THE UNIVERSITY OF HULL

Multi-Objective System Optimisation with Respect to Availability, Maintainability and Cost

being a Thesis submitted for the Degree of
Doctor of Philosophy
in the University of Hull

by

Shawulu Hunira Nggada B. Tech (Hons), MSc

March, 2012

To the loving memory of my father, Katsala Hunira Nggada and grandmother, Miti Bata

ABSTRACT

Safety critical engineering systems are becoming increasingly larger and more complex. One way of ensuring the dependability of such systems is via architectural redundancy and replication of components. Use of redundancy has its limitations though, as it can increase the size, weight and cost of a system beyond acceptable levels. An alternative approach to improving dependability is by designing the system with preventive maintenance (PM) in mind. A well articulated PM policy can reduce the occurrence of system failure, thereby improving dependability attributes such as safety, reliability and availability as well as cost.

In a typical scenario, components of the system are maintained periodically at a fixed time interval (month, year, etc). This interval may vary from component to component and therefore the determination of an optimal PM schedule for all components in the system is non trivial. The options for maintenance are simply too many to exhaustively enumerate and evaluate, and therefore the choice of an optimal PM schedule that provide the best trade-offs between dependability and cost becomes a search and optimisation problem. It is precisely this problem that this thesis addresses.

Firstly, the thesis investigates the effects of perfect and imperfect preventive maintenance policies on system reliability, availability and cost by establishing mathematical models for both policies. Secondly, a multi-objective optimisation approach is formulated for PM scheduling that takes into account dependability and cost, and finally the approach is evaluated on two case studies using a well-established semi-automated dependability analysis tool - HiP-HOPS. The approach allows automatic model transformation such as substitution of components as well as PM maintenance to be applied by Genetic Algorithms as mechanisms for automatically improving design and achieving trade-offs between dependability and cost.

Results from case studies show that this approach can provide an effective tool for definition of PM schedules and lead to engineering and economic benefits.

ACKNOWLEDGEMENTS

I wish to express my profound appreciation and gratitude to my supervisor, Professor Yiannis I. Papadopoulos whom without his continual support, encouragement and impeccable supervision this work would not have been possible. I also greatly value his patience and understanding throughout this period. Indeed, I will always be grateful to him.

I am also grateful to Dr Leonardo Bottaci and Dr Peter Robinson whom through their guidance and feedback I was able to iteratively refine and to produce this work.

Many thanks to Dr Martin D. Walker for his contribution in the development of HiP-HOPS and the encouragement I received from him. I am in particular indebted to Dr David J. Parker for introducing me to HiP-HOPS and also for his implementation of optimisation in HiP-HOPS for which part of this work is an extension.

I have enjoyed the friendship of Dr Ian Wolforth through whom I developed interest in "Formula 1" which like other sports of interest has helped balance my adrenalin level from time to time. I also thank Dr Septavera Sharvia, Dr Mayur Sarangdhar and Dr Zurinahni Zainol for their encouragement throughout the period.

Also thanks to my colleagues Zhibao Mian, Ernest E. Edifor, Julius T. Nganji, Nabil A. Hashish, Hossein Miri, Nongnuch Poolsawad and M. Mostafizur Rahman for their friendship. Special thanks to Amer Dheedan and Nidhal Mahmud whom we shared knowledge and encouraged one another.

I would also like to thank Amanda Millson, Helen El-Sharkawy, Colleen Nicholson, Joan Hopper and Jo Clappison for their tremendous assistance with administrative work in the Computer Science Department. Others who have helped ensure the working of infrastructures I have used in the Department also deserve my appreciation, Mark Bell, David Glover, Mike Bielby and Adam Hird.

I am also indebted to others outside of the University of Hull, and I would like to thank Judith Whiteley, Anthony Whiteley and Dr Ibrahim M. Ibrahim for the encouragement I received from them. I am always indebted to my brother Engr Hyelampa H. Nggada FNSE and his wife Mrs Hyelhara H. Nggada for the all-round support I enjoyed from them throughout this period. Also a warm heart of gratitude goes to another brother of mine Dr Yakubu H. Nggada and his wife Mrs Zharakahyel Y. Nggada for their encouragement. I will always be grateful to my loving mother Ya'asugu Hunira Nggada for her support.

The last but not the least, I deeply thank my wife Mrs Noami S. Nggada whom together we went through this journey. In particular I value her belief in me and her super effort for relentlessly running the family while I was studying. To crown it all, I appreciate my children Miryam, Jennifer and Esther whom throughout this period have not had enough of my time but remained thoughtful.

AUTHOR'S DECLARATION

I declare that the material contained in this thesis is original work conducted solely by the author between December 2007 and March 2012. Some aspects of the work covered in this material have previously been presented and published in international conferences. Specifically, part of the work in chapter 4 forms the basis for Nggada et al (2010a), while some aspects of chapter 6 form the basis for Nggada et al (2010b). Nggada et al (2010a) also appeared in a book chapter (Nggada et al, 2010c).

CONTENTS

1 INTRODUCTION ---------- 1

1.1 Motivation ---------- 1

1.1.1 Difficulties in Design of Safety Critical Systems ---------- 2

1.1.2 Promising Approaches to Addressing Difficulties in Design ---------- 3

1.1.2.1 Improving Dependability through Maintenance ---------- 4

1.1.2.2 Selecting Architectural Components and Maintenance Options ---------- 5

1.2 Research Hypothesis ---------- 6

1.3 Aims and Objectives ---------- 6

1.4 Thesis Structure ---------- 7

2 BACKGROUND I – DEPENDABLE SYSTEMS ---------- 9

2.1 System Safety ---------- 9

2.1.1 Existing Safety Analysis Techniques ---------- 11

2.1.1.1 Qualitative Safety Analysis Techniques ---------- 12

2.1.1.2 Quantitative Safety Analysis ---------- 13

2.1.2 Emerging Safety Analysis Tools ---------- 15

2.1.2.1 HiP-HOPS ---------- 15

2.1.2.2 STPA ---------- 18

2.2 System Maintainability ---------- 20

2.2.1 Maintenance as a Factor in Safety ---------- 20

2.2.2 Segments of System Downtime ---------- 22

2.2.3 Maintenance Models ---------- 23

2.2.3.1 PAR Model ---------- 24

2.2.3.2 PAS Model ---------- 24

2.2.3.3 Shock Model ---------- 25

2.2.4 Improvement Factor Assessment ---------- 25

2.2.5 Advances in System Maintenance ---------- 26

2.2.5.1 Limitation of Work on Optimisation of Maintenance ---------- 29

2.2.5.2 Innovative Approach to Maintenance Problem ---------- 30

3 BACKGROUND II - OPTIMISING SYSTEM DESIGN ---------- 33
3.1 Single Objective System ---------- 34
3.2 Multi-Objective System ---------- 34
3.2.1 Single-Objective Approach to Multi-Objective System ---------- 35
3.2.2 Pareto Optimality Approach to Multi-Objective System ---------- 36
3.3 Search Techniques ---------- 39
3.3.1 Hill Climbing ---------- 39
3.3.2 Simulated Annealing ---------- 42
3.3.3 Genetic Algorithm ---------- 44
3.3.3.1 Niched Pareto Genetic Algorithm II ---------- 48
3.3.3.2 Pareto Envelop-Based Selection Algorithm II ---------- 50
3.3.3.3 Non-Dominated Sorting Genetic Algorithm II ---------- 51
3.3.3.4 Comparison of Multi-Objective Genetic Algorithm Techniques ---------- 55
3.4 Discussion ---------- 56

4 DYNAMIC EFFECTS OF MAINTENANCE ---------- 58
4.1 Perfect Preventive Maintenance ---------- 59
4.1.1 Universal Modelling of the Effect of PPM on Component Reliability ---------- 60
4.1.2 Weibull Distribution Modelling of PPM ---------- 63
4.1.2.1 Reliability ---------- 63
4.1.2.2 Unavailability ---------- 65
4.1.3 PPM Cost ---------- 66
4.2 Imperfect Preventive Maintenance ---------- 66
4.2.1 Universal Modelling of the Effect of IPM on Component Reliability ---------- 68
4.2.2 Weibull Distribution Modelling of IPM ---------- 72
4.2.2.1 Reliability ---------- 72
4.2.2.2 Unavailability ---------- 72
4.2.2.3 IPM Cost ---------- 77
4.2.2.4 System Reliability and Availability Calculation ---------- 78
4.3 Discussion ---------- 79

5 VALIDATION OF DERIVED MODELS FOR RELIABILITY AND UNAVAILABILITY ---------- 80
5.1 Validation of the Established IPM Weibull Reliability Model ---------- 80
5.2 Validation of the Established IPM Weibull Unavailability Model ---------- 89
5.3 Chapter Summary ---------- 92

6 THE APPROACH TO OPTIMISATION OF PREVENTATIVE MAINTENANCE ---------- 93
6.1 PM Encoding ---------- 94
6.2 Generating a PM Individual ---------- 96
6.2.1 Primary Constraints ---------- 96
6.2.2 Secondary Constraints ---------- 97
6.2.2.1 Expert Judgement ---------- 98
6.2.2.2 Architecture Modification through Component Substitution ---------- 99
6.3 Defining the PM Optimisation Problem ---------- 103
6.3.1 Definition of the PM Scheduling Optimisation from Fundamentals ---------- 103
6.3.2 Composite Definition of the PM Scheduling Optimisation ---------- 105
6.4 Diversity in PM Encoding ---------- 106
6.4.1 Evolving a New PM Encoding from Existing Ones ---------- 106
6.4.1.1 Based on Primary Constraints ---------- 107
6.4.1.2 Based on Secondary Constraints ---------- 108
6.5 NSGA II for PM Scheduling ---------- 109
6.6 PM Optimisation Space ---------- 111
6.6.1 PM Solution Space ---------- 111
6.6.2 Feasible PM Region ---------- 111
6.7 Chapter Summary ---------- 113

7 EVALUATION ---------- 115
7.1 Fuel Oil Service System (FOSS) ---------- 115
7.1.1 FOSS Description ---------- 115
7.2 Component Evaluation Models ---------- 120
7.2.1 FOSS Single-Component Evaluation ---------- 120

7.2.1.1 PPM Evaluation---120
7.2.1.2 IPM Evaluation ---121
7.2.1.3 Composite Evaluation of PPM and IPM Models ---123
7.2.2 FOSS Reliability Evaluation---124
7.3 Evaluation of PM Scheduling Optimisation on FOSS ---126
7.3.1 Evaluation of PPM Scheduling Optimisation on FOSS---126
7.3.1.1 Through Primary Constraints---127
7.3.1.2 Through Secondary Constraints ---131
7.3.1.3 FOSS PPM Optimisation from Fundamentals versus Component Substitution 140
7.3.1.4 Composite Evaluation---141
7.3.2 Evaluation of IPM Scheduling Optimisation on FOSS---146
7.3.2.1 Through Primary Constraints---146
7.3.2.2 Through Secondary Constraints ---153
7.3.2.3 FOSS IPM Optimisation from Fundamentals versus Component Substitution 158
7.3.2.4 Composite Evaluation---161
7.4 Further Evaluation of PM Scheduling Optimisation---165
7.4.1 Further Case Study – Aircraft Wheel Brake System---165
7.4.2 PPM Composite Evaluation on AWBS ---170
7.5 Comparison between Manually and Automatically Optimised PM Schedules -------181
7.5.1 Manual PPM Scheduling Optimisation ---181
7.5.2 Automated PM Scheduling Optimisation---183
7.5.3 Automatically versus Manually Obtained PPM Schedules---186
7.6 Revisiting the Research Hypothesis ---198
7.6.1 Objective I - Understand earlier work and understand limitations, e.g. restrictions on system modelling, and modelling of dependability and cost attributes---198
7.6.2 Objective II - Investigate modelling the effects of preventative maintenance --199
7.6.3 Objective III - Investigate cost modelling ---199
7.5.4 Objective IV - Investigate application of heuristics on systems optimisation, especially application of genetic algorithm ---199
7.6.5 Objective V - Define and model the optimisation problem ---200
7.6.6 Objective VI - Design and implement an appropriate optimisation algorithm -200
7.6.7 Objective VII - Evaluate the approach via application on a case study ----200
7.7 Chapter Summary---200

8 CONCLUSIONS ---------- 201
8.1 Research Contributions ---------- 203
8.2 Thesis Limitations ---------- 204
8.3 Suggestion for Further Work ---------- 205

REFERENCES ---------- 207

APPENDIX A – The Puzzle of PM using Exponential Distribution ---------- 216

APPENDIX B – FOSS Optimal PPM Schedules under Component Substitution in Early Generations ---------- 222

APPENDIX C – FOSS Optimal PPM Schedules under Composite Constraint in Early Generations ---------- 226

APPENDIX D – FOSS Optimal IPM Schedules under Component Substitution in Early Generations ---------- 229

APPENDIX E – FOSS Optimal IPM Schedules under Composite Constraint in Early Generations ---------- 231

LIST OF ABBREVIATIONS

ATF	Average Time Failure
AWBS	Aircraft Wheel Brake System
BAO	Bad As Old
BSCU	Brake System Control Unit
CD	Crowding Distance
CMD/AS	Command/Alternative Selector
CMOSA	Classical Simulated Annealing Based Multi-Objective Algorithm
CoMI	Coefficient of Maintenance Interval
CP	Candidate Population
CPU	Central Processing Unit
DT	Down Time
ET	Expert Time
FMEA	Failure Modes and Effects Analysis
FMECA	Failure Modes, Effects and Criticality Analysis
FOSS	Fuel Oil Service System
FTA	Fault Tree Analysis
GA	Genetic Algorithm
GAN	Good As New
HAZOP	Hazards and Operability Analysis
HiP-HOPS	Hierarchically-Performed Hazard Origin and Propagation Studies
HV	High Value
IEC	International Electrotechnical Commission
IPM	Imperfect Preventive Maintenance
LV	Low Value
MIL-HDBK	Military Handbook
MTBF	Mean Time Between Failures
MTTF	Mean Time To Failure
NPGA	Niched Pareto Genetic Algorithm
NSGA	Non-Dominated Sorting Genetic Algorithm
OSA	Orthogonal Simulated Annealing

PAR	Proportional Age Reduction
PAS	Proportional Age Setback
PESA	Pareto Envelop-Based Selection Algorithm
PFD	Probability of Failure on Demand
PM	Preventive Maintenance
PMS	Preventive Maintenance Schedule
PPM	Perfect Preventive Maintenance
RBD	Reliability Block Diagram
RT	Risk Time
STAMP	System-Theoretic Accident Model and Processes
STPA	STamP Analysis
UT	Up Time
WBS	Wheel Brake System

LIST OF FIGURES

Figure 2.1 - Safety analysis techniques 11
Figure 2.2 - Segments of system downtime 22
Figure 3.1 - Pareto optimality 37
Figure 3.2 - Demonstration of the act of dominance 38
Figure 3.3 - Overview of solution space for hill climbing problem 40
Figure 3.4 - Illustration of hill climbing using a chess board 41
Figure 3.5 - A typical binary encoding 46
Figure 3.6 - Typical parent chromosomes 47
Figure 3.7 - Resultant off-spring from uniform crossover 47
Figure 4.1 - Universal modelling of reliability under PPM 61
Figure 4.2 - Component effective age at first PM stage 68
Figure 4.3 - Universal modelling of reliability under IPM 69
Figure 6.1 - One to one mapping between components and CoMIs 95
Figure 6.2 - Concept of architecture modification under PM 100
Figure 6.3 - Typical CoMIs from parents to be recombined 107
Figure 6.4 - Child CoMIs following uniform crossover 107
Figure 6.5 - Typical CoMIs from parents under component substitution 108
Figure 6.6 - Child CoMIs from uniform crossover under component substitution 109
Figure 6.7 - Illustration of a system model with its CoMIs 111
Figure 7.1 - Fuel oil service system (FOSS) 117
Figure 7.2 - *Service tank* (component) reliability under No PM, and under PPM and IPM 124
Figure 7.3 - FOSS reliability under No PM, and under PPM and IPM 126
Figure 7.4 - Pareto frontier of PPM schedules under primary constraints 128
Figure 7.5 - Pareto frontier of PPM schedules under expert judgement 132
Figure 7.6 - Pareto frontier of PPM schedules under component substitution 136
Figure 7.7 - Pareto frontier of PPM schedules under composite constraint 142
Figure 7.8 - Pareto frontier of IPM schedules under primary constraints 146
Figure 7.9 - Pareto frontier of IPM schedules under expert judgement 153
Figure 7.10 - Pareto frontier of IPM schedules under component substitution 156
Figure 7.11 - Pareto frontier of IPM schedules under composite constraint 162

Figure 7.12 - Aircraft Wheel Brake System.....169
Figure 7.13 - AWBS Pareto frontier of PPM schedules under composite constraint...172
Figure 7.14 - Pareto frontier of manually obtained PPM schedules 183
Figure 7.15 - Pareto frontier of PPM schedules obtained through automation.....186
Figure 7.16 – Manually obtained PPM schedules and Pareto frontier of automatically obtained PPM schedules 187
Figure 7.17 - Extracts of PPM schedules in accordance with experts' unavailability and cost value limits 188

Figure A.1 - Component reliability under IPM and No PM using exponential distribution 221

LIST OF TABLES

Table 5.1 – First principle and derived model evaluations of component reliability at maintenance stages 85
Table 5.2 – More detailed first principle and derived model evaluations of component reliability (from 0, 5, 10, .., 295, 300 time unit) 86
Table 5.3 - First principle and derived model evaluations of component unavailability 91
Table 7.1 - Components failure expression for the fuel oil service system 118
Table 7.2 - Summary of evaluation functions used for reliability 119
Table 7.3 - Summary of evaluation functions used for optimising PM schedules 119
Table 7.4 - *Service tank* reliability under No PM and under PPM 121
Table 7.5 - *Service tank* reliability under No PM and under IPM 122
Table 7.6 - FOSS reliability under No PM, and under PPM and IPM 125
Table 7.7 - Results summary for the Pareto frontier of PPM schedules 128
Table 7.8 - Short name representation of actual components name of the FOSS 129
Table 7.9 - A subset of optimal PPM schedules; a tabular representation 130
Table 7.10 - Selected components for expert judgement with their respective expert PM times 131
Table 7.11 - Results summary for the Pareto frontier of PPM schedules under expert judgement 132
Table 7.12 - A subset of optimal PPM schedules under expert judgement; a tabular representation 133
Table 7.13 - Components with their implementations and short name representation . 135
Table 7.14 - Results summary for the Pareto frontier of PPM schedules under component substitution 137
Table 7.15 - Number of occurrences of component implementations forming part of the FOSS PPM optimisation solution vector under component substitution 137
Table 7.16 - A subset of optimal PPM schedules under component substitution; a tabular representation 139
Table 7.17 - Results summary for the Pareto frontier of PPM schedules under composite constraint 142
Table 7.18 - Number of occurrences of component implementations forming part of the FOSS composite PPM optimisation solution vector 143

Table 7.19 - A subset of optimal PPM schedules under composite constraint; a tabular representation 145
Table 7.20 - Results summary for the Pareto frontier of IPM schedules under primary constraints 147
Table 7.21 - A subset of optimal IPM schedules under primary constraints; a tabular representation 150
Table 7.22 - Optimal IPM schedules under primary constraints in generation 1 151
Table 7.23 - Optimal IPM schedules under primary constraints in generation 2 151
Table 7.24 - Optimal IPM schedules under primary constraints in generation 3 152
Table 7.25 - Results summary for the Pareto frontier of IPM schedules under expert judgement 154
Table 7.26 - A subset of optimal IPM schedules under expert judgement; a tabular representation 155
Table 7.27 - Results summary for the Pareto frontier of IPM schedules under 157
Table 7.28 - Number of occurrences of component implementations forming part of the FOSS IPM optimisation solution vector under component substitution 157
Table 7.29 - A subset of optimal IPM schedules under component substitution; a tabular representation 160
Table 7.30 - Results summary for the Pareto frontier of IPM schedules under composite constraint 162
Table 7.31 - Number of occurrences of component implementations forming part of the FOSS composite IPM optimisation solution vector 163
Table 7.32 - A subset of optimal IPM schedules under composite constraints; a tabular representation 164
Table 7.33 - Components failure expression for the aircraft wheel brake system 167
Table 7.34 - Short name representation of actual components name of the AWBS 170
Table 7.35 - Selected components for expert judgement with their respective expert PM times 171
Table 7.36 - Components with their implementations and short name representation 171
Table 7.37 - Results summary for the Pareto frontier of PPM schedules under composite constraint 173

Table 7.38 - Number of occurrences of component implementations forming part of the AWBS composite PPM optimisation solution vector 174
Table 7.39 - A subset of AWBS optimal PPM schedules under composite constraint; a tabular representation 175
Table 7.40 - AWBS optimal PPM schedules under composite constraint in generation 1 177
Table 7.41 - AWBS optimal PPM schedules under composite constraint in generation 2 178
Table 7.42 - AWBS optimal PPM schedules under composite constraint in generation 3 179
Table 7.43 - Manually obtained PPM schedules 182
Table 7.44 - Results summary of automated PPM optimisation 184
Table 7.45 - A subset of optimal PPM schedules obtained through automation 185
Table 7.46 - Improvements in unavailability and cost of automatically obtained PPM schedules over M1 190
Table 7.47 - Improvements in unavailability and cost of automatically obtained PPM schedules over M2 191
Table 7.48 - Improvements in unavailability and cost of automatically obtained PPM schedules over M3 193
Table 7.49 - Improvements in unavailability and cost of automatically obtained PPM schedules over M4 195
Table 7.50 - Improvements in unavailability and cost of automatically obtained PPM schedules over M5 196
Table 7.51 - Improvements in unavailability and cost of automatically obtained PPM schedules over M6 197

Table A.1 - Component reliability using exponential distribution under IPM and No PM 220

Table B.1 – FOSS optimal PPM schedules under component substitution found in generation 1 222

Table B.2 – FOSS optimal PPM schedules under component substitution found in generation 2 223
Table B.3 – FOSS optimal PPM schedules under component substitution found in generation 3 224

Table C.1 – FOSS optimal composite PPM schedules found in generation 1 226
Table C.2 – FOSS optimal composite PPM schedules found in generation 2 227
Table C.3 – FOSS optimal composite PPM schedules found in generation 3 228

Table D.1 – FOSS optimal IPM schedules under component substitution found in generation 1 229
Table D.2 – FOSS optimal IPM schedules under component substitution found in generation 2 229
Table D.3 – FOSS optimal IPM schedules under component substitution found in generation 3 230

Table E.1 – FOSS optimal composite IPM schedules found in generation 1 231
Table E.2 – FOSS optimal composite IPM schedules found in generation 2 232
Table E.3 – FOSS optimal composite IPM schedules found in generation 3 232

1 INTRODUCTION

1.1 Motivation

Safety critical systems are those systems like aircraft and nuclear power plants that cause hazards for people and the environment (Storey, 1996). The dependability of such systems encompasses attributes such as safety, reliability and availability and it is paramount. As modern systems become larger, handle greater volumes of energy and hazardous materials, or become more complex employing complex networked architectures, dependability becomes a growing concern.

