FACTOR                      30156608915
 HYPERVOLUME THRESHOLD               0.0005
 POPULATION SIZE                     100
 GENERATIONS                         10000000
 CROSSOVER PROBABILITY               1
 MUTATION PROBABILITY                0.01
;Whichever termination condition of Generations, Time limit, or Hypervolume convergence periods is
reached first will result in termination (use with caution).


[END]
```

End of "R21.opm"

Table D.1: *Demand loading condition for the MOD benchmark.*

Node	Demand (ℓps)	MaxPres m	Node	Demand (ℓps)	MaxPres m	Node	Demand (ℓps)	MaxPres m
1	0.000 06	35.007	91	0.002 00	39.305	181	0	44.108
2	0.001 45	34.875	92	0.002 24	38.860	182	0.000 01	43.953
3	0.005 13	35.797	93	0.000 20	38.571	183	0.001 84	43.366
4	0.002 76	37.254	94	0.002 27	36.861	184	0.000 04	42.690
5	0.000 96	38.235	95	0.001 44	37.332	185	0.001 68	42.155
6	0.000 78	38.545	96	0.002 67	37.395	186	0.002 37	41.674
7	0.001 03	38.545	97	0.000 60	37.529	187	0.000 09	40.806
8	0.002 12	38.413	98	0.002 76	37.503	188	0.001 23	41.325
9	0.001 16	36.321	99	0.000 05	37.761	189	0.003 90	41.271
10	0.001 78	37.497	100	0.002 06	39.724	190	0.001 17	41.157
11	0.000 44	38.000	101	0.003 19	40.243	191	0.001 70	40.728
12	0.000 03	37.112	102	0.004 69	40.840	192	0.002 15	40.732
13	0.000 57	36.426	103	0.001 70	40.716	193	0.004 00	42.296
14	0.000 36	37.481	104	0.000 02	40.754	194	0.004 46	40.095
15	0.003 30	33.243	105	0.000 10	41.123	195	0.000 05	41.111

Node	Demand (ℓps)	MaxPres m	Node	Demand (ℓps)	MaxPres m	Node	Demand (ℓps)	MaxPres m
16	0.000 03	35.150	106	0.001 11	39.650	196	0.000 01	40.155
17	0.002 10	34.971	107	0.001 02	40.227	197	0.004 12	39.473
18	0.004 67	37.906	108	0.000 88	40.203	198	0.000 39	40.061
19	0	37.739	109	0.002 33	40.546	199	0.000 15	39.966
20	0.001 53	36.785	110	0.000 31	40.580	200	0.001 82	39.565
21	0.004 61	37.188	111	0.000 45	42.183	201	0.003 43	39.796
22	0.001 06	36.877	112	0.002 96	39.742	202	0.000 62	37.800
23	0.001 32	37.513	113	0.008 12	40.287	203	0.004 17	38.297
24	0.002 74	39.295	114	0.001 76	39.576	204	0.002 26	39.469
25	0.000 59	39.387	115	0.005 96	38.544	205	0.001 01	37.735
26	0.000 45	39.846	116	0	43.811	206	0.000 54	38.303
27	0.000 64	40.175	117	0.006 34	43.905	207	0.000 77	36.621
28	0.007 11	38.355	118	0	43.769	208	0.000 33	36.465
29	0.000 93	38.204	119	0	43.797	209	0.001 15	37.637
30	0.000 04	38.403	120	0.003 03	43.480	210	0.001 53	37.262
31	0.000 02	38.361	121	0.001 85	43.468	211	0	37.842
32	0.002 93	38.700	122	0.001 77	42.755	212	0.000 30	38.010
33	0.002 34	41.239	123	0.001 48	42.500	213	0.000 32	37.200
34	0.001 94	41.163	124	0	42.452	214	0.000 56	34.201
35	0.001 19	40.987	125	0.001 32	42.402	215	0	34.651
36	0.001 41	41.800	126	0.002 24	40.740	216	0.000 19	33.502
37	0.002 98	41.853	127	0.001 26	42.229	217	0.001 42	33.340
38	0.002 11	41.935	128	0.005 39	42.640	218	0.000 59	39.451
39	0.007 74	40.935	129	0.001 00	42.083	219	0.001 48	40.580
40	0.004 29	42.905	130	0.001 61	41.498	220	0.000 92	42.356
41	0.007 78	43.119	131	0.004 71	40.874	221	0.000 33	40.333
42	0.003 75	41.833	132	0.002 64	38.134	222	0.000 06	39.403
43	0.002 37	41.001	133	0.002 11	38.806	223	0.000 46	42.951
44	0.001 42	40.929	134	0.001 51	38.976	224	0.000 72	42.755
45	0.000 32	40.726	135	0.000 84	38.940	225	0	42.434
46	0.001 14	40.363	136	0.001 05	38.583	226	0	42.556
47	0.001 23	40.820	137	0.001 16	39.133	227	0.000 20	42.843
48	0.001 37	40.794	138	0.002 45	39.443	228	0.000 08	43.460
49	0.001 18	42.823	139	0.001 66	40.375	229	0.001 30	43.450
50	0.001 81	41.155	140	0	35.150	230	0.001 07	36.008
51	0.000 97	41.668	141	0	35.396	231	0.001 03	38.816
52	0.000 55	41.722	142	0	34.659	232	0.000 03	39.110
53	0.002 77	35.224	143	0.000 80	34.659	233	0.001 15	39.612
54	0.000 65	37.377	144	0.000 33	35.051	234	0.001 43	39.642