One approach to improving dependability is via system analysis prior to deployment, identification of flaws and improvements of system design. A second approach, typically combined with prior analysis is via redundancy and component replication. In this approach, patterns of fault tolerant architectures are employed to detect and respond to component failures in real-time by replacing failed components with replicas which continue to provide functions or by exploiting component redundancies in more complex reconfigurations of the system. Although useful, this approach has its limitations because redundancy can increase cost and weight beyond acceptable levels. A third approach for improving dependability is to employ a scheme of preventative maintenance (PM). A well designed PM schedule for some or all the components of a system can reduce the occurrence of failures, thereby improving dependability properties.

Due to the existence of large numbers of potential PM scheduling options, PM scheduling is naturally formulated as optimisation problem. Finding optimal trade-offs between dependability and cost in PM scheduling is precisely the topic addressed in this thesis.

This section firstly discusses some difficulties in the design of engineering systems, and presents the motivation of this work, leading then to a research hypothesis tested in this thesis and a set of research objectives that define the scope of this work.

1.1.1 Difficulties in Design of Safety Critical Systems

Safety critical systems span numerous application areas including oil and gas, automotive, aerospace (e.g. aircraft and spacecraft), nuclear, chemical, various forms of power generation and manufacturing industries. Society becomes increasingly more reliant on the functions provided by such systems. These systems become increasingly distributed and computer controlled, and this poses new challenges to engineers.

New systems for example introduce new failure modes, electronic and computer controlled systems can fail by commission or inadvertent delivery of functions that can be particularly hazardous. The identification, analysis and mitigation of such new failure modes are of key concern. Systems become more integrated and the density of functions provided by electronic components increases. The likelihood of failures of such components therefore increases. Advances in design theory mean an increasing amount of possibilities for architectural configurations that can deliver a set of functions. Competition among suppliers of components also means a range of options for hardware and software components that can be used to materialise a system. Such components may be similar in their functional profile but would typically have different reliability and cost characteristics. The choice of appropriate components in itself poses a combinatorial optimisation problem.

But it is not only the complexity of systems that grows. Systems also become larger; they handle larger volumes of energy and materials and bring together larger numbers of components in networked and distributed architectures. Increasing scale could mean more options on architectural design and selection of components from implementations. Finally in systems which are subjected to PM, increasing scale also means an increasing number of potential options for PM maintenance scheduling, and this poses additional difficulties in the choice of optimal schedules that can maximize trade-offs between dependability and cost.

1.1.2 Promising Approaches to Addressing Difficulties in Design

To effectively design a safety critical system, engineers need to understand not only how the system should work but also how it can fail. Due to the complexity of the task, design and analysis is typically assisted by computerised tools. With respect to dependability analysis, several classical techniques are in use today: they include techniques like Failure Modes and Effects analysis (FMEA), Fault Tree Analysis (FTA), and Hazards and Operability studies (HAZOP). These techniques emerged in the post-war period (40s - 60s) and have become popular since. Today they are widely applied and their results are often used for improving design and certification of systems (Leveson, 2002). However, these techniques are mostly manually applied, which means that the increasing scale and complexity of modern computer-based systems challenges their applicability and usefulness in design (Galloway et al, 2002). This therefore calls for new, perhaps automated approaches to addressing the problem of dependability assessment.

Even if we assume that the difficulties in dependability analysis have been overcome, the analysis might show that a design does not meet its requirements. Possible revisions might require modifications of the architecture, for example, substituting a component with a more expensive and more reliable component, or substituting part of the architecture with another improved design. In typical designs though the options become too many to consider exhaustively, and the question is how can designers arrive at designs that can achieve dependability requirements with minimal costs. For instance, in a system with 10 components where each component has 10 versions (or implementations), the design space consists of 10,000,000,000 different potential designs. From this point onwards, each design within this design space is termed as *design variant*. Exploring such design space through the use of manual techniques in the search for a design variant that will improve set design objectives is practically infeasible.

Various techniques have been developed in the last 15 years to partly automate and improve dependability analysis and design optimisation of safety critical systems. These techniques will be discussed in relevant chapters of this thesis. A prominent technique

which is supported by a state-of-the-art software tool that performs automated analysis and design exploration is HiP-HOPS (**Hi**erarchically-**P**erformed **H**azard **O**rigin and **P**ropagation **S**tudies). Although the work in this technique has addressed several of the difficulties in modern design of dependable systems, none of this work has yet addressed the problem of maintenance scheduling.

1.1.2.1 Improving Dependability through Maintenance

The components of a system can fail at any time because of the randomness of hardware failures which typically follow probabilistic distributions. The more complex the system, the more it tends to fail frequently and thus, not only reliability but also the availability becomes a concern. In general, the objective of maintenance is to reduce the frequency of failure of components and of the system as a whole, and where failure has occurred, to restore the system back to operation. The former is termed as *preventive maintenance*, while the latter *corrective maintenance*. Maintenance can therefore be said to improve system reliability and availability especially in the case of preventive maintenance.

During preventive maintenance, the state of a component is inspected and where it can not be improved, replacement is carried out, otherwise maintenance activities such as oiling, topping, cleaning, adjusting and tightening are appropriately performed on the component to improve its condition thus reducing the rate at which component fatigue accumulates. One challenge is the timing of maintenance for each component. For instance, it may be possible to clean the spark plug of an automobile engine every six months, nine or twelve months. This decision is not easy when there are many components and many possibilities for maintenance intervals. While the objective is always to create a maintenance schedule for the whole system, the design space of possible maintenance schedules is huge and the choice of a schedule is extremely difficult. The objective of the problem can be defined as finding one or more optimal maintenance schedules that help to achieve optimal trade-offs between dependability requirements and costs. The solution to the problem typically requires a search and

optimisation process, especially in the case of systems with numerous constituent components.

1.1.2.2 Selecting Architectural Components and Maintenance Options

We have so far discussed a number of ways of optimising dependability and cost including choosing among alternative implementations and different maintenance options. Such options do not necessarily need to be treated separately in an optimisation process.

Suppose again, for example, that an engineering system has 10 components and that each of these has several implementations, then it is possible to perform preventive maintenance analysis on the system model and substitution of its components with their respective implementations at the same time. The result of such analysis will be set of optimal preventive maintenance schedules applied on potentially different configurations of the components of the system. This is useful especially at early design stages when the system model could be modified based on a selected optimal preventive maintenance schedule. The advantage of this approach is that the system engineer is well informed about the effects of preventive maintenance on the dependability attributes of the system model based on chosen constituent components, thereby saving design cost and time.

Therefore, extending preventive maintenance analysis with component substitution provides the system engineer with a wider range of optimal system design variants which are influenced by optimal preventive maintenance schedules. An automation of this form of analysis will reduce the design complexity by automatically searching through the huge design space and to produce set of optimal design variants of the system.

1.2 Research Hypothesis

The hypothesis put forward and tested in this thesis is that;

> "The optimisation of complex safety critical systems with respect to availability and cost taking into account the dynamic effects of scheduled preventive maintenance is both feasible and beneficial and can be achieved through a novel integration of state-of-the-art model based safety analysis technique with recent work on meta-heuristics."

1.3 Aims and Objectives

To enable testing of the above hypothesis, the main aim of the thesis is defined as investigating the effects of periodic maintenance policies on the design of safety critical systems, and establishing and demonstrating the scheduling optimisation of such policies through automation.

Thus, the thesis firstly investigates the effect of preventive maintenance on system reliability, availability and cost. Secondly it investigates the possibility of optimising preventive maintenance schedules by extending a mature model-based safety analysis tool (HiP-HOPS) with new capabilities for dependability analysis and optimisation under assumptions of preventive maintenance capability. HiP-HOPS already incorporated mature analysis and optimisation options. It was therefore selected as an appropriate platform for extension and experimentation. Finally, the optimisation of preventive maintenance schedules by allowing for components substitution with respective implementations is investigated.

A number of steps or objectives have been set out to logically progress with the work and achieve the aims set out above, ultimately enabling the testing of the stated hypothesis:

(i) Understand earlier work and understand limitations, e.g. restrictions on system modelling, and modelling of dependability and cost attributes
(ii) Investigate modelling the effects of preventative maintenance
(iii) Investigate cost modelling
(iv) Investigate application of heuristics on systems optimisation, especially application of genetic algorithm
(v) Define and model the optimisation problem
(vi) Design and implement an appropriate optimisation algorithm
(vii) Evaluate the approach via application on case study

1.4 Thesis Structure

The thesis is organised into eight chapters, with chapter one being the introduction. The reminder of the thesis is organised as follows.

Chapter 2: provides relevant background on dependability of systems focusing on relevant definitions and discussion of state-of-the-art techniques for safety, reliability, maintainability and availability analysis. It also investigates earlier work in system maintenance and identifying the approach of this work due to gaps in this literature.

Chapter 3: discusses literature on the optimisation of engineering systems, covering approaches to optimisation, search heuristics, and selection techniques and provides a comparison among different approaches.

Chapter 4: develops mathematical models under which the reliability, availability (and unavailability) and cost of a component or a system can be calculated under assumptions of perfect and imperfect preventive maintenance.

Chapter 5: presents a numerical validation of the developed mathematical models for component reliability and unavailability under the assumptions of imperfect preventive maintenance. The numerical validation consists of a comparison between results

returned by the theoretical models derived in chapter 4 with results returned by calculations of reliability and availability from first principles.

Chapter 6: develops an approach to optimising perfect and imperfect preventive maintenance schedules by setting constraints, defining the optimisation problem and finally establishing an algorithm for the optimisation.

Chapter 7: this chapter applies the established evaluation models developed in chapter 4 and the optimisation method developed in chapter 6 on a case study performed on a model of a *fuel oil service system* that supplies the main engine of a ship. The approach is further evaluated on an *aircraft wheel brake system* to demonstrate application on a larger example and test scalability. Additionally, a comparison is made between manually enumerated preventive maintenance schedules based on expert judgement and those that are automatically obtained with the optimisation algorithms developed in this thesis. The chapter also evaluates the approach established in this thesis against the set research objectives and eventually the hypothesis.

Chapter 8: this chapter provides a summary of this work and draws conclusions. It highlights the contributions made by this work in dependability analysis of safety critical systems, points out limitations and proposes areas for future work.

2 BACKGROUND I – DEPENDABLE SYSTEMS

This chapter presents two relevant literature reviews on the analysis of dependable systems. Firstly, there is a review on safety analysis with focus on techniques involved. Secondly, a review on system maintainability with focus on maintenance is discussed.

A dependable system is one possessing the property that justifies one's reliance on it (Storey, 1996). The term "justifies" refers to the fact that the system needs to attain certain level of design and operation standards. Dependability is a property that encompasses many attributes which include safety, reliability, maintainability and availability (Storey, 1996). It is practically infeasible to fully achieve these attributes with deterministic certainty. However, a system design which takes them into account from early design stage through to completion is likely to attain the level required by the demands imposed by the given application.

2.1 System Safety

The term *safety as applies to systems refers to the property of the system that it will not endanger human life or the environment* (Storey, 1996). The constituent elements of the environment may be arguable; however, it comprises of structures to which damage to would result in economic or societal loss. It may as well comprise of ecological life and therefore environment may be specific to designated area of operation for which a system is designed. When safety measures are considered in system design right from inception, hazards are more likely to be identified early and mitigated. The cost involved in mitigating a system flaw is usually more effective during the design stage than it is after completion and deployment. The occurrence of hazard is usually what is termed as accident. Storey (1996) clearly puts the definition of accident as *an unintended event or sequence of events that causes death, injury, environment or material damage*.

While engineers are aware that systems can not be designed with absolute safety, the design of systems satisfying user requirements under prescribed operating conditions will continue. While the user may largely concentrate his requirements on functional

aspect of the system, the engineer will extend his judgement beyond those requirements by considering safety requirements on both functional and non-functional requirements. A functional requirement refers to the activities for which the system needs to perform in order to succeed the intended system function. Whereas a non-functional requirement refers to properties for which the system should posses to improving performance i.e. weight, size, maintainability, etc.

The challenge for a system engineer is in identifying hazards and to try as much as possible to prevent their occurrences, and at worst (i) reduce the occurrences and, or (ii) reduce the effect of accident. It is possible to predict the effect of hazard when its likelihood of occurrence and severity are known; the result of which is termed as *risk*. *Risk is quantitatively defined as the product of the consequences of a specific event and the probability of its occurrence* (Andrews and Moss, 1993, Storey, 1996).

A simple scenario of system safety requirement can be seen in home appliances like pressure pot; where the lid remains closed while the pot is in use. This way it prevents the user from sustaining burns as a result of exposure to steam pressure when the lid is opened. Another example is seen in automotive reverse sensor, which signals to the driver when the system such as a vehicle is close to an object. The safety measure here may evidently be seen in a densely populated area to prevent reversing over people especially kids. This last scenario is a simple case of the fact that certain safety measures are beyond system control, and the action to be taken is in the hands of the user whom the system has no control over (Leveson, 2002). For instance the driver may proceed even when being warned. It is not surprising that many of accidents come about through the interaction of normal, predictable human behaviour in the conduct of safety-related system design, development, operation and maintenance (Chambers, 2006).

One of strategies of system analysis according to Leveson (2002) is to begin by identifying possible accidents and subsequently the hazards constituting each. In the past, accidents were largely attributed to component failures, but more of accidents occurring today are as a result of poor system design as well as human factor in its

operations. Hence, it is imperative that safety analysis is performed from early system design through to completion.

2.1.1 Existing Safety Analysis Techniques

The safety of systems is undoubtedly dependent on the perceptions of the system engineers involved. However there are techniques which aid the engineers in performing safety analysis. Increasing system complexity makes it difficult for system engineers to consider all system hazards or even the most important ones, or for the operators to handle normal and abnormal operations successfully (Leveson, 1997). This then calls for the automation of existing design and analysis tools to assist system engineers.

Several of these safety analysis techniques exists, usually categorised under qualitative and quantitative analysis. Qualitative safety analysis approach comprises of diagrammatic or hierarchical description of the factors that might cause accidents, while the quantitative estimates the probability of occurrence of each cause, which in turn can be used in estimating the risk of the accident (Netjasov and Janic, 2008). A basic pictorial representation of safety analysis techniques drawn from Rouvroye and van den Bliek (2002) falling under qualitative and quantitative approaches is shown in Figure 2.1.

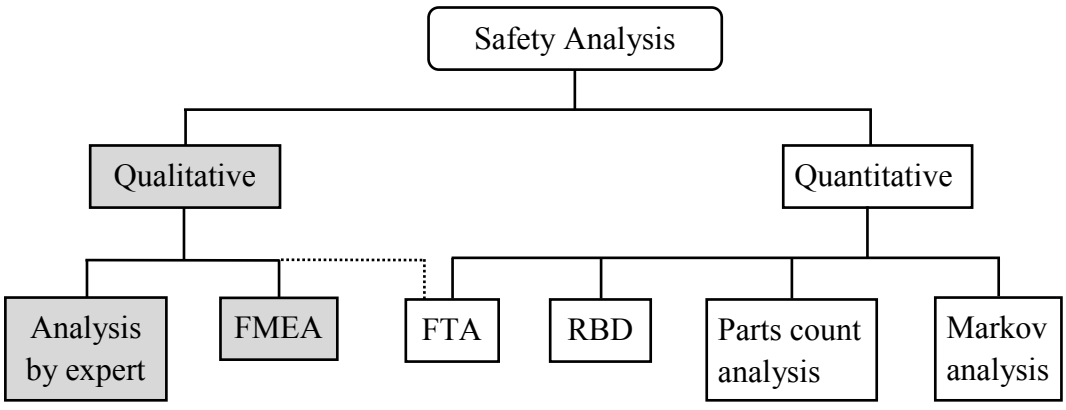

Figure 2.1 - Safety analysis techniques

2.1.1.1 Qualitative Safety Analysis Techniques

Qualitative safety analysis is essential especially where there is inadequate or no data on the components of a system. In such situation, quantitative analysis may be infeasible. At the early design stage of a system, failure data may not be adequate and therefore qualitative approach becomes essential. Two of the safety analysis techniques as described by Rouvroye and van den Bliek (2002) are *analysis by expert* and *FMEA*, and these are here briefly discussed.

Analysis by Expert is the form of safety analysis that solely relies on the previous experience of the engineer in a similar problem area. In this analysis, some existing documents containing information that may add to the quality of the design may be consulted. Some of these documents are codes of practice, guidelines, checklist, etc. The code of practice is a document which outlines the technical procedures for the design of the given system, while the guidelines inform the engineer about the steps that are necessary to achieving the system requirements. A checklist is a document that contains the list of activities for every task involved in the design of the system. On completing every task, the checklist is consulted to ensure no activity is omitted. One thing is certain, that these documents aid in reducing system design flaws.

FMEA (failure modes and effects analysis) is a classical safety analysis technique which is currently in wide use in the automotive, aerospace and other safety critical industries (Papadopoulos et al, 2004). It is also used in reliability analysis and in many cases a non exhaustive FMEA mainly focusing on enumeration of component failure modes provides a pathway to developing system fault trees. FMEA first identifies the possible failure modes for all components present in the system and then infers their effect on the component itself as well as the entire system. Hence, the failure of the system is seen to arise as a result of one or combination of components failure in conjunction with contributing factors that analysts are able to specify. External factors (i.e. electromagnetic field, temperature, humidity, mode of use, etc) may as well affect component operation and eventually system dependability properties; leading to eventual system failure. A component may have several failure modes and therefore

care should be taken not to lead the analysis into catalogues of modes that may not be of importance in addressing the risk of the component or system.

When failure events are annotated with likelihood of occurrence and severity, this becomes a quantitative extension of FMEA known as failure modes, effects and criticality analysis (FMECA). FMECA aids in identifying those sections of the system where failures are most important (Storey, 1996). A component's criticality may be viewed through a *criticality grid* as found in Birolini (2007) also known as *criticality matrix*. The further an entry is from the origin of the criticality grid, the greater is the necessity for a corrective or preventive action (Birolini, 2007).

2.1.1.2 Quantitative Safety Analysis

The quantitative techniques are useful in scenarios where there is substantial numerical/probabilistic data to work with. The safety analysis techniques falling under this category, also found in Rouvroye and van den Bliek (2002) are here briefly discussed.

Reliability block diagram (RBD) is a graphical analysis technique which represents a system as connections of a number of components in accordance with their logical relation of reliability (Guo and Yang, 2007). RBD can be used in calculating the average probability of failure on demand (PFD_{aveg}) as required by IEC 61508 for the verification of safety integrity of a system. PFD_{aveg} is the average probability that the safety function will not be able to perform its function on demand from the process it protects (Rouvroye and van den Bliek, 2002). A demonstration of obtaining PFD_{aveg} using RBD can be seen in Guo and Yang (2007).

Fault tree analysis (FTA) is a technique that analyses the failure mode of a system and considers possible events leading to such failure mode. The events may be assigned a failure data from where it becomes possible to assess the safety through a risk based calculation or probability of failure distribution. It is therefore evident that without assignment of probabilities of failure, FTA would fall under qualitative safety analysis.

Parts count analysis is a technique used for safety analysis during the early (preliminary) design stage. It is also used during proposal formulation where there is less information and one has a "feel" for the number of component parts (actual or estimated) by class or type that will be used in the system. It involves counting the number of components for each class and multiplying this number by the generic failure rate for each component class, and finally summing these products to obtain the failure rate for the system (MIL-HDBK-338B, 1998). This form of analysis is particularly useful when the system has a minimum failure rate target.

Markov Analysis is a technique which analyses the safety of a system by representing different failure states and the transition among these states. It is also referred to as *failure state diagram* (Bukowski and Goble, 1995). Many safety related factors, such as failure modes, self-diagnostic, restorations, common cause and voting, are included in Markov analysis (Guo and Yang, 2008). If there are N failure states, the Markov model will consist of an N x N matrix. If F_i and F_j are two states where one of them is operational state and the other failure state, or both are failure states, the coordinate F_{ij} of the matrix is the transition probability from F_i to F_j; i and j are row and column respectively. This implies that some of the entries are appropriately the failure rate for the transition from F_i to F_j or the repair rate from F_j to F_i.

Markov models can be built into design tools, making it very convenient for designers to utilize (Thimbleby et al, 2001). According to Guo and Yang (2008), Markov analysis is fallible and time-consuming to perform manually and the size of Markov model increases explosively as the system becomes more complex. This setback is addressed through automation as seen in Guo and Yang (2008).

More on safety analysis using Markov analysis can be found in Bukowski and Goble (1995), and Guo and Yang (2008).

2.1.2 Emerging Safety Analysis Tools

Although systems design or scale of operation is growing in complexity and consequently making analysis of such systems difficult, engineers and safety experts are addressing the problem through new techniques. Two of such emerging techniques are discussed.

2.1.2.1 HiP-HOPS

HiP-HOPS is a state-of-the-art compositional system dependability (i.e. safety, reliability and availability) analysis technique first developed by Papadopoulos and McDermid (1999) in the quest to easing system engineers' nightmare. It can be used to perform system analysis from early stages of design through to completion. It also offers a significant degree of automation and reuse, addressing problems arising from the increasing complexity of systems.

HiP-HOPS uses a deductive method in analysing the propagation of failure within a system, this implies that it begins from effects to causes. With HiP-HOPS, the topology of a system is used together with reusable local failure specifications at component level to automatically produce a network of interconnected fault trees and an FMEA for the system. HiP-HOPS is supported by a computerised tool which currently works in conjunction with modelling tools like Matlab Simulink and Simulation X - but can also be interfaced to other modelling packages.

Dependability analysis using HiP-HOPS begins from a system model that has been designed and is available in electronic form. HiP-HOPS dependability analysis consists of four phases; *failure annotation*, *synthesis*, *analysis* (Papadopoulos et al 2008) and *optimisation* (Parker and Papadopoulos 2007, Parker 2010).

The first phase involves a manual action of annotating the components of a system model with failure behaviour data. The failure behaviour added to each component describes how deviations of component parameters from intended behaviour and their causes as logical combination of internal malfunctions and deviations of inputs are

specified in logical expressions - effectively sets of local mini-fault trees. The remaining phases of system analysis with HiP-HOPS are fully automated.

The next phase is the *synthesis,* where mini fault trees are linked together producing a set of system fault trees, one fault tree for each of the deviations of system output. The *analysis* phase which is next, applies traditional fault tree analysis techniques to the fault trees producing minimal cut sets. A cut set is a set of basic events (component failure or external event) whose simultaneous failure will cause system failure (Storey, 1996 and Fard, 1997). Quantitative analysis of these cut sets can then be used for evaluating the system reliability and unavailability values.

The phase which is optional depending on the requirement of the analysis is *optimisation*. This phase has been enabled by recent work on meta-heuristics and has extended HiP-HOPS with the capability of supporting multi-objective optimisation. This capability allows the HiP-HOPS tool to search the design space, defined by the variability of a design model, for potential design solutions that are optimal, or near optimal, in terms of dependability and cost. In this approach, a variable design model for a system is one in which components and subsystems have alternative user defined implementations which can include standard fault tolerant configuration schemes. HiP-HOPS uses a multi-objective genetic algorithm to effectively search the design space defined by the permutations of the design that can arise following resolution of variability. The genetic algorithm exploits the automated fault tree and FMEA synthesis and analysis algorithms of the tool to calculate the fitness of candidate designs. The goal is to identify Pareto optimal architectures for the system which give optimal trade-offs between dependability, cost and other parameters.

HiP-HOPS defines a language for the description of failure behaviour at component level. In the basic version of this language, the failure behaviour of a component can be specified as a list of its internal failure modes (internal malfunctions) and a list of deviations of parameters as they can be observed at its outputs (output deviations). Each internal malfunction is optionally accompanied by quantitative data, for example a failure and a repair rate if these are known. Output deviations carry Boolean expressions

which describe their causes as a logical combination of internal malfunctions of the component and similar deviations of parameters at component inputs (input deviations). A general form of HiP-HOPS' failure logic is as seen below.

Output Deviation = Internal Failure OR/AND Input Deviation

The output deviation consists of two parts concatenated by a hyphen. First the failure class of the deviation and then the port at which the deviation occurs. HiP-HOPS define several failure classes of deviation and these are normally abbreviated. These are `O` for omission failure, `C` for commission failure, `V` for value failure (where `LV` could mean low value and `HV` implying high value). Each component is associated with at least a port which shows connection to a component or between two components. Hence below is a valid failure logic expression.

```
O-out1 = InternalFailure OR O-in1
```

Where:`O-out1` is the *Output Deviation* at output port 1 of the component

`InternalFailure` is the *Internal Failure* of the component

`O-in1` is the *Input Deviation* at input port 1 of the component

`OR` is the Boolean logic for the failure expression of the component

The above failure logic expression implies that the cause of omission of output at port `out1` is caused by either the internal failure of the component or omission of input at port `in1`. Deviations may as well contain parameters that convey properties of a given port. Port properties are appended to a deviation and prefixed by a hyphen. For instance, the failure logic expression below implies that the cause of omission of output at port `out1` is caused by either omission of input at port `in1` or high value of voltage at port `in1`. In this case *voltage* is the parameter.

```
O-out1 = O-in1 OR HV-in1-Voltage
```

Given the local specifications of component failure in expressions such as the above, HiP-HOPS can create and evaluate fault trees and FMEAs and do some architecture optimisation. The local failure modelling helps to focus and systematise analysis while the tool undertakes part of the effort of linking analyses at systems level. To address situations where failures need to be seen at system level by analysts, HiP-HOPS incorporates mechanisms for scoping, hierarchical annotation of models and zonal analyses.

2.1.2.2 STPA

STPA (**ST**am**P** **A**nalysis) is a safety analysis technique based on the STAMP model of accident causation (Leveson and Dulac, 2005, and Herring et al, 2007). Thus, STAMP is here first introduced.

The STAMP model is a dependability (specifically safety) analysis technique that views system failure in a different form as opposed to traditional techniques. The criticism of traditional techniques has been that because of their dependence on failure events, they neither do a good job of handling software nor system accidents where the losses stem from dysfunctional interactions among operating components rather than failure of individual components (Leveson, 2003).

In STAMP, safety is viewed as a control problem; accidents occur when component failures, external disturbances, and/or dysfunctional interactions among system components are not adequately handled or controlled (Herring et al, 2007). This relates to both system design and operation level. The challenge to the engineer using this technique is to adequately identify safety-related constraints and to ensure that they are imposed on the system at both design and operation levels. The technique also emphasises fault tolerance; a system must not only ensure enforcement of safety-related constraints, but should also be a dynamic system such that any change due to breach of constraint is adapted and the system channelled to still operate safely while accomplishing a given mission.

It is also possible to maintain the safety of a system through a controller while in operation. A controller is a dynamic entity that repeatedly monitors the parameters involved in ensuring successful system operation. Such a controller as modelled by STAMP could either be manual (human) or automated (computer), or combination of both where the manual oversees the automated.

By focusing on safety-related constraints, Leveson claims that STAMP's approach to safety analysis may be viewed as deviation from traditional safety analysis. This is however arguable at least in the case of hazard analysis as in Leveson (2003); STAMP hazard analysis has the same general goals as any hazard analysis, i.e. (i) identification of the system hazards and the safety constraints necessary to ensure acceptable risk, and (ii) accumulation of information about how those constraints could be violated to use for eliminating, reducing, and controlling hazards in the system design and operations.

According to Leveson and Dulac (2005), STPA is a new STAMP-based system analysis technique, which starts at the early life cycle stages and continues through the life of the system. Its use during design can support a safety-driven design process where the hazard analysis influences and shapes the early design decisions.

Usually, at the early design life of a system, there is little information available to the engineer and as design decisions are made, the use of STPA in hazard analysis helps to unravel information surrounding such decisions. At such early design stage where information is little, the analysis of hazard using STPA will be very general at first and will be refined and augmented as additional information emerges from the system design activities (Leveson and Dulac, 2005). Hazard analysis using STPA starts by defining an initial hierarchical control structure for the given system. A detailed description can be found in Leveson and Dulac (2005), and Herring et al (2007).

2.2 System Maintainability

Maintenance increases the life of components and the system, and is therefore an important tool for improving dependability but also performance and cost. Maintainability engineering is concerned with the efficiency and effectiveness of maintenance. Strictly speaking maintainability is a design consideration while maintenance is the consequence of that design. Birolini (2007) defines "*maintainability as a characteristic of an item, expressed by the probability that preventive maintenance or repair of the item will be performed within a stated time interval by given procedures and resources"*.

The physical feature of a component (e.g. packaging, mounting, etc) may affect the ease and speed at which maintenance can be performed. As is the case with reliability, maintainability should be built into components at their design stages. This is because maintainability cannot be easily predicted and a maintainability improvement often requires important changes in layout or construction of the item (component) considered (Birolini, 2007).

In this work, *maintainability is defined in simple terms as the probability that a system component will be retained in, or restored to, a specified condition within a period of time when maintenance is performed in accordance with prescribed procedures*.

2.2.1 Maintenance as a Factor in Safety

Maintenance refers to the action taken to retain or restore a system to its designed condition (Storey, 1996). The definition can be seen to consist of two parts; (i) to retain implies that maintenance actions are performed before failure in order to keep the system operable on demand, and (ii) to restore means to return the system back to its operable condition after failure has occurred. The former is referred to as preventive maintenance and the latter as corrective maintenance. In order to keep safety alive and to improve reliability and availability of a system, preventive maintenance is therefore preferred. Thus, the rest of this chapter focuses on preventive maintenance (PM).

A well articulated PM policy may restore a component to as new as originally designed or somewhere close. This will however, depend on several factors like:

- the level of damage to, or wear and tear of, the component
- the technical expertise of the maintenance personnel or crew
- type of tool/technology used for the maintenance

Poor maintenance practices will induce failure to a component or other components, and consequently the system. Hence it should be emphasised that maintenance should be carried out by skilled personnel and in accordance with prescribed procedures.

System components are characterised by *mean time to failure* (MTTF) and, or *mean time between failures* (MTBF). MTTF is useful in scenarios where repair is neither possible nor considered. MTBF on the other hand is useful when repair is possible or considered. Within these average time failures, a given component is expected to operate successfully. However, the component can fail at any time because of the randomness of hardware failures which typically follow probabilistic distributions. It is therefore typically required that components of safety-critical systems are maintained, where possible, in order to decrease likelihood of failure and of any possible catastrophic effects of such failure.

A catastrophic effect of failure does not only imply loss of human life but also economic loss. Hence, high maintainability of a component may be seen as paramount, for instance in production industries; where a production plant may need to be shutdown for an unplanned maintenance. High maintainability will imply that a given component can be maintained easily and fast, thereby restoring the plant to its operation and subsequently minimising the effects of loss.

In a similar scenario, the failure of the braking system of an automobile while in operation may be catastrophic depending on certain parameters like the speed, environment, etc. The occurrence of such failure can be reduced through the actions of maintenance. The fact that maintenance has the possibility of preventing the occurrence

of failure or reducing the effects of loss, infers that maintenance is paramount in keeping alive the safety of systems.

2.2.2 Segments of System Downtime

Downtime refers to the period when a component or system is out of operation due to either failure or planned maintenance (Ebeling, 1997). It consists of several segments as seen in Ebeling (1997) and is shown in Figure 2.2. The downtime segments are also briefly discussed.

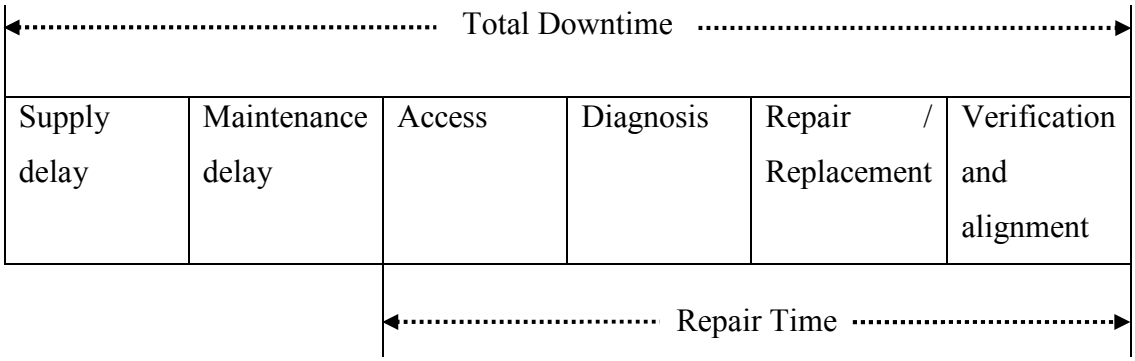

Figure 2.2 - Segments of system downtime

The *supply delay* segment is the time taken in obtaining the necessary component required for the repair. This may involve things like administrative time (i.e. forwarding the complaint, authorizing the repair, etc), time taken for spare part procurement (i.e. making and receiving the order or reaching out to procuring the spare part). The supply delay segment may not necessarily be the first repair process but may come after diagnosis. If a spare part is readily in place when failure occurs, this time may be negligible.

Maintenance delay is the time taken in obtaining maintenance resources and facilities. Resources include maintenance personnel, tools, manuals and other materials required to carry out the maintenance. Facilities may include repair or service garage.

Access is the time taken in reaching to the failed component within the system. This may include tasks like removing a cover, bracket, panel, etc before reaching the failed component.

The *diagnosis* - also referred to as troubleshooting - is the time allocated to finding out what caused the failure.

The *Repair or Replacement* segment is the time taken in carrying out the actual repair, replacement or servicing of the component.

Verification and alignment segment is dedicated to making sure that the restored component is operating satisfactorily.

The repair time as seen in Figure 2.2 is a parameter that is used in the evaluation of the availability and cost of a system where repair is possible. The repair time consists of four segments (access, diagnosis, repair/replacement and, verification and alignment). The supply and maintenance delay times are purely affected by external factors. The repair time can be greatly reduced if the component is designed and packaged well for easy access and disassemble.

2.2.3 Maintenance Models

Maintenance models describe the effects of maintenance actions on a component based on a selected parameter. Such a parameter could be for instance component *age* or *shock (damage) level* which measure deterioration of the component. *Age reduction models* focus on the "effective age" of components following maintenance actions, while *shock* models focus on the level of component damage or shock. Two models (PAR and PAS) belonging to the age reduction model, and the shock model are discussed below.

2.2.3.1 PAR Model

The PAR (proportional age reduction) model assumes that each maintenance activity reduces proportionally the age gained from previous maintenance (Sanchez et al, 2009). This simply means that each PM activity is assumed to only reduce a portion of the component age. This proportional age reduction is dependent on an *improvement factor f*, where *f* lies between 0 and 1, inclusive ($0 \leq f \leq 1$).

According to Tsai et al (2001, eq 5) and Sanchez et al (2009, eq 9) a maintenance activity conducted at the *i-th* time t_i, with an improvement factor f_i for a given component of a system reduces the component age W_i as shown in equation 2.1.

$$W_i^+ = (1 - f_i)t_i \tag{2.1}$$

Where t_i is the time at which the *i-th* maintenance is carried out and the plus sign symbolises that the effect of age reduction applies only after the PM activity. W_i^+ is known as the *effective age* of the component.

From equation 2.1, it is obvious that when the improvement factor f_i is of value 1, the effective age of the component becomes 0, and the component's condition is known to be *good-as-new* (GAN) and if f_i is of value 0, the component's condition has not improved in which case it is known to be *bad-as-old* (BAO) (Martorell et al, 1999).

2.2.3.2 PAS Model

The PAS (*proportional age setback*) model assumes that each maintenance activity shifts the origin of time from which the age of the component is evaluated (Martorell et al, 1999). This model also uses the concept of *improvement factor f* and assumes that the maintenance activity reduces proportionally in a factor of *f* the age of the component at the time of maintenance. Just as in the case of PAR, when $f = 0$, the component assumes BAO and when $f = 1$, it assumes GAN. However, PAS model has a generalization that each maintenance activity reduces the component's entire age.

The age of a component after *n* PM stage according to Sanchez et al (2009, eq 1) is given as in equation 2.2.

$$W_n^+ = t_n - \sum_{i=1}^{n-1} (1 - f_i)^i f_i t_{n-i} \tag{2.2}$$

2.2.3.3 Shock Model

The shock model was proposed by Kijima and Nakagawa and the concept described here is found in Pham and Wang (1996).

Considering a component which is subjected to shocks occurring randomly, at time t = 0, the damage level of the component is assumed to be 0. When the component experiences a shock, the component suffers a non-negative random damage (incremental damage). Each shock contributes to the current damage level of the component and the value is constant between shocks. The component then fails when the accumulated damage level exceeds a specified level. In order to keep the component functional, a maintenance intervention is necessary.

According to the proposed cumulative damage shock model by Kijima and Nakagawa (Pham and Wang, 1996), each PM intervention reduces the damage level by a factor of *100(1-b)%, 0 ≤ b ≤ 1*. It can therefore be observed that if *b = 1*, the PM intervention results into a PM that assumes BAO, and if b = 0, it assumes GAN. In a later work, Kijima and Nakagawa established that the level of damage after the *k-th* PM stage is $b_k Y_k$. Where Y_k is its damage level before the *k-th* PM stage and b_k is the *improvement factor* at the *k-th* PM stage.

2.2.4 Improvement Factor Assessment

The assessment or estimation of the improvement factor for a component is one of the parameters of uncertainty in maintenance. This work adopts Malik's proposal on

improvement factor, also adopted and subsequently modelled by Tsai et al (2001) and is described as follows.

The number and type of maintenance activities earmarked upon for a component depends on its design characteristics. The following are possible maintenance activities that may be performed on a component: *lubricating*, *cleaning*, *tightening*, *adjusting*, *topping* and *simple repair* (e.g. replacement of seals and rings). Most literatures refer to simple repair as *minimal repair*, and this name will be adopted from now onwards. The assessment of improvement factor depends on the probability of performing each maintenance activity and their respective improvement level. The *probability of performing a particular maintenance activity* and the *improvement level* for component *i* are represented by p_{ij} and d_{ij} respectively; *i* is and index representing the *i-th* component and *j* is the index of the *j-th* activity.

According to Tsai et al (2001, eq 7), the improvement factor for a component under *k* number of maintenance activities is calculated as shown in equation 2.3.

$$f_i = \frac{1}{P_{S_i}} \sum_{j=1}^{k} p_{ij} d_{ij} \quad ; \quad 0 \le p_{ij} \le 1, \quad 0 \le d_{ij} \le 1 \tag{2.3}$$

Where:

$$P_{S_i} = \sum_{j=1}^{k} p_{ij} \tag{2.4}$$

2.2.5 Advances in System Maintenance

Research has recently focused on system maintenance with focus on optimisation. Some of the most relevant works are discussed below.

Tsai et al (2001) investigated the optimal activities-combination that maximises system unit-cost life at each PM stage. The defined activities are *1P* – simple preventive

maintenance which changes the system reliability to some newer point, and *2P* – preventive replacement which restores the system reliability to complete new. The proportional age reduction (PAR) model was used in modelling the effects of maintenance on system reliability. In summary, the approach searches for activities-combination that can be performed on respective components of a system at each PM stage. The activities-combination refers to *1P* and *2P*, and the challenge is to establish which one will be performed at a given PM stage for each component. The PM interval begins with a base value I_b through to a maximum value I_m, using a step value I_s. The search finds an optimal PM interval I_o; $I_b \leq I_o \leq I_m$, where I_o is the most suitable PM interval for the system. The activities-combination is determined using genetic algorithm by pursuing unit-cost life maximisation. The system reliability evaluation method assumes that a single component failure can cause the system to break down and therefore components are combined in series.

Artana and Ishida (2002) investigated the optimisation of maintenance schedules by considering components in wear-out-phase using a spreadsheet-modelling tool. As it is in most maintenance scheduling optimisation problems, the strategy involves obtaining optimum PM intervals for components whilst minimising total cost. The cost evaluation takes into account maintenance cost, operational cost, downtime cost and penalty cost. Constraints imposed on the optimisation are reliability and availability indexes at system level. Each index is a boundary consisting of lower and upper limits. A component is replaced when it can no longer attain the lower limit of reliability and availability.