55	0.001 38	38.016	145	0.000 34	34.795	235	0.004 86	39.505
56	0.008 28	38.084	146	0.001 02	36.549	236	0.004 71	41.959
57	0.001 22	38.365	147	0.001 23	36.890	237	0.001 34	40.087
58	0.003 85	38.451	148	0.000 09	36.549	238	0.001 87	38.343
59	0.002 62	37.735	149	0.000 43	38.814	239	0.000 82	39.195
60	0.002 78	39.016	150	0.001 56	39.183	240	0.000 94	39.329
61	0.001 56	39.451	151	0.000 80	38.690	241	0.000 09	41.582
62	0.001 16	39.395	152	0.001 38	38.688	242	0.001 28	41.434

Node	Demand (ℓps)	MaxPres m	Node	Demand (ℓps)	MaxPres m	Node	Demand (ℓps)	MaxPres m
63	0.001 22	36.549	153	0.000 53	38.481	243	0.000 43	42.590
64	0.002 93	36.058	154	0.000 59	36.246	244	0.000 51	42.498
65	0.001 13	36.693	155	0.002 33	36.996	245	0	42.452
66	0.001 12	36.282	156	0.000 03	36.964	246	0.000 01	42.446
67	0.000 48	35.773	157	0.000 31	37.421	247	0.000 01	43.795
68	0.001 37	35.547	158	0.008 49	37.745	248	0	43.168
69	0.002 26	34.799	159	0.000 32	38.615	249	0.001 75	38.204
70	0.001 31	33.911	160	0.000 21	38.732	250	0.001 03	38.669
71	0.001 06	33.688	161	0.000 02	39.796	251	0.001 26	37.555
72	0.000 38	33.436	162	0.001 23	39.131	252	0.001 36	36.487
73	0.001 76	33.047	163	0.000 99	39.507	253	0.000 14	37.850
74	0.000 56	32.670	164	0.000 55	38.573	254	0	37.595
75	0	33.065	165	0.000 78	38.235	255	0.001 96	37.727
76	0.004 64	33.408	166	0.000 27	41.833	256	0.002 21	43.003
77	0.001 03	33.757	167	0.000 27	41.746	257	0.000 62	35.849
78	0.003 08	35.895	168	0.000 07	41.616	258	0	34.957
79	0.001 60	37.585	169	0.009 47	40.415	259	0.000 50	34.919
80	0.004 49	37.751	170	0.002 64	38.407	260	0.000 11	34.919
81	0.001 25	37.687	171	0.001 02	38.451	261	0.000 12	33.949
82	0.000 87	37.455	172	0.000 88	38.459	262	0.000 22	33.714
83	0.000 72	38.617	173	0.000 49	38.483	263	0.001 28	33.546
84	0.000 49	38.046	174	0	42.743	264	0.000 19	36.745
85	0.003 92	38.339	175	0.001 12	42.590	265	0.000 22	38.537
86	0.000 94	39.509	176	0	42.701	266	0.001 19	37.691
87	0.003 33	38.888	177	0	43.017	267	0.001 69	38.289
88	0.004 17	39.608	178	0.000 01	43.384	268	0.000 43	38.888
89	0.001 44	38.914	179	0	43.404	-	-	-
90	0.001 84	38.800	180	0	43.306	-	-	-

Appendix E

Contents of Dissertation CD

A compact disc accompanies this dissertation, including software, data and electronic documents, organized into the folders **Code**, **EPANET2**, **GSL** and **WDS Benchmarks**. These folders are described next.

Code

This folder includes the C++ source code of the optimisation software developed for this dissertation, the majority of which appear in the subfolder *Waternet*. The most important files here include:

alg_admoea.h
alg_anima.h
alg_de.h
alg_greedy.h
alg_nsgaii.h
alg_pbb.h
alg_pso.h
alg_speaii.h
algorithms.h
earchive.h
elements.h
epanet2.dll
epanet2.h
epanet2.lib
epanet2vc.lib
network.h
optimizer.h
options.h
stdafx.h
utils.h
Waternet.cpp
Waternet2.sln

EPANET2

This folder contains the setup file for EPANET Version 2.00.12, as well as the source code and

programmers toolkit in zipped format. The EPANET 2 manual is also included here, as well as the source code making up the OOTEN toolkit. These programs are freely available for download. EPANET is available from the US EPA website [81]. These files are:

EN2setup.exe
EN2source.zip
EN2toolkit.zip
EPANET2manual.pdf
ooten.zip

GSL

The freely available open source GNU Scientific Library v1.13 was used for this dissertation. The software and reference manual are provided as:

gsl-1.13.tar.gz
gsl-ref.ps

WDS Benchmarks

The input and optimisation files of the ten WDS systems used in this dissertation are included in this folder. These files include:

black.inp
black.opm
exnet.inp
exnet.opm
foss_poly_1.inp
foss_poly_1.opm
hanoi.inp
hanoi.opm
modena.inp
modena.opm
nytun.inp
nytun.opm
pescara.inp
pescara.opm
R21.inp
R21.opm
trp.inp
trp.opm
tln.inp
tln.opm