Bris et al (2003) investigated the maintenance problem by considering a series-parallel system and employed genetic algorithm for optimising PM schedules using Matlab as a tool. The maintenance model used was that which returns a component to as good-as-new. The optimisation aims at minimising cost based on a given availability constraint; $A(t) \geq A_0$, $0 < t \leq T_M$, where A_0 is the lower limit availability and T_M is mission time. The approach to the problem also involves finding the solution vector T_v of system component inspection periods which itself is dependent on finding the optimal first inspection time vector T_0. In a nutshell, this investigation obtains the optimal system

inspection times satisfying a constraint imposed by targeting requirement for minimum system availability.

Sheu et al (2006) investigated the problem of maximising availability of a repairable system under preventive maintenance. The repairable system considered has three different maintenance policies; imperfect preventive maintenance, perfect preventive maintenance and failed preventive maintenance. At imperfect preventive maintenance stage, the failure rate of the system is considered to be same as prior to the PM stage. Perfect preventive maintenance stage improves the system to as good-as-new. A failed preventive maintenance occurs when major repair exists after which the system returns to good-as-new.

Castro (2009) investigated a model of imperfect preventive maintenance with dependent failure modes. Two failure modes were identified; *maintainable* and *non-maintainable* failure mode. The former refers to the kind of failure for which its likelihood can be reduced through preventive maintenance activities (oiling, cleaning, minimal repair, etc) whereas the latter refers to the failure that is related to the inherent design of the component. The failure rate of the maintainable failures is assumed to depend on the total non-maintainable failures since installation of the component. The concept behind the assumption is that the non-maintainable failures are as a result of wear-out portions of the component for which maintenance actions can not reach or affect but contribute to the failure of the component. Castro showed this dependency by modelling the maintainable failure rate of a component to take into account its non-maintainable failure rate. The optimisation problem is to determine the optimal length between preventive maintenances and the total number of preventive maintenances before component replacement. The optimisation was formulated to minimise expected cost rate and aimed at finding an optimal PM interval T and the total number of possible PM stages N where replacement is carried out at the N-*th* PM stage.

Lust et al (2009) investigated the maintenance of a system with time-window. A system classified with time-window is one that performs sequence of missions and is maintained in between missions. Systems falling under this category are military and

production systems. The time duration in-between end of mission and start of the next mission is what is termed as *time-window*. The problem is to find the best choice of maintenance actions to be performed on a multi-component system, so as to maximise the system reliability and also to improve availability. A typical time-window has small duration, and the idea is to pick a subset of maintenance actions which fit into the time-window and to yield improved reliability when the system is restarted. Two maintenance actions defined by Lust et al (2009) are (i) replacement - which brings component age to 0 after maintenance (GAN), and (ii) minimal repair - which subjects the age of the component to BAO (unchanged) after repairing a failed component. Budget was not considered as part of the problem following the assumption that cost of planned maintenance is negligible compared to cost of unplanned maintenance. Therefore, the optimisation problem was based on reliability and time-window. The approach uses Tabu search technique and Weibull distribution for estimating the reliability of the components and eventually of the system. The system model was based on a series-parallel arrangement.

Owing to the numerous existing maintenance models, coupled with lack of standardised maintenance framework (Márquez, 2007), previous works in system maintenance offered maintenance policies in different ways and approaches.

2.2.5.1 Limitation of Work on Optimisation of Maintenance

Most of the earlier work on evaluation and optimisation of system maintenance assumes that the system is series-parallel arrangement of components. This simplifying assumption makes reliability and availability evaluations possible using reliability block diagrams (RBDs). RBDs offer a simple and quick method for evaluating system reliability and availability. However, the model becomes problematic when faced with the complexity of the architectures of modern systems. Some limitations are discussed below:

- Systems are typically composed of subsystems with hierarchies of components and many connections to and from each component. These connections often

violate the series parallel assumption and create bridges and complex network configurations.

- The failure behaviour of such architectures cannot be accurately described using RBDs. The RBD is a diagram which is constructed by answering the following questions; which components of the system under consideration are necessary for the fulfilment of the required function and which can fail without affecting it (Birolini, 2007). The necessary ones are arranged in series otherwise in parallel. Hence the RBD is a simplification of the original system model and is confined to series-parallel arrangement
- While RBD allows for only two failure states (success and failure), in reality a given component may have several failure modes (Parker 2010), which may include the omission of function but also incorrect delivery in terms of value and time.
- The architecture of the model of a system may be modified in order to improve dependability and cost following the results of optimisation. In this case a new RBD must be constructed to reflect the new failure behaviour of the system. This creates difficulties and makes the process slow, error prone and inefficient. Ideally some kind of automation would allow updating of the failure model (RDB) when the architecture is naturally modified in the course of the evolution of a system.

2.2.5.2 Innovative Approach to Maintenance Problem

To overcome the limitation of earlier work on evaluation and optimisation of maintenance this work proposes:

a) Use of HiP-HOPS - a contemporary dependability analysis technique that overcomes the limitations of RBDs outlined above. HiP-HOPS can deal with complex networked architectures and multiple failure modes.

b) A novel combination of HiP-HOPS with Genetic Algorithms which enables automatic exploration of the typically enormous space that defines the

possibilities for maintenance policies and schedules in a system. The genetic algorithm can look within this space for optimal schedules that can achieve dependability requirements with minimal costs.

This work attempts to close the existing gaps in optimising maintenance schedules in the following way.

Case i - evaluation of system design

- Establish system component models for both perfect and imperfect preventive maintenance policies for the following, using the proportional age reduction (PAR) model:
  - reliability
  - availability (and subsequently unavailability)
  - cost
- The use of minimal cut sets synthesised from HiP-HOPS analysis to evaluating system reliability and availability/unavailability. This makes the dynamic effects of maintenance possible on systems with hierarchical structures
- Extending HiP-HOPS with capabilities for evaluating system reliability, availability and cost in scenarios where maintenance is possible

Case ii – maintenance optimisation strategy:

- Establishing approaches to optimising preventive maintenance schedules using genetic algorithms under the following constraints
  (a) Primary constraints
    - Constraint on system's shortest PM interval
    - Constraint on component PM time
  (b) Secondary constraints
    - Expert judgement
    - Architecture modification through component substitution
  (c) Composite constraint
    - A combination of primary and secondary constraints

The next chapter focuses on the problem of optimisation and explores various methods for optimisation that were considered in the context of this work.

3 BACKGROUND II - OPTIMISING SYSTEM DESIGN

A system design can have many possibilities for implementation even when the architecture is fixed and the only variability lies in the potential implementations of each component. The design space constituting such variants of the design can rapidly become enormous as the number of components grows. In situations where component replication and redundancy are possible in various locations of the architecture the design space grows further. In such cases, the concern of the engineer is to search within this design space for solutions that achieve design objectives in an optimal or near optimal way. Finding those solutions is an optimisation problem.

Not only architectural transformations like the ones described above, but also maintenance schedules can be used to improve the performance of design objectives like dependability and cost. Components of a system might be maintained according to a different time schedule each and, therefore, overall a system can have numerous potentially feasible PM schedules. Once more the concern for the engineer here is to find PM schedules that best achieve the design objectives of the system.

In general, an optimisation problem is modelled as shown below (Gen and Cheng, 1997 and Konak et al, 2006). A more specific definition is discussed in section 3.2.

$max\ \mathbf{F}(\boldsymbol{x})$ ..... (i)

such that:

$g_i(\boldsymbol{x}) \leq b_i \quad i = 1..k;$ ..... (ii)

Where $\boldsymbol{x}$ is an m-dimensional vector known as decision variable vector, i.e. $\boldsymbol{x} = \{ x_1, x_2, .., x_m \}$ and $\boldsymbol{x} \in \boldsymbol{X}$, where $\mathbf{X}$ is the solution space and $\mathbf{F}(\boldsymbol{x})$ is known as decision vector. Each $x_i \in \boldsymbol{x}$ is referred to as a decision variable. The left hand side of the constraint; $g_i(\boldsymbol{x})$ is a real value function, whereas b_i could either be a predefined value or the result of another real value function.

$\mathbf{F}(\boldsymbol{x})$ in equation (i) above is also referred to as the objective functions (also known as criterion functions). The goal of the optimisation is the maximisation (max) or

minimisation (min) of these functions. The objective functions are attributes of the design and normally include *cost* and one or more of the following: *reliability*, *availability*, *safety*, *weight*, etc.

Equation (ii) is known as the inequality constraint. If this is in the form $g_i(\boldsymbol{x}) = b_j$, then it is referred to as equality constraint (Gen and Cheng, 1997). When a constraint is present, the optimisation must conform to it. A solution $\boldsymbol{x} \in \boldsymbol{X}$ which satisfies the constraint is known as a feasible solution. A collection of all potential feasible solutions defines the feasible region.

3.1 Single Objective System

A single objective system is one in which there is only a single objective function. For instance the design objective may just be to maximise reliability. However in real life problems, systems design comprise of several objectives. Multiple objectives can be combined into a single objective via a weighting approach, see for example the approach discussed in section 3.2.1.

3.2 Multi-Objective System

A multi-objective system is one in which there are multiple objective functions and these are treated separately. It is a stretched form of the general optimisation modelling and is expressed below as found also in Huang et al (2005) and Konak et al (2006).

$max\ \mathbf{F}(\boldsymbol{x}) = \{ f_1(\boldsymbol{x}), f_2(\boldsymbol{x}), f_3(\boldsymbol{x}), .., f_m(\boldsymbol{x}) \}$

such that:

$\boldsymbol{x} \in \boldsymbol{X}$

$g_i(\boldsymbol{x}) \leq b_i \qquad i = 1..k$

Where: $f_1, f_2, f_3, \ldots, f_m$ are objective functions

In a situation where not all of the objective functions are subjected to the same goal for optimisation (i.e. minimisation or maximisation) then the following techniques may be used in converting from one to the other.

(i) Option one - inversion approach

$$f_i^{'} = \begin{cases} \frac{1}{f_i} & (converting\ to \min f_i) \\ \frac{1}{(1 + f_i)} & (converting\ to \max f_i) \end{cases}$$

(ii) Option two - negation approach

$$f_i^{'} = \begin{cases} -1.(f_i) & converting\ either\ way \end{cases}$$

To optimise a system model, one of two basic approaches is employed. These are discussed next.

3.2.1 Single-Objective Approach to Multi-Objective System

In this approach, multi-objectives are first turned into a single-objective which is then optimised. This is done in either of two ways (Konak et al, 2006) described below.

The first approach converts all but one objective into a constraint. For instance if the objective functions are reliability, weight and cost, the optimisation problem can be modelled as:

$max\ \mathbf{F(x)} = \{ R(\mathbf{x}) \}$
such that:
$W(\mathbf{x}) \leq b_w$
$C(\mathbf{x}) \leq b_c$

Where: R is system reliability, W is system weight and C is system cost, b_w and b_c are real target values for weight and cost respectively.

In this case, weight and cost are transformed into constraints while keeping reliability as the single objective function. The constraints formulation largely depends on the system design requirements.

The second approach is what is called a *weighted sum*, where all the objective functions are combined into a single composite function. The modelling for a weighted sum approach is as shown below (Huang et al, 2005).

$$max\ f = \sum_{i=1}^{k} w_i f_i(\mathbf{x})$$

Where: w_i is the weight of the *i-th* objective function; $w_i > 0;\ i = 1..k$

f_i is the *i-th* objective function

A drawback of this approach is that there is no formalised way to assigning the objective function weightings. The assignment depends on engineering judgement.

3.2.2 Pareto Optimality Approach to Multi-Objective System

The Pareto optimality approach returns a set of optimal solutions that are trade-offs among the objective functions. This is quite different from the previous approach discussed, in that the single-objective approach returns a single solution rather than a set of solutions that are trade-offs (Konak et al, 2006). An optimal solution is a non-dominated solution which is formally defined as follows (Weise, 2008).

$$x_1 \vdash x_2 \Leftrightarrow \forall i{:}\mathbb{N} \mid 0 < i \leq n \bullet w_i f_i(x_1) \geq w_i f_i(x_2) \wedge \exists j{:}\mathbb{N} \mid 0 < j \leq n \bullet w_j f_j(x_1) > w_j f_j(x_2)$$

$$w_i, w_j = \begin{cases} -1, \text{if } f_i, f_j \text{ are to be minimised} \\ 1, \text{if } f_i, f_j \text{ are to be maximised} \end{cases}$$

Where: the symbol ⊢ means dominate

In basic terms, a non-dominated solution is one which is better than all other solutions in at least one objective function and not worse of in any other objective function. The set of non-dominated set within the feasible region is referred to as *Pareto optimal set*, and each element within this set is referred to as *Pareto front*. The term *Pareto front* is also used in referring to a point which is optimal in objective functions space relative to already existing optimal solution(s). The term *Pareto frontier* is also used to refer to Pareto optimal set (Huang et al, 2005). Pareto optimal set can be of varied sizes, however the size usually increases with increase in the number of objective functions (Konak et al, 2006).

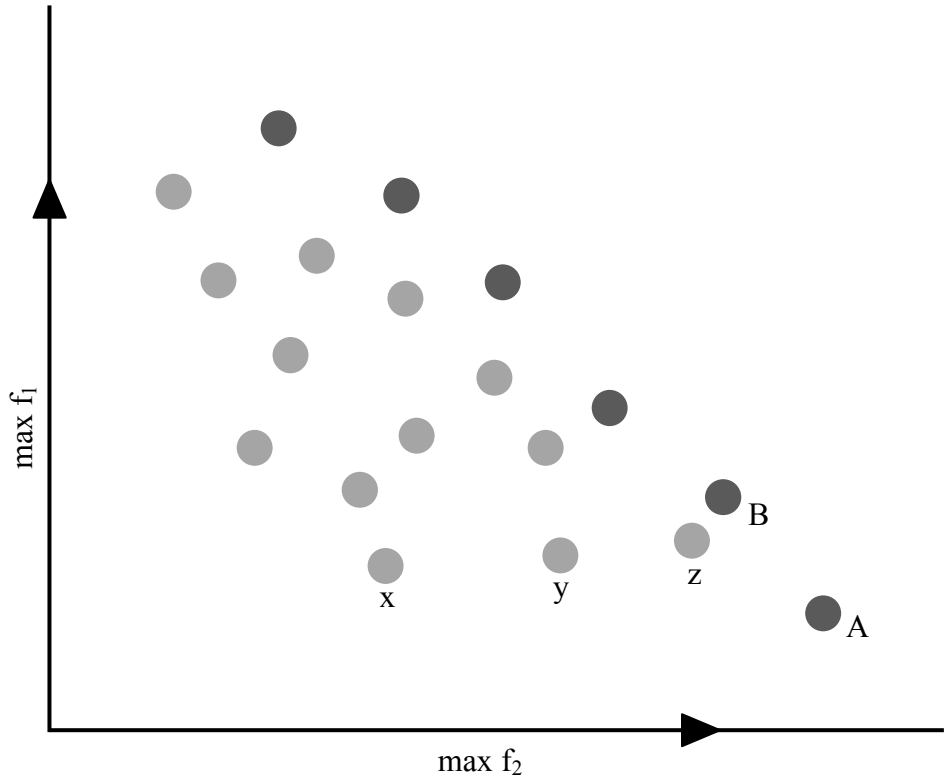

Figure 3.1 - Pareto optimality

As an illustration, supposing F is a 2-order objective function defined as *max* $F = \{f_1, f_2\}$, Figure 3.1 is an arbitrary graph demonstrating the notion of Pareto optimality. It should be noted that the arrow direction on the axes away from origin symbolizes

maximisation. If an objective function is to be minimised, then the arrow is directed towards origin.

The dark and light grey circles are all solutions within the feasible region; however, all the darker circles form the Pareto optimal set. Figure 3.2 demonstrates how to visualize the act of dominance imposed by the Pareto optimal set.

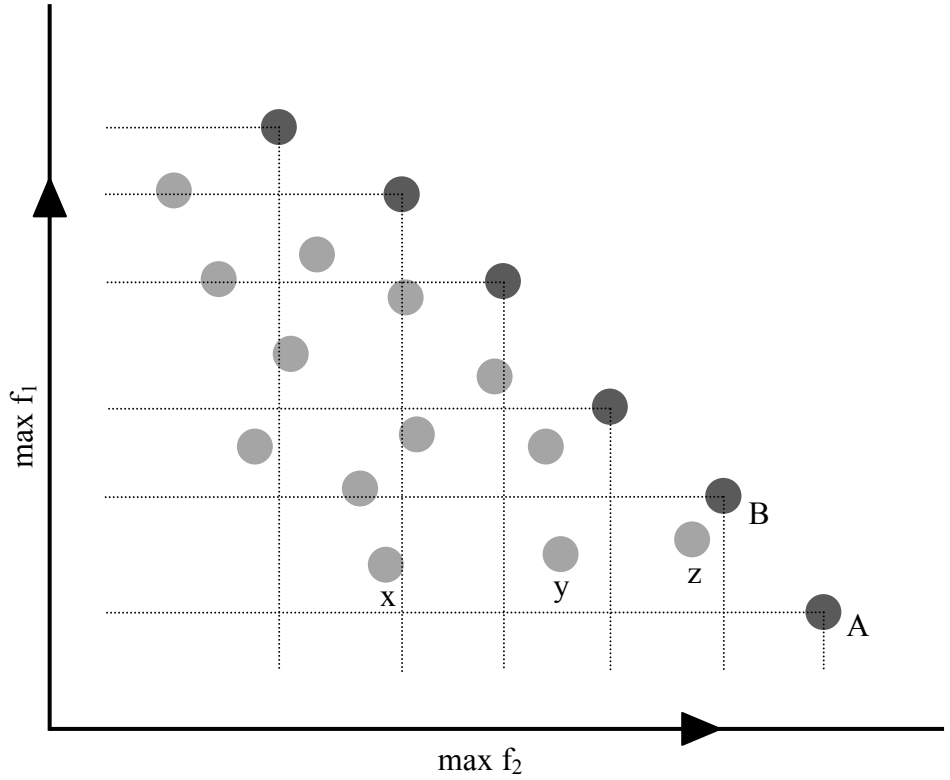

Figure 3.2 - Demonstration of the act of dominance

The act of dominance can be visualised with the help of the dotted lines emerging from each Pareto front. Any solution enclosed within the dotted lines is dominated by the solution from where these lines emanate. For instance B dominates x, y, and z. On the other hand, solution A dominates no solution, but is not itself dominated by any and therefore is a member of the Pareto optimal set. It is clear from the graph that moving from one solution on the Pareto frontier to the other infers compromising improvement in the other objective function. For a system design, an engineer will select one of the

Pareto front solutions based on requirements; optimisation is therefore crucial in the early design stage of a system.

3.3 Search Techniques

The aim of optimising a system is to find a Pareto optimal set of designs. This Pareto optimal set is a subset of the feasible region, and similarly the feasible region is a subset of the solution space. This entails that the Pareto optimal set needs to be searched for, either manually or with the help of automation. Several search techniques exist for solving optimisation problems, few of which are discussed below.

3.3.1 Hill Climbing

In this approach the solution space can be imagined as a landscape where an optimal solution is represented by the top of the highest hill. Supposing a viewer wants to establish a location to properly view a landscape within a particular land area, the best location will be the top of the highest hill. The viewer begins the search for this best location from an initial position, P_0 and progresses to nearby position or neighbourhood by taking a step P_{0+1}. Hence at the *i-th* position, the next neighbourhood would be P_{i+1}. A guided search will be to compare the current location P_i with P_{i+1}, such that if the height of the latter is higher, then it replaces the current. If this works well, the viewer will eventually progress to the highest hill top where the best perspective of the landscape is possible.

However, there might be occurrence of local optima, where the search may get stuck. Local optima are elevated heights within the selected land area for the search, where none of them is the best height. For this reason, hill climbing is generally categorised as a local search technique.

The concept of hill climbing is about attaining an optimal height, and for this reason an objective function value will be referred here in the context of hill climbing as *height*. The algorithm for hill climbing from Hart (2006) and, MacFarlane and Tuson (2009) is described as follows.

i. Generate an initial solution S_i in the search space **X**

ii. Apply a move operator to evolve with a solution S_{i+1} within the neighbourhood of S_i

iii. Evaluate the fitness (height) of S_{i+1}

iv. If the height of $S_{i+1} > S_i$ then $S_i = S_{i+1}$

v. Return to step ii until termination criterion applies

The termination criterion may be certain number of CPU elapsed time, number of iterations, user request for termination, etc as it suites the problem.

Like any other search technique, the effectiveness of hill climbing is problem dependent. Among other areas where hill climbing has been applied are information retrieval problems (MacFarlane and Tuson, 2009) and in security systems; online signature verification (Muramatsu, 2008). However, hill climbing is generally effective where there are few local optima. According to Hart (2006), hill climbing is also very easy to implement and to give fairly good solutions very quickly. The concept of hill climbing may be viewed from Figure 3.3 below.

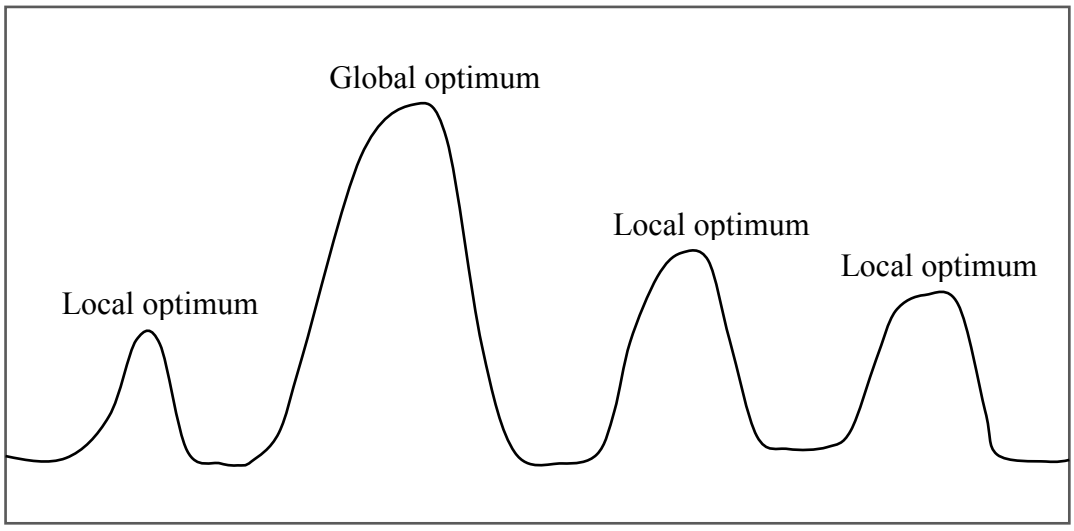

Figure 3.3 - Overview of solution space for hill climbing problem

It can be seen that within the solution space (enclosed within the rectangle), there are three local optima. The desired solution will be to have a solution that is a global optimum.

In system design terms, the hill climbing search is not a search for better location to viewing a landscape but a search for a design solution that optimises the objective functions. This illustration can be described using the chess board in Figure 3.4. The chess board represents the feasible region where each location (square piece) can represent a PM schedule. In practice the size of this feasible region may be larger however this is sufficient for the purpose of illustration. Assuming that the queen (which in this case is the searchlight) is currently on location $P_{i,j}$ where i is row and j column with initial values 4 and 5 respectively. In this illustration unlike in real chess game, the queen is restricted to one step movement at a time, therefore the locations marked "*x*" on the board defines possible moves.

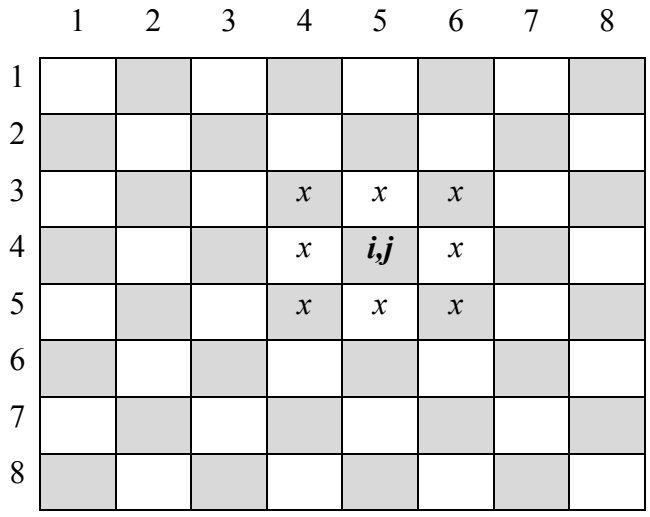

Figure 3.4 - Illustration of hill climbing using a chess board

One of challenges in hill climbing is in defining the move operator to meet the optimisation problem. In this scenario, the move operator could be defined to allow for any of the set of movements M_s (seen below) and in addition to boundary checks, etc.

M_s = { {i+1,j-1}, {i, j-1}, {i-1,j-1}, {i-1,j}, {i-1,j+1}, {i,j+1}, {i+1, j+1}, {i+1, j} }.

If any of these movements results into a better solution, then the current i,j shifts to the new location of better solution. This process continues until termination criterion is reached.

As mentioned earlier, a hill climbing search may get stuck in a local optimum, for instance one of those in Figure 3.3. One way to prevent this is to use an extended form of hill climbing known as *iterated hill climbing*. The concept behind this extended technique is to restart the hill climbing with a different initial solution S_0 when local optima has, or is suspected to have been found (MacFarlane and Tuson, 2009). The challenge here is to draw a condition by which local optima may be identified. One way to do this is when there appears to be no substantial solution improvement after a preset number of searches.

Another extended form of hill climbing that attempts to overcome the problem of local optima is *smart hill climbing*. Its algorithm consists of two main phases, a global and a local search phase. The goal of the global search phase is to cover the search space as broadly as possible in order to identify a good start for the local search phase. The local search phase then starts from the point returned by the global search, and then searches around its neighbourhood for a better solution (Xi et al, 2004).

3.3.2 Simulated Annealing

Simulated annealing is a search technique that is based on the analogy between optimisation and the annealing process within the branch of *Physics* known as *thermodynamics* (Fleischer, 1995, MacFarlane and Tuson, 2009). It reduces the chance of a search getting stuck in a local optimum by allowing moves to inferior solutions (Hart, 2006). An element such as metal has high chance of taking any desired shape when it is heated under high temperature. The metallic element becomes stronger and stiff as it cools. Using this analogy, a solution search at high temperature may be flexible, and hence accepting poor quality solutions with the aim that they may become better as the temperature slowly cools down. Solution acceptance grows in rigidity as

the temperature cools down. Therefore, two crucial criteria need to be established for optimising systems using simulated annealing:

- The probability by which poor quality solutions are accepted
- The choice of initial and final temperature, and the cooling schedule by which the temperature decreases as the search progresses

The probability by which poor solutions are accepted is given by equation 3.1 below (Correia et al, 2001, eq 17, Hart, 2006, MacFarlane and Tuson, 2009, eq 1).

$$P(T_k) = e^{-(\frac{\Delta quality}{T_k})} \tag{3.1}$$

Where: $\Delta quality$ is the change in quality between the current solution S_C and new solution S_N; quality = S_N - S_C, T_k is the temperature at the *k-th* time-step and $P(T_k)$ is the probability at time T_k

A solution is accepted if $P(T_k) > R$, where R is a uniform random number between 0 and 1 (Hart, 2006). The term uniform implies that same value of R is considered throughout the search. The initial system temperature T_0 must be set high enough so that at initial stage all states proposed are accepted (Correia et al, 2001). According to MacFarlane and Tuson (2009) there is no reason why the cooling schedule should be of any particular form, or even monotonically decreasing - the choice is problem-dependent. However, a commonly known cooling schedule that generally works well is presented below in equation 3.2 (Lundy and Mees, 1986 cited MacFarlane and Tuson, 2009, eq 2, Correia et al, 2001, eq 19).

$$T_{k+1} = \beta T_k \tag{3.2}$$

T_{k+1} and T_k are system temperatures at k and $k+1$ successive iterations and β is the cooling parameter usually taken in the range 0.8 to 0.95 (Correia et al, 2001).

A simulated annealing algorithm from Hart (2006) and, Wei-zhong and Xi-Gang (2009) for a maximisation optimisation problem is as follows.

i. Initialise k = 0
ii. Set an initial system temperature T_0
iii. Generate an initial solution, call it the current solution S_C and evaluate it
iv. Apply a move operator on S_C to get a new solution S_N and evaluate it.
v. If the new solution is better than the current solution ($S_N > S_C$) then $S_C = S_N$
vi. If it is worse, accept S_N ($S_C = S_N$) with probability described in equation 3.1
vii. If stop criterion applies, then stop, otherwise $T_{k+1} = \beta T_k$, k= k+1 and return to step iv

3.3.3 Genetic Algorithm

The Genetic algorithm (GA) is a search technique that draws its analogy from natural selection (Marseguerra et al, 2006, MacFarlane and Tuson, 2009) and aims at finding the global maximum or minimum of a given real objective function f($\boldsymbol{x}$) subjected to possibly one or more constraints g($\boldsymbol{x}$) (Marseguerra et al, 2006). This natural selection is based on evolutionary theory on how species evolve. In a natural ecosystem, species that are weak, in other word unfit, are brought to extinction through natural selection. The fitter organisms within a specified region of the ecosystem have higher chances of passing their genes to the next generation through reproduction. As reproduction progresses over generations, the population will eventually comprise of species that are dominant (best fit).

Genetic algorithms use terms borrowed from *Biology*. In general, the decision variable vector $\boldsymbol{x} \in \mathbf{X}$ is termed as chromosome or individual (Konak et al, 2006). Where $\mathbf{X}$ as mentioned earlier is the solution space, a decision variable $x_i \in \boldsymbol{x}$ is termed as gene, implying a discrete element of the chromosome. Thus, in system design, a chromosome can represent a variant of the model or PM schedule of the system. In PM scheduling, a gene can represent for instance the preventive maintenance time internals of the

components of the system. The borrowed terms used in GA are in most cases suitably redefined to fit the algorithmic context (Marseguerra et al, 2006).

A GA does not operate on a system model, but on a mapping that exists between the model and the chromosome. This mapping is what is termed as *encoding*. Encoding the solution of a problem into a chromosome is a key issue (Gen and Cheng, 1997) to the success of the optimisation, since the efficiency of move operators and evaluation of fitness rely on it. According to Gen and Cheng (1997) the earlier encoding by Holland using binary strings is not a natural encoding for the problems from industrial engineering as it is difficult to directly apply. More so, according to Marseguerra et al (2006) there is no result that suggests binary codification provides better results than real (integer) codification, or vice versa. Therefore, it could be said that genetic encoding today is more of problem dependant suited for efficiency and effectiveness of the GA for the given problem.

Most common genetic operators in GA are crossover and mutation. Crossover combines the genes from two chromosomes referred to as parents to create a new chromosome referred to as off-spring or child. The parents are normally selected from chromosomes that are fit, which in principle will produce better off-springs. However, selection from one class of chromosomes may lead to local convergence and therefore the use of the other move operator becomes imperative. Mutation injects new traits by altering a gene, thereby reintroducing diversity into the population. In typical GA implementations, the mutation rate is very small and depends on the length of the chromosome (Konak et al, 2006).

A typical GA procedure comprises of the following steps, similarly contained in Konak et al (2006) and MacFarlane and Tuson (2009).

i. Set $i = 1$ and initialise generation counter $k = 1$

ii. Randomly generate N solutions to form an initial solution P_k and evaluate the fitness of each solution; fitness is usually evaluated through objective functions

iii. Select two parents *x* and *y* from P_k subject to fitness values

iv. Using crossover rate (probability) p_c, apply crossover on the two parents to produce an off-spring

v. With mutation rate (probability) p_m mutate the off-spring to inject diversity into the population P_k

vi. Evaluate the fitness of the off-spring and add it to Q_i

vii. If the size of Q_i does not equal N, then return to step iii

viii. Replace all weaker species in P_k with superior ones in Q_i and keep them in P_{k+1}

ix. Set $k = k + 1$, $i = i + 1$,

x. If stop criterion applies then stop with P_k being the set of decision vectors, else return to step iii

A typical binary encoding is in the form shown in Figure 3.5, consisting of strings of the digits "0" and "1". It is a string of twelve bits, each bit being a gene, while the whole string of bits represents the chromosome. The length of the chromosome is the total number of bits present, in this case 12.

1	0	1	1	0	1	0	0	1	0	1	1

Figure 3.5 - A typical binary encoding

With reference to *step iv* in the GA procedure, there are several ways by which crossover can be performed on a chromosome. Two of such are single-point and uniform crossover (Ashlock, 2005). In the case of single-point crossover, a random locus on the chromosome is selected and then serves as the point of crossover. For instance if A and B are two parent chromosomes each with index counter $i = 1..12$, and 7 being the random locus for crossover, then if *O* represents the resultant off-spring, *O* will consist of bits $A_1..A_6$, appended with $B_7..B_{12}$

It therefore appears that single-point crossover may not infuse much of diversity into the population. On the other hand, uniform crossover decides at each off-spring locus which parent will contribute its gene to that location. This strategy involves a computationally expensive process by random selection at each locus, however diversity is preserved. Let A and B be two parent chromosomes as seen in Figure 3.6. Assuming no parent genes are returned in succession by the random checker, then an example of possible resultant off-spring following uniform crossover is as shown in Figure 3.7. The shaded boxes represent the bits crossed from parent B.

A	1	0	1	1	0	1	0	0	1	0	1	1
B	0	1	1	0	1	0	0	1	0	1	1	0

Figure 3.6 - Typical parent chromosomes

O	1	1	1	0	0	0	0	1	1	1	1	0

Figure 3.7 - Resultant off-spring from uniform crossover

Increased number of GAs has successfully been used to treat optimisation problems in reliability (Painton and Campbell, 1995, Coit and Smith, 1996, Parker and Papadopoulos, 2007) and maintenance strategy (Wang et al, 1996, Tsai et al, 2001). This is largely due to GA's support for multi-objective optimisation. The tendency reveals that GA is an efficient tool to rapidly obtain the optimal solutions of PM policy.
In addition, recent work on multi-objective evolutionary algorithms has resulted in improved selection techniques which provide fast and efficient approximation of optimal Pareto fronts (Corne et al, 2000). Three of these techniques are Niched Pareto Genetic Algorithm (NPGA) II, Pareto Envelop-Based Selection Algorithm (PESA) II and Non-Dominated Sorting Genetic Algorithm (NSGA) II. Each of the techniques is oriented towards maintaining development of the Pareto frontier in a well spread manner. The main difference among the different selection techniques is the precise way in which the degree of isolation of an individual is estimated (Corne et al, 2001).

3.3.3.1 Niched Pareto Genetic Algorithm II

The NPGA II extends the traditional GA to multiple objectives through the use of Pareto domination ranking and fitness sharing, otherwise known as niching (Erickson et al, 2002). This implies that NPGA II introduces two processes to the traditional GA; (i) fitness ranking through Pareto domination to support problems that are multi-objective in nature, and (ii) fitness sharing to support a unique form of diversity. Hence it can be said that this new introductions do not only guide the search towards the Pareto optimal front, but also maintain population diversity in the Pareto optimal solution set through successive generations (Zhang et al, 2009). NPGA II is a modification of NPGA. NPGA uses probabilistic methods for selection which can result in a noisier search, whereas the method of deciding tournaments used by NPGA II is deterministic (Kunle, 2005). From Borges and Barbosa (2000), and Erickson et al (2002), a procedure for NPGA II can be outlined as follows.

i. Randomly generate an initial population P of size N
ii. $\forall p \in P$ and $\forall f \in p$, where f is objective function, evaluate f
iii. $\forall p \in P$, rank p according to domination rank
iv. Perform tournament selection as follows;
    a) Select 2 neighbourhood groups G_1 and G_2 according to radial distance
    b) $\forall g \in G_i$; i = 1..2, select candidate $o_i = g$ if g is the candidate with lowest rank in G_i, else where there is tie select o_i using fitness sharing
v. With crossover probability p_c, perform crossover on o_1 and o_2 producing an off-spring o
vi. With mutation probability p_m, mutate the off-spring o
vii. $\forall f \in o$, evaluate f, rank o and re-rank all $p \in P$
viii. Replace a weaker solution in P with o
ix. If stop criteria not reached then go to step iv, else stop giving all $p \in P$ with domination rank equalling 0 as the set of solutions

The selection process for recombination (reproduction) in NPGA II begins once all individuals have been assigned domination ranking. The domination rank for an individual i is the number of individuals dominating i. Individuals at the Pareto front are

those having domination rank equalling 0. To perform recombination, a tournament selection is used, which involves selecting a group of individuals within the ranked population. The individuals comprising this group are therefore referred to as candidates and are compared against each other by domination rank; the lowest in rank emerges as the winner for recombination.

When two or more candidates are of same lowest domination rank, then the tournament selection results in a tie. In order to break the tie, NPGA II uses fitness sharing, which is a tie breaking technique. Fitness sharing promotes the spread of individuals around the Pareto front and thereby injecting diversity into the population. It is also concerned with the population density around each candidate. The population density around each candidate is calculated within a specified Cartesian distance in objective function space called the niche radius as in equation 3.3 (Erickson et al, 2002, eq 1). An objective function space may be defined as the area covering the two optimal solutions at both ends of the Pareto frontier. For instance if f_1 and f_2 are two objective functions and s_x and s_y are the two optimal solutions at the opposite ends of the Pareto frontier, then the objective function space is the area covering [$f_1(s_x)$, $f_2(s_x)$] and [$f_1(s_y)$, $f_2(s_y)$].

$$CD_i = \sum_{\forall j \epsilon CP} \left(1 - \frac{d_{ij}}{\sigma_{share}}\right) \tag{3.3}$$

Where: CP is the candidate population

CD_i is the niche count (crowding density) or population density of candidate *i*

d_{ij} is the scaled radial distance between candidate *i* and *j*, each scaled by objective function space

σ_{share} is the scaled niche radius in objective function space

Where the values of the objective functions are each scaled as follows:

$$O_i' = \frac{O_i - O_{i,min}}{O_{i,max} - O_{i,min}} \tag{3.4}$$

Where: O_i', $O_{i,min} and\ O_{i,max}$ are the scaled, minimum and maximum values respectively of the *i-th* objective function O_i.

The sharing method described above has no form of fitness degradation implemented according to the value of the objective function *i* and its niche count. Instead, the best fit candidate is that which has the least number of individuals in its niche. It is likely that the higher the radial distance, the more likely it is that a candidate will be selected within a group for recombination. This mechanism helps promote an efficient Pareto optimal front convergence. The termination criterion depends on optimisation requirements, i.e. it could be after certain number of iterations, size of non-dominated individuals, etc. In general, the NPGA II procedure could be modified to meet optimisation requirements and to also improve performance for the given problem at hand.

3.3.3.2 Pareto Envelop-Based Selection Algorithm II

PESA II like NPGA II is an extension of the traditional GA to supporting multi-objective systems through the use of domination ranking. PESA II maintains two populations referred to as *internal* and *external*, the latter is often referred to as *archive* population. PESA II makes use of a region based selection method known as hyper-boxes. Each solution in the archive population inhabits a location within a hyper-box. The use of hyper-boxes ensure an even spread of decision vectors in the Pareto front. This implies that the performance of PESA II may however be affected by the size of the hyper-box. The optimal solutions are those decision vectors that are non-dominated within the solution space and are kept in the archive population.

A general procedure for PESA II, similar to that found in Corne et al (2000, 2001), and Parker and Papadopoulos (2007) is as follows.

i. Generate at random an initial internal population P_I of N individuals
ii. Set external population P_E to empty set
iii. $\forall p \in P_I$ and $\forall f \in p$, where *f* is objective function of *p*, evaluate *f*

iv. $\forall p \in P_I$, rank R_p (rank of p) according to domination

v. If P_E is empty then $\forall p \in P_I$ such that $R_p = 0$ (non-dominated), move p into P_E else do the following sub-task:

   a) Compare each member of P_I with that of P_E for dominance and re-rank each element of P_I and P_E

   b) $\forall p \in P_E$ such that $R_p > 0$ remove p from P_E

   c) $\forall p \in P_I$ such that $R_p = 0$ move p into P_E

vi. If the size of P_E is exceeded (i.e. > N), then repeatedly find a hyper-box with the highest *squeeze factor* and randomly remove one individual until size of P_E equals N

vii. If termination criteria is reached, then stop and return members of P_E as the optimal solutions, else delete the current content of P_I and do the following sub-task until N new candidate solutions are generated into P_I

   d) Select 2 hyper-boxes at random and pick one having smaller *squeeze factor*, in the event of a tie, it is broken by random selection. A squeeze factor refers to the number of decision vectors in a hyper-box

   e) Select at random 2 parents p_1 and p_2 from the hyper-box

   f) With crossover probability p_c produce a single off-spring o via crossover on p_1 and p_2

   g) With mutation probability p_m mutate o to hopefully inject a new trait into the population

viii. Return to step iii

PESA II is an improvement over PESA, in that in PESA, selection for recombination is only done on the archive population containing non-dominated individuals. More on PESA is found in Corne et al (2000, 2001).

3.3.3.3 Non-Dominated Sorting Genetic Algorithm II

Like NPGA II and PESA II, NSGA II is also a true multi-objective optimisation algorithm and an evolution of the NSGA algorithm. Although only NSGA II will be discussed here, it would be useful to start with the problems in the original NSGA that

motivated this subsequent work. Firstly, NSGA uses a *sharing* parameter for ensuring diversity and the right tuning of this parameter is difficult. Secondly NSGA is a non-elitist algorithm, while elitism can speed up the performance and prevent loss of good solutions (Deb et al, 2000). NSGA II like NPGA II and PESA II uses the concept of Pareto frontier to produce optimal or near optimal solutions. However, NSGA II is quite different in the sense that it can produce solutions in multiple fronts; first, second and to the *k-th* front; where *k* is a preset number of fronts of optimality. In NSGA II, sharing is replaced with crowding comparison.

NSGA II begins by randomly generating an initial population *P* consisting of N number of individuals. Two entities are calculated; (i) n_p, the number of solutions which dominate the solution p in P and (ii) S_p, the set of solutions for which solution p dominates (Deb et al, 2000). All the solutions with $n_p = 0$ are stored in F_1, which forms the current set of optimal solutions (in this case the first front). For all dominated sets of solutions S_p, in the first front visit each q of S_p, decrement n_q by 1. If this results to 0, q is added to the set F_2, which at the end of the process becomes the current set of optimal solutions for the second front. The process is repeated for every current solution until k fronts are obtained.

A procedure for NSGA II which is similarly found in Deb et al (2000, 2002) and Favuzza et al (2006) is presented as follows.

i. Set population index t = 1
ii. Generate at random an initial population *P* of N number of individuals
iii. For each solution $p \in P_t$, find n_p number of solutions that dominate p and S_p set of solutions for which p dominates
iv. Set front index $i = 1$
v. Add all p with $n_p = 0$ into the set F_i, the i-th front (i.e. rank of p; $R_p = i$)
vi. For each $p \in F_i$ assign crowding distance to p
vii. Increment front index by 1, implying $i = i + 1$
viii. For each $p \in F_{i-1}$, visit each $q \in S_p$ and decrement n_q by 1. If by doing this, n_q becomes 0 then add q into the set F_i (q belongs to front i, $R_q = i$)

ix. Repeat step vi to find subsequent fronts until front-termination criteria is reached

x. Increment population index by 1; t = t + 1

xi. Perform breeding

xii. If stop criteria not reached then go to iii else stop

At the end of the optimisation, solutions at the *i-th* front will in no doubt be better than those at the *(i+1)-th* front.

The crowding distance assignment contained in *step vi* of the NSGA II procedure is aimed at maintaining diversity, and it is evaluated as follows.

a) Sort all l number of solutions in a given front in ascending order of the objective function *fm* and compute the crowding distance *CD* as follows:

$$CD_{jm} = \frac{f_m(x_{j+1}) - f_m(x_{j-1})}{f_m(x_{max}) - f_m(x_{min})}, \qquad j = 2..l-1, \tag{3.5}$$

b) Repeat step "a" above for each objective function and then find the crowding distance of solution *j* as:

$$CD_j = \sum_{m=0}^{M} CD_{jm} \tag{3.6}$$

Where M is the number of object functions.

Breeding in NSGA II is described as follows (Deb et al, 2002).

i. Set child population Q_t to an empty set
ii. Using binary tournament selection on parent population P_t, select 2 parents p_1 and p_2 using crowding comparison operator
iii. With crossover probability p_c apply crossover on p_1 and p_2 to produce an off-spring o
iv. With mutation probability p_m mutate off-spring o
v. Add off-spring o to Q_t
vi. Repeat step ii until size of Q_t equals N
vii. Let $Z_t = P_t \cup Q_t$, sort Z_t based on domination rank
viii. Set front index $j = 1$
ix. For each $p \in F_j$ in Z_t add p to P_{t+1} if size of $P_{t+1} < N$
x. Increment front index by 1; $j = j+1$
xi. Repeat step ix if size of $P_{t+1} < N$

The crowding comparison operator is used to determine the preference of an individual over another with the aim to maintaining diversity in the Pareto front. This is specified below, as found in Deb et al (2002).

- Given two solutions x and y, solution x is preferred over solution y if $R_x < R_y$ or ($R_x = R_y$ and $CD_x > CD_y$). Where R_x and R_y are domination ranks for solutions x and y respectively

NSGA II has been shown to be computational efficient (Favuzza et al, 2006, Yijie and Gongzhang, 2008).

3.3.3.4 Comparison of Multi-Objective Genetic Algorithm Techniques

According to (Favuzza et al, 2006) a multi-objective optimisation algorithm must achieve the following two goals.

- Guide the search towards global optimality
- Maintain diversity in the Pareto front, so that all trade-offs among parameters of the optimisation are explored and the search is not polarised in particular regions of the space (e.g. only very low cost or high dependability solutions are explored)

In a given feasible region, there could be neighbourhoods within the population in any given generation where individuals in such neighbourhoods are only superior to their neighbours. This is a clear scenario of local optimum. The first goal above is aimed at preventing the convergence of search within such local optimum. The second goal aims at populating as much as possible the Pareto front with non-dominated individuals. This entails that the three variations of multi-objective optimisation selection techniques i.e. NPGA II, PESA II and NSGA II try to achieve these goals in their own specific ways and hence comparison could be established based on their unique approaches.

In NPGA II, weaker solutions in the current population are replaced with better ones from a child population. This implies that breeding is some how biased towards the best solutions within the current population, and this can affect global optimality. Diversification on the other hand is through region based selection which largely depends on a radial distance and a sharing parameter. For a given problem, the radial distance may need to be adjusted for different optimisation runs in search for an ideal value.

To achieve global optimality and to main diversity, PESA II uses a hyper-box. A hyper-box is selected at random within the objective function space implying that selection for breeding is not only restricted to best fit individuals. This is because through genetic operators, it may be possible to produce promising solutions from weaker individuals.

In order to maintain a spread along the Pareto front, a hyper-box with the least number of individuals is preferred for breeding. The size of the hyper-box affects the convergence of solutions at the Pareto front. Currently, there appears to be no clear guideline in literature on how the size of the hyper-box is estimated. The use of hyper-box dimensions causes problems similar to those introduced when specifying the sharing parameters of other algorithms (Kunle, 2005) such as the NPGA II.

To maintain global optimality, NSGA II promotes breeding from both fit and less fit individuals especially at early generations. As generations evolve there is likelihood that the current population may mostly contain individuals at the first front. Hence selection pressure is less at early generations to allow for discovery of path for global optimality that may arise from the current less fit individuals. To maintain an even spread on the Pareto front, NSGA II does not make use of a selection region such as hyper-boxes or group region for diversification. It makes use of a fast non-dominated ranking (Favuzza et al, 2006) and crowding comparison, this makes it more efficient and effective. NSGA II is computationally efficient and has also proved to be quite efficient in many different applications (Favuzza et al, 2006).

The general argument about region based selection (used in NPGA II and PESA II) is that when there is a non-uniform distribution of individuals containing large clusters, region based selection will treat the large clusters and the isolated individuals as equal groups with equal probability of participating in each tournament (Kunle, 2005). This scenario however does not affect the performance of crowding comparison.

3.4 Discussion

There is no conclusive argument that a particular search technique is better than another. However, for a given type of optimisation problem, one search technique may be preferable over another. For instance, hill climbing and simulated annealing may be preferred over GA for a simple problem with fewer individuals in the solution space. Although hill climbing and simulated annealing may be viewed to operate like a steady state GA (suitable for a single-objective problem), there is no indication that the two can

not be applied on a multi-objective problem. Therefore preference may be based on computational expense, efficiency and the ease of implementation for the given problem.

A variation of hill climbing known as hybrid immune-hill climbing optimisation algorithm has been used for global multi-objective optimisation in design and manufacturing (Yildiz, 2009). Similarly simulated annealing extensions like orthogonal simulated annealing (OSA) and classical simulated annealing based multi-objective algorithm (CMOSA) are suited for multi-objective optimisation problems (Suman et al, 2009).

There is a large body of work on applying GAs to multi-objective optimisation problems including problems that have high complexity and enormous solution spaces. GAs have demonstrated the ability to perform multi-directional search and find multiple solutions in a single run, converge speedily to the Pareto front with high degree of accuracy and handle non-linear optimisation problems with ease (Suman, 2004, Suman et al, 2009). It is these attributes that make GAs suitable for many engineering problems, and the preferred choice of optimisation technique for the work described in this thesis.

4 DYNAMIC EFFECTS OF MAINTENANCE

Maintenance is carried out on the components of a system in order to improve the reliability of components and of the system as a whole. The extent of that improvement depends on several factors which determine whether a component is returned to a state as *good-as-new* or somewhere in between that state and its state before maintenance. A periodic maintenance action that improves the state of a component to good-as-new is termed as perfect preventive maintenance (PPM), whereas that which improves the state to a certain degree is termed imperfect preventive maintenance (IPM).

The state of a component in maintenance usually refers to performance parameters; such as component shock level and age. This work focuses on the component age. The effectiveness of maintenance in reducing the age of a component defines the lifecycle reliability, availability and cost of the component. The dynamic effects of maintenance on these parameters are explored in this chapter and in particular the novel evaluation models for component reliability, availability/unavailability and cost under IPM and PPM policies are established.

According to Márquez (2007), maintenance modelling is quite under-developed due to some factors, two of which are highlighted below.

- *Lack of maintenance management models*: maintenance currently lacks models that could improve the understanding of the underlying dimension of maintenance.
- *Wide diversification of the maintenance problem*: maintenance comprises of set of activities which make it difficult to have information and support system in one place to ease the improvement process.

The above sums up the views of Sanchez et al (2009) with respect to imperfect maintenance, that "in many cases, there is limited knowledge on the proper model to represent a problem, and thus results in that for a particular imperfect maintenance model, there are multiple competing models producing a different approximation of the same problem."

This chapter also explores the appropriate literature on maintenance modelling, and identifies a way of modelling perfect and imperfect preventive maintenance that can further be used for the purpose of optimising maintenance schedules.

4.1 Perfect Preventive Maintenance

The age reduction model presumes that after each maintenance action, the age of a given component assumes a new value known as the *effective age*. If W is the current age of a component, its effective age is represented as W^+. The plus sign indicates that this new age only takes effect after the maintenance action. Under periodic maintenance, preventive maintenance is carried out at an interval known as *PM time* T_p. It is likely that several components will have different PM times. This work assumes that a given system has a shortest maintenance interval known as *PM interval T*, based on which components PM time could be obtained as seen in equation 4.1.

$$T_{pi} = \boldsymbol{\alpha}_i T \qquad ; \boldsymbol{\alpha}_i : \mathbb{N} \bullet \boldsymbol{\alpha}_i \geq 1 \tag{4.1}$$

Where: T_{pi} is the PM time for the *i-th* component

$\boldsymbol{\alpha}_i$ is the coefficient of maintenance interval (CoMI) for the *i-th* component

At the *j-th* PM action, the effective age W_j^+ of a component is expected to fall within a boundary as shown below.

$0 < W_1^+ < T_p$; j = 1

$W_1^+ < W_2^+ < W_1^+ + T_p$; j = 2

$W_2^+ < W_3^+ < W_2^+ + T_p$; j = 3

In general, this can be represented as:

$$W_{j-1}^+ < W_j^+ < W_{j-1}^+ + T_p \tag{4.2}$$

Using PM time for the *i-th* component and at the *j-th* PM stage, equation 2.1 can be transformed into equation 4.3 below.

$$W_j^+ = (1 - f_i)T_{pi} \tag{4.3}$$

The assumption that the improvement factor assumes a value equalling 1 for a PPM implies that equation 4.3 evaluates to a value 0 under PPM. This also implies that equation (inequality) 4.2 does not apply for a component under PPM policy.

4.1.1 Universal Modelling of the Effect of PPM on Component Reliability

To model the reliability of a component under PM, two scenarios are considered:

i) The probability of surviving until PM time nT_p

ii) The probability of surviving the remaining time $t - nT_p$; $nT_p \leq t \leq \tau$

Where: n is the total number of PM stages since t = 0

t is the calendar age of the component

τ is the useful life of the component or the scale of time under consideration

The total number of PM stages n that can be performed on a component under PPM is predetermined as follows.

$$n = \begin{cases} Q\left(\frac{MTTF}{T_p}\right) & ;\ MTTF \leq RT \\ Q\left(\frac{RT}{T_p}\right) & ;\ MTTF > RT \end{cases}$$

Where $MTTF$ is the mean time to failure of the component, RT is the useful system operational life time also known as system risk time and Q is the integer quotient of the division.

According to Tsai et al (2004, eq 1), the reliability model of a system on the *j-th* PM stage can be constructed as shown in equation 4.4.

$$R_j(t) = R_{0j}R_{vj}(t) \tag{4.4}$$

Where R_{oj} is the probability of surviving until the *j-th* PM stage, and R_{vj} is the probability of surviving the remaining time. According to Birolini (2007) a component (or item) which is new at $t_0 = 0$, and with PM intervals t_1, t_2, ... are statistically independent. Hence the independence in reliabilities of successive PM times R_{oj} and remaining time R_{vj}.

To model the universal reliability of a component under PPM, Figure 4.1 is considered which shows the PM stages of a component, from when *j = 1* to *j = n*.

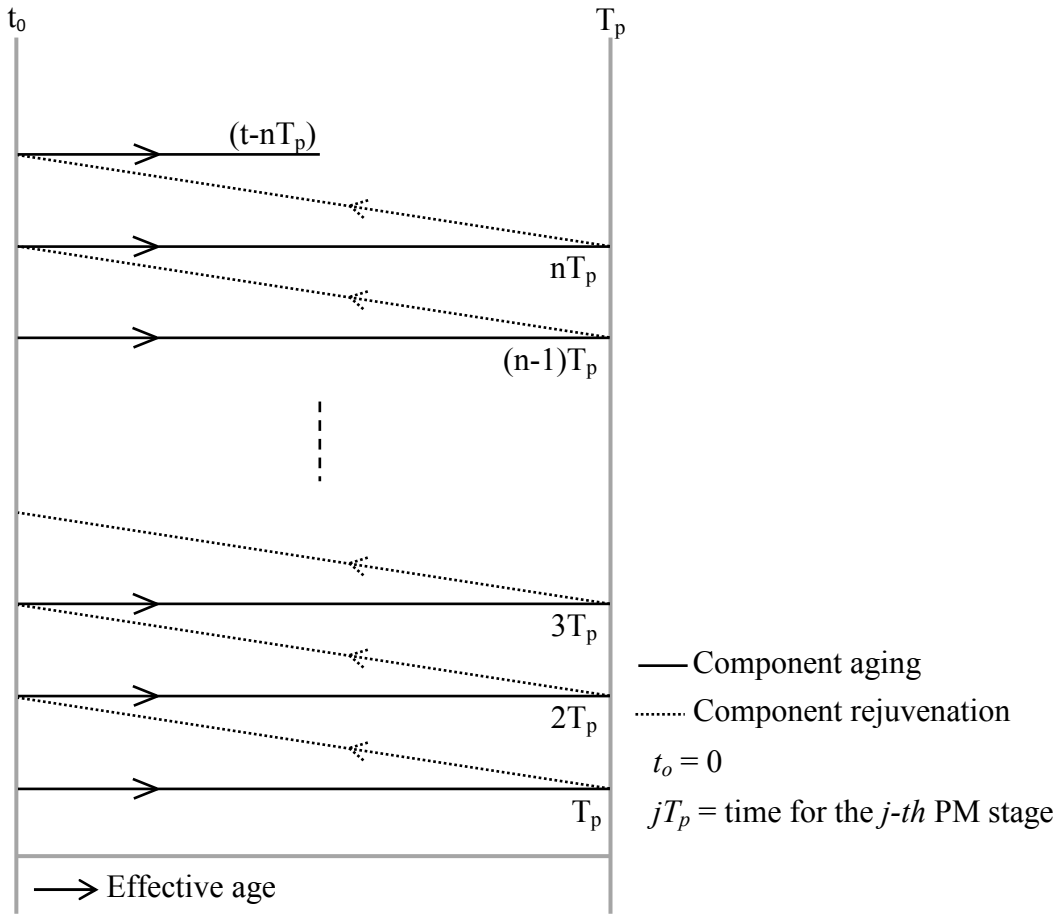

Figure 4.1 - Universal modelling of reliability under PPM

Let t_{rem} represent the remaining time **after** any PM stage, such that $t_{rem} = t - jT_p$. **At** any PM stage, $t_{rem} = 0$, implying that $R(t_{rem}) = 1$. Therefore, at every PM stage the following applies.

$$R(t) = R(T_p) \qquad ; j = 1 \wedge t = T_p$$
$$R(t) = R(T_p)\,R(T_p) \qquad ; j = 2 \wedge t = 2T_p$$
$$R(t) = R(T_p)\,R(T_p)\,R(T_p) \qquad ; j = 3 \wedge t = 3T_p$$

And more generally:

$$R(t) = \prod_{j=1}^{n} R(T_p) \tag{4.5}$$

Equation 4.5 of course transforms to 4.6 which gives the probability of surviving the *n-th* PM stage, where n:ℕ and $t = nT_p$.

$$R(t) = R(T_p)^n \tag{4.6}$$

For a scenario where $nT_p < t < (n+1)T_p$, $t_{rem} = t - nT_p$, hence:

$$R(t_{rem}) = R(t - nT_p) \tag{4.7}$$

Using equation 4.4, the universal model for reliability under PPM is therefore the product of equation 4.6 and 4.7, and is as expressed in equation 4.8.

$$R_{pu}(t) = R(T_p)^n R(t - nT_p) \tag{4.8}$$

Where R_{pu} is the cumulative reliability of a component under perfect preventive maintenance. The first part of the product is the probability of surviving n PM stages, while the second is the probability of surviving the remaining time.

4.1.2 Weibull Distribution Modelling of PPM

Based on equation 4.8, dependability parameters such as reliability and availability (or unavailability) can be established for a component whose failure characteristics follow a given probability distribution. In this thesis, it is assumed that components failure characteristics follow the Weibull distribution. This is a common assumption made for components that carry mechanical parts. Note that the alternative of assuming a fixed failure rate which leads to exponential distribution leads to a paradoxical but mathematically proven result according to which PPM has absolutely no effect on component reliability (appendix A).

4.1.2.1 Reliability

The reliability model with no PM under Weibull distribution is as shown in equation 4.9 (Márquez 2007, eq 4.9). Where γ, θ and β are location, scale and shape (slope) parameter respectively.

$$R(t) = exp\left[-\left(\frac{t-\gamma}{\Theta}\right)^{\beta}\right] \tag{4.9}$$

The Weibull model for reliability based on the universal model of reliability under PPM can be established through the use of equation 4.9. For simplicity, let the following representations apply.

$$u = \left(\frac{t-\gamma}{\theta}\right)^{\beta}$$

$$e = \exp$$

It then follows that equation 4.9 can take the form $R(t) = e^{-u}$. The first scenario is to establish the probability of surviving until n PM stages from equation 4.6.

$$R(T_p)^n = \prod_{j=1}^{n} e_j^{-u}$$

$$R(T_p)^n = e_1^{-u}\, e_2^{-u}\, e_3^{-u} \ldots e_{n-1}^{-u}\, e_n^{-u} \quad (4.10)$$

For all e_j^{-u} ; j = 1 .. n, u is constant, since $t = T_p$ at each PPM stage and also the Weibull parameters being constants. Therefore $\forall j : \mathbb{N} \mid 1 < j \leq n \bullet e_{j-1}^{-u} = e_j^{-u}$ and hence the following applies; since all e_j^{-u} are equal:

Let $e^{-u} = e_j^{-u}$; for all $j = 1 .. n$

Thus equation 4.10 can be rewritten as:

$$R(T_p)^n = \prod_{j=1}^{n} e^{-u}$$

Applying the laws of indices on the above, results into the following:

$$R(T_p)^n = e^{-(u+u+u+\cdots+u+u)}$$

Where u is n-tuple and therefore $u+u+u+ \ldots +u+u = nu$; hence the following holds:

$$R(T_p)^n = e^{-nu} \quad (4.11)$$

Substituting for u in equation 4.11, the probability of surviving until the *n-th* PM stage is:

$$R(T_p)^n = exp\left[-n\left(\frac{t-\gamma}{\Theta}\right)^{\beta}\right] \quad (4.12)$$

Utilising still equation 4.9, the probability of surviving the remaining time $trem = t - nT_p$, is:

$$R(t-nT) = exp\left[-\left(\frac{(t-nT_p)-\gamma}{\Theta}\right)^{\beta}\right] \quad (4.13)$$

Combining equations 4.12 and 4.13 in the form of equation 4.8, gives the Weibull model for component reliability under PPM as shown in equation 4.14.

$$R_{pc}(t) = exp\left[-n\left(\frac{t-\gamma}{\Theta}\right)^{\beta}\right] exp\left[-\left(\frac{(t-nT_p)-\gamma}{\Theta}\right)^{\beta}\right] \quad (4.14)$$

$nT_p \leq t \leq (n+1)T_p$

For a component where life is considered to begin from origin (t = 0, as assumed in this work), the location parameter γ takes a value 0 and equation 4.14 reduces to a 2-parametric Weibull distribution model as seen in equation 4.15.

$$R_{pc}(t) = exp\left[-n\left(\frac{T_p}{\Theta}\right)^{\beta}\right] exp\left[-\left(\frac{t-nT_p}{\Theta}\right)^{\beta}\right] \quad ; nT_p \leq t \leq (n+1)T_p \quad (4.15)$$

Equation 4.15 is however found in Ebeling (1997, pp205).

4.1.2.2 Unavailability

The assumption under PPM is that there is no repair and therefore the unavailability of a component under PPM is as seen in equation 4.16; U_{pc} being the unavailability of the component under PPM.

$$U_{pc} = 1 - R_{pc}(t) \quad (4.16)$$

4.1.3 PPM Cost

The total preventive maintenance cost varies in response to variation in the total number of PM stages for a component. With the assumption that no repair is carried out in PPM policy, the total component cost model is a simple one. Considering the *i-th* component of a system, its total cost is as seen in equation 4.17.

$$C_{pci} = n_i C_{ppmi} + C_{ci} \tag{4.17}$$

Where C_{pci} is the total cost for the *i-th* component under PPM
C_{ppmi} is the cost of performing PPM for the *i-th* component
C_{ci} is the unit cost of the *i-th* component
n_i is the total number of PM stages for the *i-th* component

Using equation 4.17, the total system cost C_{ps} under PPM is established as:

$$C_{ps} = \sum_{i}^{m} C_{pci} \tag{4.18}$$

Where *m* is the number of system components identified for PPM.

4.2 Imperfect Preventive Maintenance

Under imperfect preventive maintenance, it is presumed that each maintenance activity improves the state of a component by some degree, depending on its effectiveness (Martorell et al, 1999, Márquez 2007). This implies that the new effective age of the component lies in-between its effective age at the previous maintenance stage and the age at current maintenance stage, hence equation 4.2 fully applies. This also suggests that the improvement factor *f* is less than 1, i.e. $0 \leq f < 1$. By equation 4.3, it is obvious that the effective age of a component after the first PM stage is:

$$W_1^+ = (1 - f_1)T_{p1} \qquad ; j = 1$$

Similarly, the age after the second PM stage is:

$$W_2^+ = W_1 + (1 - f_2)T_{p2} \qquad ; j = 2$$

Substituting for W_1^+ in W_2^+ gives:

$$W_2^+ = (1 - f_1)T_{p1} + (1 - f_2)T_{p2} \qquad ; j = 2$$

Therefore according to the principle of mathematical induction, the age of a component at the *n-th* PM stage can in general be modelled as:

$$W_n^+ = (1 - f_1)T_{p1} + (1 - f_2)T_{p2} + \cdots + (1 - f_{n-1})T_{p(n-1)} + (1 - f_n)T_{pn} \qquad ; j = n$$

For simplicity W_n^+ can be expressed as in equation 4.19.

$$W_n^+ = \sum_{j=1}^{n} (1 - f_j)T_{pj} \tag{4.19}$$

In scenarios where both PM time T_{pj} and improvement factor f_j ($j = 1..n$) are constants, then equation 4.19 simplifies to 4.20.

$$W_n^+ = (1 - f)nT_p \tag{4.20}$$

Where n is the total number of PM stages attained since when $t = t_0 = 0$ (when component was new). The total number of PM stages n for a given component of a system under IPM is similar to that under PPM and is predetermined as follows.

$$n = \begin{cases} Q\left(\dfrac{MTBF}{T_p}\right) & ; MTBF \leq RT \\ Q\left(\dfrac{RT}{T_p}\right) & ; MTBF > RT \end{cases}$$

Where *MTBF* is the mean time between failures of the component, *RT* and Q remain the system risk time and the integer quotient of the division respectively as discussed for a component under PPM.

Considering the first maintenance stage where $j = 1$, Figure 4.2 is a graphic illustration of the evaluation of equation 4.20. τ is the useful life or time scale considered for the entire analysis of the component.

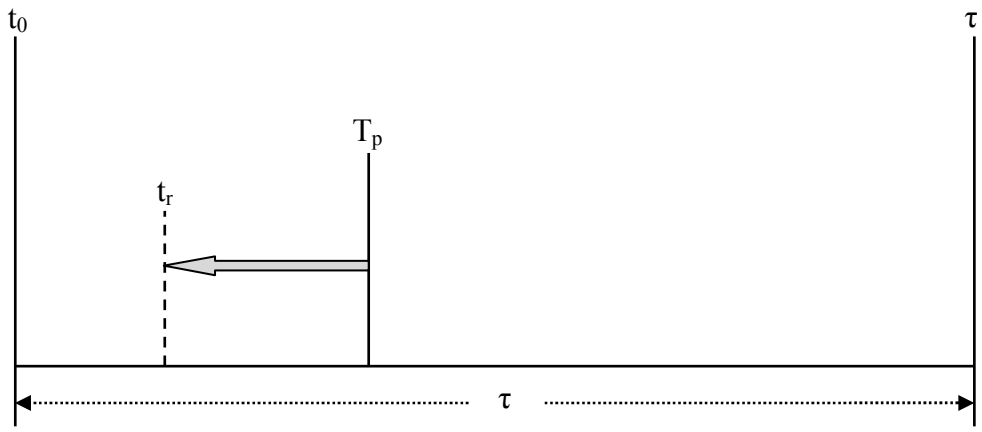

Figure 4.2 - Component effective age at first PM stage

Figure 4.2 shows that maintenance action at time $t = T_p$, rejuvenates the component age to an effective age t_r, and this further implies that the next maintenance stage will be carried out at a time $t_r + T_p$. According to equation 4.20, the evaluation of t_r is as shown in equation 4.21, and this simple representation will from now onward be used for further mathematical derivations.

$$t_r = (1 - f)T_p \tag{4.21}$$

4.2.1 Universal Modelling of the Effect of IPM on Component Reliability

To model the universal reliability of a component under IPM, Figure 4.3 is considered. The meanings of the lines used in Figure 4.3 are same as those of Figure 4.1.

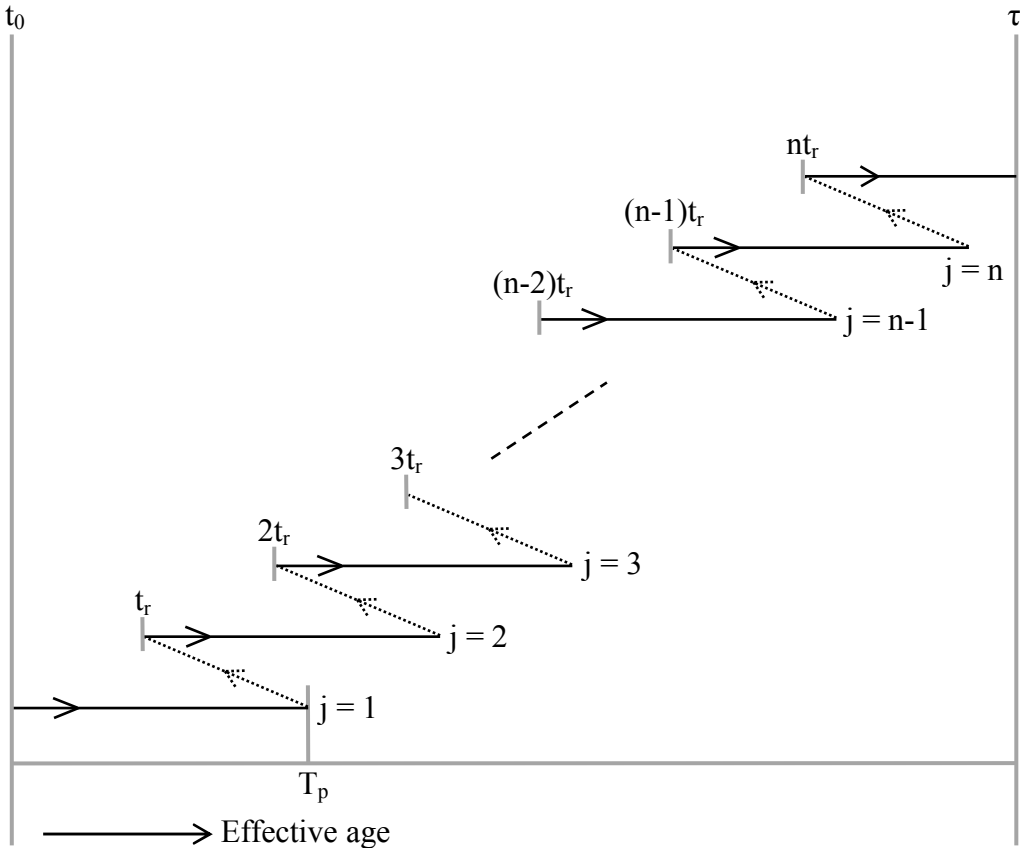

Figure 4.3 - Universal modelling of reliability under IPM

Figure 4.3 shows the PM stages of a component, from when *j = 1* to *j = n*. **At** any PM stage where $t_{rem} = 0$, the probability of surviving the *j-th* PM stage is modelled as follows.

$$R(t) = R(T_p) \qquad ; j = 1 \wedge t = T_p$$

$$R(t) = R(T_p)\, R(t_r, t_r + T_p) \qquad ; j = 2 \wedge t = 2T_p$$

$$R(t) = R(T_p) R(t_r, t_r + T_p) R(2t_r, 2t_r + T_p) \qquad ; j = 3 \wedge t = 3T_p$$

Therefore, according to the principle of mathematical induction, it follows that:

$$R(t) = R(T_p) R(t_r, t_r + T_p) R(2t_r, 2t_r + T_p) \ldots R((n-2)t_r, (n-2)t_r + T_p) R((n-1)t_r, (n-1)t_r + T_p) \qquad ; j = n \wedge t = nT_p \qquad (4.22)$$

In general equation 4.22 can be simplified as shown in equation 4.23, $t = nT_p$.

$$R(T_p)^n = R(T_p)\prod_{j=2}^{n} R\left((j-1)t_r, (j-1)t_r + T_p\right) \tag{4.23}$$

To establish the evaluation for component reliability in the form R(x,y) as seen in equation 4.23 where x and y are two points (intervals) in time units, the probability of failure of the component between these two points is considered. The probability of failure F(x,y) between two points x and y can be given as:

$$F(x,y) = F(y) - F(x) \qquad ; x \leq y \tag{4.24}$$

According to Márquez (2007, eq 4.10):
R(t) = 1 – F(t) ; therefore *F(t) = 1 - R(t)*

Hence, equation 4.24 becomes:

$$\begin{aligned}
1 - R(x,y) &= 1 - R(y) - \left(1 - R(x)\right) \\
1 - R(x,y) &= 1 - R(y) - 1 + R(x) \\
1 - R(x,y) &= 1 - 1 - R(y) + R(x) \\
1 - R(x,y) &= R(x) - R(y) \\
-R(x,y) &= -1 + R(x) - R(y) \\
R(x,y) &= 1 - R(x) + R(y)
\end{aligned} \tag{4.25}$$

Starting from origin (i.e. $t = t_0 = 0$) to T_p, the reliability of this interval using equation 4.25 is:

$$\begin{aligned}
R(0,T_p) &= 1 - R(0) + R(T_p) \\
R(0,T_p) &= 1 - 1 + R(T_p) \\
R(0,T_p) &= R(T_p) \\
R(T_p) &= R(0,T_p)
\end{aligned} \tag{4.26}$$

Substituting for $R(T_p)$ into equation 4.23 gives:

$$R(T_p)^n = R(0, T_p) \prod_{j=2}^{n} R\left((j-1)t_r, (j-1)t_r + T_p\right)$$

The above simplifies to equation 4.27.

$$R(T_p)^n = \prod_{j=1}^{n} R\left((j-1)t_r, (j-1)t_r + T_p\right) \tag{4.27}$$

Equation 4.27 gives the modelling for the probability of the component surviving *n* PM stages. The second aspect to deal with is the probability of the component surviving the remaining time t_{rem} which is trivial from Figure 4.3 and is given by equation 4.28.

$$R(t_{rem}) = R(nt_r, nt_r + (t - nT_p)) \tag{4.28}$$

Hence the product of equations 4.27 and 4.28 gives the universal model for reliability $R_{iu}(t)$ under IPM as shown in equation 4.29.

$$R_{iu}(t) = \left(\prod_{j=1}^{n} R\left((j-1)t_r, (j-1)t_r + T_p\right)\right) R(nt_r, nt_r + (t - nT_p)) \tag{4.29}$$

Using equation 4.24, equation 4.29 is transformed into 4.30 as:

$$R_{iu}(t) = \left(\prod_{j=1}^{n} \left(1 - R\left((j-1)t_r\right) + R\left((j-1)t_r + T_p\right)\right)\right) \left(1 - R(nt_r) + R(nt_r + (t - nT_p))\right) \tag{4.30}$$

4.2.2 Weibull Distribution Modelling of IPM

Using Weibull distribution, the modelling of cost and dependability parameters such as reliability and availability (or unavailability) under IPM is here established.

4.2.2.1 Reliability

The component reliability model under IPM using Weibull distribution can be established from equation 4.30; the universal model for component reliability under IPM. Equation 4.30 is used in conjunction with equation 4.9 to give equation 4.31.

$$R_{ic}(t) = \prod_{j=1}^{n}\left(1 - exp\left[-\left(\frac{(j-1)t_r - \gamma}{\theta}\right)^{\beta}\right] + exp\left[-\left(\frac{\left((j-1)t_r + T_p\right) - \gamma}{\theta}\right)^{\beta}\right]\right)\left(1 - exp\left[-\left(\frac{nt_r - \gamma}{\Theta}\right)^{\beta}\right] + exp\left[-\left(\frac{(nt_r + (t - nT_p)) - \gamma}{\Theta}\right)^{\beta}\right]\right) \quad (4.31)$$

Equation 4.31 gives the Weibull model for component reliability under IPM. It comprises of an iterative evaluation of the probability of surviving until the *n-th* PM stage.

4.2.2.2 Unavailability

Under IPM, repair of components is taken into account. However, one of objectives of IPM is to improve availability either through quick but effective repair or reducing to a very minimal level the occurrence of failure that will infer corrective maintenance. Hence, the assumption in this work is that no failure that will bring the system to halt resulting into corrective maintenance occur in-between PM stages. As a result, minimal repair (Martorell et al, 1999) is considered. Hence a component requiring minimal repair implies that in operation terms it is performing in degraded mode. Minimal repair

actions target simple parts of a component; for instance according to Tsai et al (2004) these could include replacing a seal, spring, bearing, etc.

The availability of a component A_{ic} normally depends on reliability and maintenance. Thus availability can be modelled using the standard availability expression seen in equation 4.32 (van Dijkhuizen and van der Heijden, 1999, Artana and Ishida, 2002, eq 22 and Goto et al, 2006, eq 18).

$$A_{ic} = \frac{UT}{UT + DT} \tag{4.32}$$

Where: UT is the up time of the component
DT is the down time of the component

Let μ_m be the mean time for minimal repair of the component
μ be the mean time to repair of the component
$\lambda(t)$ be the hazard rate (also referred to as failure rate) of the component

Then, UT and DT can be defined as:

$$UT = T_p - \mu_m \int \lambda(t)dt \tag{4.33}$$

$$DT = \mu + \mu_m \int \lambda(t)dt \tag{4.34}$$

Similar expression of equations 4.33 and 4.34 are found in Tsai et al (2004, eq 14 and 15) while Sheu et al (2006, eq 3) has similar expression to equation 4.33.

To solve for *λ(t)*, let the following hold;

$$N(t) = \int \lambda(t)\, dt$$

According to Birolini (2007, eq 1.5):

$$\lambda(t) = -\frac{dR(t)/dt}{R(t)} \tag{4.35}$$

Using Weibull distribution, the following applies;

$$dR(t) = e^{\left[-\left(\frac{t-\gamma}{\theta}\right)^{\beta}\right]}dt$$

Since it is assumed that a component life begins at time t = 0, the location parameter γ takes a value 0 and therefore the above is simplified to:

$$dR(t) = e^{-\left(\frac{t}{\theta}\right)^{\beta}}dt$$

Let u = $-\left(\frac{t}{\theta}\right)^{\beta} = -\frac{t^{\beta}}{\theta^{\beta}}$

$$\frac{du}{dt} = -\beta\frac{t^{\beta-1}}{\theta^{\beta}}$$

Let $v = e^{u}$

$$dv = e^{u}du$$

$$\frac{dv}{du} = e^{u}$$

By using chain rule:

$$\frac{dR(t)}{dt} = \frac{dv}{du}\frac{du}{dt} = e^{u}\left(-\beta\frac{t^{\beta-1}}{\theta^{\beta}}\right)$$

$$\frac{dR(t)}{dt} = e^{\left(-\frac{t^{\beta}}{\theta^{\beta}}\right)}\left(-\beta\frac{t^{\beta-1}}{\theta^{\beta}}\right)$$

$$\frac{dR(t)}{dt} = -e^{\left(-\frac{t^{\beta}}{\theta^{\beta}}\right)}\left(\beta\frac{t^{\beta-1}}{\theta^{\beta}}\right) \tag{4.36}$$

Substituting equation 4.36 into 4.35, gives:

$$\lambda(t) = -\left[\frac{\left(-e^{\left(-\frac{t^{\beta}}{\theta^{\beta}}\right)}\left(\beta\frac{t^{\beta-1}}{\theta^{\beta}}\right)\right)}{e^{-\left(\frac{t}{\theta}\right)^{\beta}}}\right]$$

$$\lambda(t) = -\left(-\frac{e^{\left(-\frac{t^{\beta}}{\theta^{\beta}}\right)}\left(\beta\frac{t^{\beta-1}}{\theta^{\beta}}\right)}{e^{\left(-\frac{t^{\beta}}{\theta^{\beta}}\right)}}\right)$$

$$\lambda(t) = \beta\frac{t^{\beta-1}}{\theta^{\beta}}$$

$$N(t) = \int \lambda(t)dt = \int \beta\frac{t^{\beta-1}}{\theta^{\beta}}dt$$

$$N(t) = \frac{\beta}{\theta^{\beta}}\int t^{\beta-1}dt$$

$$N(t) = \frac{\beta}{\theta^{\beta}}\left(\frac{1}{\beta}\,t^{\beta}\right)$$

$$N(t) = \frac{1}{\theta^{\beta}}\left|\,t^{\beta}\,\right| \tag{4.37}$$

The limits of the integration will be the effective age at the previous PM stage (as lower limit) and the age at the current PM stage (as upper limit). Therefore:

$$N(t) = \int_{W_{j-1}^{+}}^{W_j} \lambda(t)dt$$

$$N(t) = \frac{1}{\theta^{\beta}} \left| t^{\beta} \right|_{W_{j-1}^{+}}^{W_j} \tag{4.38}$$

Where *j* represents the *j-th* PM stage. Substituting the above in both equations 4.33 and 4.34, gives equations 4.39 and 4.40 respectively.

$$UT = T_p - \frac{\mu_m}{\theta^{\beta}} \left| t^{\beta} \right|_{W_{j-1}^{+}}^{W_j} \tag{4.39}$$

$$DT = \mu + \frac{\mu_m}{\theta^{\beta}} \left| t^{\beta} \right|_{W_{j-1}^{+}}^{W_j} \tag{4.40}$$

Substituting equation 4.39 and 4.40 into equation 4.32 gives the following:

$$A_{ic} = \frac{T_p - \frac{\mu_m}{\theta^{\beta}} \left| t^{\beta} \right|_{W_{j-1}^{+}}^{W_j}}{\left(T_p - \frac{\mu_m}{\theta^{\beta}} \left| t^{\beta} \right|_{W_{j-1}^{+}}^{W_j}\right) + \left(\mu + \frac{\mu_m}{\theta^{\beta}} \left| t^{\beta} \right|_{W_{j-1}^{+}}^{W_j}\right)}$$

The above equation is only a reflection of the first PM stage, and therefore for *n* number of PM stages, it transforms into equation 4.41.

$$A_{ic} = \frac{\sum_{j=1}^{n} \left(T_p - \frac{\mu_m}{\theta^{\beta}} \left| t^{\beta} \right|_{W_{j-1}^{+}}^{W_j}\right)}{\sum_{j=1}^{n} \left[\left(T_p - \frac{\mu_m}{\theta^{\beta}} \left| t^{\beta} \right|_{W_{j-1}^{+}}^{W_j}\right) + \left(\mu + \frac{\mu_m}{\theta^{\beta}} \left| t^{\beta} \right|_{W_{j-1}^{+}}^{W_j}\right)\right]} \tag{4.41}$$

The unavailability of a component U_{ic} is therefore as expressed in equation 4.42.

$$U_{ic} = 1 - A_{ic} \tag{4.42}$$

4.2.2.3 IPM Cost

As mentioned earlier, maintenance has no standardized framework yet and this includes cost modelling. Several cost models exit in literature from simple to complex. Also for each problem domain, maintenance cost models may vary; for instance the cost model for production industry may differ from that for an aviation industry. In whatever scenario, a choice of what form of cost model to use may depend on the maintenance model under consideration. As the work in this thesis is not focused on a given problem domain (e.g. automotive, aviation, etc) the cost model to be established for the evaluation of PM schedule fitness is a generic one allowing for subsequent addition of specific parameters. The total cost of the *i-th* component of a system under IPM and taking minimal repair into account can therefore be expressed as shown in equation 4.43.

$$C_{ci} = C_{mri} \sum_{j=1}^{n} N(t) + n_i C_{pmi} + C_i \tag{4.43}$$

Where C_{ci} is the IPM total cost for the *i-th* component

C_{mri} is the cost of minimal repair for the *i-th* component

C_i is the unit cost of the *i-th* component

C_{pmi} is the cost of performing IPM for the *i-th* component at each PM stage

The derivation of N(t) is as expressed in equation 4.38, and so substituting for this in equation 4.43 gives equation 4.44.

$$C_{ci} = C_{mri} \sum_{j=1}^{n} \left(\frac{1}{\theta^{\beta}} \Big| t^{\beta} \Big|_{W_{j-1}^{+}}^{W_j} \right) + n_i C_{pmi} + C_i \tag{4.44}$$

The total IPM cost C_{is} for a system is the summation of all the total IPM cost of its constituent components. This is expressed in equation 4.45.

$$C_{is} = \sum_{i=1}^{m} \mathrm{C_{ci}} \tag{4.45}$$

Where: i is the index of the *i-th* component of the system

m is the number of components in the system identified for PM

4.2.2.4 System Reliability and Availability Calculation

Having calculated reliability and availability of components under maintenance, system reliability and unavailability are evaluated using the Esary-Proschan approximation (Jin and Coit 2003, eq 2.1) which is applied on the minimal cut sets of the fault trees produced for the system by a HiP-HOPS analysis. The Esary-Proschan approximation is shown in equation 4.46.

$$O_s = \prod_{i=1}^{n} \left(1 - \prod_{j=1}^{m} (1 - O_{ij}) \right) \tag{4.46}$$

Where: O_{ij} and O_s refer to the evaluated objective functions on component and system respectively. Such evaluated objective functions in this work are reliability and unavailability.

n is the number of cut sets and m the order of the *i-th* cut set.

Equation 4.46 provides the means by which availability and reliability are integrated as objectives of the optimisation performed by the GA as this is discussed in chapter 6.

4.3 Discussion

System reliability and availability largely depend on the reliability and availability of components. In turn, component reliability and availability depend both on manufacturing quality and maintenance. It is therefore common to first investigate the effectiveness of maintenance on the reliability of components and then to project results at system level.

This chapter has looked at relevant literature and developed generic models for calculating the reliability, availability and cost of a component under assumptions of PPM and IPM, and assuming a Weibull distribution describing the failure characteristics of the component. The Weibull models for reliability and unavailability under IPM are numerically validated in chapter 5. The established models are then applied and evaluated on case studies in chapter 7.

5 VALIDATION OF DERIVED MODELS FOR RELIABILITY AND UNAVAILABILITY

The derived Weibull model for component reliability and unavailability in equations 4.31 (on page 72) and 4.42 (on page 77) respectively are in this chapter numerically validated. Although the correctness of the derived Weibull models can be verified by checking their mathematical derivations and the assumptions about maintenance that underpin this model, it is also useful to perform a numerical validation as a confirmation that the mathematics is correct. The numerical validation consists of a comparison between results returned by the theoretical models derived in chapter 4 with results returned by calculations of reliability and unavailability from first principles.

5.1 Validation of the Established IPM Weibull Reliability Model

The validation of the Weibull model for component reliability under IPM is one which involves the process of evaluation of reliability from first principles. Results are then compared against those generated from the derived model of equation 4.31. The key to the evaluation is the use of equation 4.4 which is the fundamental model established in literature for component reliability under preventive maintenance. As mentioned on page 61, equation 4.4 shows that component reliability under preventive maintenance is a product of two reliabilities: (i) the probability of the component surviving until the current maintenance stage, and (ii) the probability of surviving the remaining time after the last maintenance stage. To evaluate the Weibull model for component reliability, all basic (or fundamental) models which form part of the derived model in equation 4.31 are used. To understand how equation 4.4 works, its two constituent parts are explained below.

(i) Probability of surviving until the current maintenance stage

The probability of surviving until the current maintenance stage refers to cumulative reliabilities from current maintenance stage down to all previous stages. For instance if the current maintenance stage is j and assuming that the maintenance policy started when component age was 0 (i.e. when component was new) then all previous maintenance stages range from $j-1$, $j-2$, $j-3$ down to $j-(j-1)$, $j-j$. Where in simple terms $j-$

(j-1) and *j-j* are 1 and 0 maintenance stages respectively. If $j = 1$, then the probability of surviving until the first maintenance is the cumulative reliability at $j = 0$ and $j = 1$. Similarly the reliability at j = 2 will consist of the product of reliability at $j = 0$, $j = 1$ and $j = 2$. The value of the reliability at 0 maintenance stage (i.e. when $j = 0$) is 1 since the maintenance policy is relative to when component was new. It should also be noted that for $j > 1$, the count to the *j-th* maintenance stage starts from the effective age at the *(j-1)-th* maintenance stage. This is to say that if the effective age at the *(j-1)-th* maintenance stage is W_{j-1}^{+} and the preventive maintenance interval is T_p, then the *j-th* maintenance is performed at $W_{j-1}^{+} + T_p$.

(ii) Probability of surviving the remaining time after the last maintenance stage

This second aspect of equation 4.4 refers to the probability of the component surviving after the last maintenance stage. This implies that the reliability of the component at any time after the last maintenance stage is evaluated relative to the component's effective age after the last maintenance stage. For illustrative purpose only assuming *x* is the effective age and *y* is a given time after the last maintenance stage then the reliability of the component between the two points *x* and *y* is an interval reliability i.e. R(*x*,*y*). To evaluate interval reliability, equation 4.25 is used.

In general, when the reliability of a given component is considered for two scenarios, under *no PM policy* and *under PM policy*, the variation in reliability values for the two scenarios lies in the age values at the considered time. Under no preventive maintenance policy, effective age does not exist and therefore reliability evaluation under this policy is straightforward. In contrast under preventive maintenance, component rejuvenation is possible and hence effective age is considered. Having established that the difference lies in age, the Weibull model for component reliability under no PM policy as seen in equation 4.9 (on page 63) is therefore also used in the evaluation of component reliability under PM from first principles. Further more, this is also justified by the fact that by using the proportional age reduction model, the fundamental difference that exists between *no PM* and *PM* policy is in terms of time (age). The age of the component after each maintenance stage is evaluated using equation 4.3 (on page 60) and this is also used in the evaluation from first principles.

A general age pattern for a component under preventive maintenance is illustrated in Figure 4.3 (on page 69). The figure also gives a pictorial view on how preventive maintenance can affect the age of a component and hence the needed considerations in the evaluation of its reliability at any given time. To validate the derived Weibull model for component reliability, both evaluations (first principle analysis and derived model) are performed under the following assumptions.

Improvement factor f = 0.875
Preventive maintenance time T_p = 30
Time scale θ = 300

The effort and time required for first principle analysis can not be underestimated and therefore a sufficient but relatively high time scale of 300 units (assumed in days) is considered. A summary of the validation is presented in Table 5.1 while its more detailed form in Table 5.2. The summary is only presented for the purpose of quick view and it is an extraction at maintenance stages from the detailed form. Hence Table 5.1 shows validations at time intervals of 0, 30, 90, .., 270, 300. In contrast the more detailed presentation of the validation in Table 5.2 shows reliability evaluations at time intervals with a time step of 5; i.e. 0, 5, 10, .., 295, 300. Due to the large number of columns in both tables, sub-headers are represented in symbols. However both the full names and their respective symbols are described as follows.

Time (t) - This is the time series for which reliability evaluation is considered.

Maintenance Stage - This is represented by j and it is the maintenance stage at any given time t. It is evaluated as the quotient of the division of t by the preventive maintenance interval T_p.

Effective Age at j-1 - This is appropriately represented as W_{j-1}^{+} however for simplicity in its subsequent use in the tables it is represented as *p1*.

Age at j - This is the component age at the *j-th* maintenance stage and it is appropriately represented as W_j and as it is with the case above, it is represented by *p2*.

Surviving until j-th Maintenance Stage - The reliability of surviving until the current maintenance stage consists of three sub-columns; previous reliability at jT_p, reliability between current stage *j* (*p2*) and effective age W_{j-1}^{+} at *j-1* (*p1*), and cumulative reliability $R1$, these are described as follows.

Previous Reliability at jT_p - This is denoted by R_{prev} and it is the component reliability evaluation at the *j-th* maintenance stage.

Reliability between Current Stage j (p2) and Effective Age W_{j-1}^{+} *at j-1 (p1)* - This is the reliability between the two stated intervals and it is represented by $R(p1, p2)$.

Cumulative Reliability $R1$ - This is the actual evaluation of the first part of equation 4.4 which is the cumulative reliability at current and previous maintenance stages. It is represented by $R1$. For any given maintenance stage, it is the product of R_{prev} and $R(p1, p2)$; i.e. $R1 = R_{prev}\, x\, R(p1, p2)$. At the next maintenance stage, the previous reliability will take the value of $R1$; i.e. at *(j+1)-th* maintenance stage $R_{prev} = R1$.

Surviving Remaining Time - This is the second part of equation 4.4 and also consists of three columns; effective age at *j*, time after last stage, and reliability between effective age at *j* (W_j^{+}) and time after last stage, these are described as follows.

Effective Age at j - This is represented by W_j^{+} and for simple referencing purposes within the table, it is denoted by $p3$.

Time after Last Stage - This is a given time since the last maintenance stage and it is denoted by $p4$.

Reliability Between Effective Age at j (W_j^+) and Time after Last Stage - This is the interval reliability for the stated range and it is denoted by $R2 = R(p3, p4)$. This is the evaluation which gives the probability of surviving the remaining time since the last maintenance stage.

Overall Reliability - This gives the Weibull model for component reliability at a given time. Thus, this is the column which contains the final result of first principle analysis. It is also the full evaluation of equation 4.4.

It can be observed from both tables, and in particular Table 5.2 that the two evaluations produce the same or almost the same values. Hence, it can be said that both evaluations produce the same approximation which then suggest the validity of equation 4.31.

Table 5.1 – First principle and derived model evaluations of component reliability at maintenance stages

First Principle Analysis											**Derived Model**
				Surviving until *j-th* Maintenance Stage			**Surviving Remaining Time**			**Overall Reliability**	
Time (t)	j	$p1 = W_{j-1}^{+}$	$p2 = W_j$	R_{prev}	$R(p1, p2)$	$R1$	$p3 = W_j^{+}$	$p4$	$R2$	$(R1\ x\ R2)$	
0	0	0	0	1	1	1	0	0	1	1	1
30	1	0	30	1	0.990049834	0.990049834	3.75	3.75	1	0.990049834	0.99005
60	2	3.75	33.75	0.990049834	0.987579741	0.977753159	7.5	7.5	1	0.977753159	0.977753
90	3	7.5	37.5	0.977753159	0.985121242	0.963205406	11.25	11.25	1	0.963205406	0.963205
120	4	11.25	41.25	0.963205406	0.982676614	0.946519427	15	15	1	0.946519427	0.946519
150	5	15	45	0.946519427	0.980248115	0.927823884	18.75	18.75	1	0.927823884	0.927824
180	6	18.75	48.75	0.927823884	0.977837977	0.907261429	22.5	22.5	1	0.907261429	0.907261
210	7	22.5	52.5	0.907261429	0.975448404	0.884986713	26.25	26.25	1	0.884986713	0.884987
240	8	26.25	56.25	0.884986713	0.973081568	0.861164258	30	30	1	0.861164258	0.861164
270	9	30	60	0.861164258	0.970739605	0.835966252	33.75	33.75	1	0.835966252	0.835966
300	10	33.75	63.75	0.835966252	0.968424615	0.809570296	37.5	37.5	1	0.809570296	0.80957

Table 5.2 – More detailed first principle and derived model evaluations of component reliability (from 0, 5, 10, .., 295, 300 time unit)

First Principle Analysis											**Derived Model**
				Surviving until *j-th* Maintenance Stage			**Surviving Remaining Time**			**Overall Reliability**	
Time (t)	j	$p1 = W_{j-1}^{+}$	$p2 = W_j$	R_{prev}	$R(p1, p2)$	$R1$	$p3 = W_j^{+}$	$p4$	$R2$	$(R1\ x\ R2)$	
0	0	0	0	1	1	1	0	0	1	1	1
5	0	0	0	1	1	1	0	5	0.999722261	0.999722261	0.999722
10	0	0	0	1	1	1	0	10	0.998889506	0.998889506	0.99889
15	0	0	0	1	1	1	0	15	0.997503122	0.997503122	0.997503
20	0	0	0	1	1	1	0	20	0.995565417	0.995565417	0.995565
25	0	0	0	1	1	1	0	25	0.993079612	0.993079612	0.99308
30	1	0	30	1	0.990049834	0.990049834	3.75	3.75	1	0.990049834	0.99005
35	1	0	30	0.990049834	0.990049834	0.990049834	3.75	8.75	0.999305905	0.989362645	0.989363
40	1	0	30	0.990049834	0.990049834	0.990049834	3.75	13.75	0.998057748	0.988126908	0.988127
45	1	0	30	0.990049834	0.990049834	0.990049834	3.75	18.75	0.996257607	0.986344678	0.986345
50	1	0	30	0.990049834	0.990049834	0.990049834	3.75	23.75	0.993908476	0.984018921	0.984019
55	1	0	30	0.990049834	0.990049834	0.990049834	3.75	28.75	0.991014254	0.981153498	0.981153
60	2	3.75	33.75	0.990049834	0.987579741	0.977753159	7.5	7.5	1	0.977753159	0.977753
65	2	3.75	33.75	0.977753159	0.987579741	0.977753159	7.5	12.5	0.9988902	0.976668048	0.976668
70	2	3.75	33.75	0.977753159	0.987579741	0.977753159	7.5	17.5	0.99722781	0.975042641	0.975043
75	2	3.75	33.75	0.977753159	0.987579741	0.977753159	7.5	22.5	0.995015595	0.972879641	0.97288
80	2	3.75	33.75	0.977753159	0.987579741	0.977753159	7.5	27.5	0.992257232	0.970182642	0.970183

85	2	3.75	33.75	0.977753159	0.987579741	0.977753159	7.5	32.5	0.988957293	0.966956117	0.966956
90	3	7.5	37.5	0.977753159	0.985121242	0.963205406	11.25	11.25	1	0.963205406	0.963205
95	3	7.5	37.5	0.963205406	0.985121242	0.963205406	11.25	16.25	0.998475534	0.961737032	0.961737
100	3	7.5	37.5	0.963205406	0.985121242	0.963205406	11.25	21.25	0.996400467	0.959738316	0.959738
105	3	7.5	37.5	0.963205406	0.985121242	0.963205406	11.25	26.25	0.993778246	0.957212579	0.957213
110	3	7.5	37.5	0.963205406	0.985121242	0.963205406	11.25	31.25	0.990613224	0.954164012	0.954164
115	3	7.5	37.5	0.963205406	0.985121242	0.963205406	11.25	36.25	0.986910641	0.950597664	0.950598
120	4	11.25	41.25	0.963205406	0.982676614	0.946519427	15	15	1	0.946519427	0.946519
125	4	11.25	41.25	0.946519427	0.982676614	0.946519427	15	20	0.998062295	0.944685351	0.944685
130	4	11.25	41.25	0.946519427	0.982676614	0.946519427	15	25	0.99557649	0.942332489	0.942332
135	4	11.25	41.25	0.946519427	0.982676614	0.946519427	15	30	0.992546711	0.939464744	0.939465
140	4	11.25	41.25	0.946519427	0.982676614	0.946519427	15	35	0.988977979	0.936086869	0.936087
145	4	11.25	41.25	0.946519427	0.982676614	0.946519427	15	40	0.984876192	0.932204449	0.932204
150	5	15	45	0.946519427	0.980248115	0.927823884	18.75	18.75	1	0.927823884	0.927824
155	5	15	45	0.927823884	0.980248115	0.927823884	18.75	23.75	0.997650868	0.925644303	0.925644
160	5	15	45	0.927823884	0.980248115	0.927823884	18.75	28.75	0.994756647	0.922958975	0.922959
165	5	15	45	0.927823884	0.980248115	0.927823884	18.75	33.75	0.991322134	0.919772352	0.919772
170	5	15	45	0.927823884	0.980248115	0.927823884	18.75	38.75	0.98735301	0.916089704	0.91609
175	5	15	45	0.927823884	0.980248115	0.927823884	18.75	43.75	0.982855825	0.911917109	0.911917
180	6	18.75	48.75	0.927823884	0.977837977	0.907261429	22.5	22.5	1	0.907261429	0.907261
185	6	18.75	48.75	0.907261429	0.977837977	0.907261429	22.5	27.5	0.997241636	0.904758872	0.904759
190	6	18.75	48.75	0.907261429	0.977837977	0.907261429	22.5	32.5	0.993941698	0.901764965	0.901765
195	6	18.75	48.75	0.907261429	0.977837977	0.907261429	22.5	37.5	0.990105646	0.898284664	0.898285

200	6	18.75	48.75	0.907261429	0.977837977	0.907261429	22.5	42.5	0.985739816	0.894323714	0.894324
205	6	18.75	48.75	0.907261429	0.977837977	0.907261429	22.5	47.5	0.980851394	0.889888637	0.889889
210	7	22.5	52.5	0.907261429	0.975448404	0.884986713	26.25	26.25	1	0.884986713	0.884987
215	7	22.5	52.5	0.884986713	0.975448404	0.884986713	26.25	31.25	0.996834978	0.88218571	0.882186
220	7	22.5	52.5	0.884986713	0.975448404	0.884986713	26.25	36.25	0.993132394	0.878908973	0.878909
225	7	22.5	52.5	0.884986713	0.975448404	0.884986713	26.25	41.25	0.988898368	0.875161916	0.875162
230	7	22.5	52.5	0.884986713	0.975448404	0.884986713	26.25	46.25	0.984139874	0.870950712	0.870951
235	7	22.5	52.5	0.884986713	0.975448404	0.884986713	26.25	51.25	0.978864729	0.866282279	0.866282
240	8	26.25	56.25	0.884986713	0.973081568	0.861164258	30	30	1	0.861164258	0.861164
245	8	26.25	56.25	0.861164258	0.973081568	0.861164258	30	35	0.996431267	0.858090993	0.858091
250	8	26.25	56.25	0.861164258	0.973081568	0.861164258	30	40	0.992329481	0.854558681	0.854559
255	8	26.25	56.25	0.861164258	0.973081568	0.861164258	30	45	0.987701403	0.850573146	0.850573
260	8	26.25	56.25	0.861164258	0.973081568	0.861164258	30	50	0.982554643	0.846140941	0.846141
265	8	26.25	56.25	0.861164258	0.973081568	0.861164258	30	55	0.976897633	0.841269325	0.841269
270	9	30	60	0.861164258	0.970739605	0.835966252	33.75	33.75	1	0.835966252	0.835966
275	9	30	60	0.835966252	0.970739605	0.835966252	33.75	38.75	0.996030876	0.832648199	0.832648
280	9	30	60	0.835966252	0.970739605	0.835966252	33.75	43.75	0.991533691	0.828888704	0.828889
285	9	30	60	0.835966252	0.970739605	0.835966252	33.75	48.75	0.986515843	0.824693952	0.824694
290	9	30	60	0.835966252	0.970739605	0.835966252	33.75	53.75	0.98098556	0.820070822	0.820071
295	9	30	60	0.835966252	0.970739605	0.835966252	33.75	58.75	0.974951878	0.815026868	0.815027
300	10	33.75	63.75	0.835966252	0.968424615	0.809570296	37.5	37.5	1	0.809570296	0.80957

5.2 Validation of the Established IPM Weibull Unavailability Model

The validation of the Weibull model for component unavailability under IPM also involves first principle analysis. Results using such analysis are compared with the results yielded by the derived model of equation 4.42. First principle analysis uses equation 4.32 (established in literature) on page 73 as the fundamental model where all other elements considered in the validation form a part. From equation 4.32, the validation of the unavailability model will require establishing the component's uptime (UT) and downtime (DT). The uptime is estimated using equation 4.39 while the downtime using equation 4.40 (both on page 76).

Basically, the downtime between the *(j-1)-th* and *j-th* PM stage is the total length of time the component has been in failed state. According to the PM policy used in this thesis (section 4.2.2.2 on page 72), each time the component has failed, a minimal repair is performed. Therefore the product of the number of failures and the minimal repair time will give the total downtime. Similarly, the uptime between the *(j-1)-th* and *j-th* PM stage is the total length of time the component was not in a failed state. Since the PM is periodic and at time T_p then the total uptime between the *(j-1)-th* and *j-th* PM stage will be T_p less the downtime within these stages. To estimate the number of failures between the *(j-1)-th* and *j-th* PM stage the effective age W_{j-1}^{+} at the *(j-1)-th* PM stage and the age $W_j = W_{j-1}^{+} + T_p$ at the *j-th* PM stage are considered as seen in equation 4.38 (on page 76). To evaluate the component unavailability at the *j-th* PM stage, the uptime and downtime from current PM stage down to all previous stages are summed and considered.

The same value for improvement factor, PM time T_p and time scale θ used in the validation of the Weibull model for component reliability is also used for the unavailability. Additional parameters such as MTTR and minimal repair time are assumed with values 4 and 2 respectively in hours. The first principle analysis is contained in Table 5.3 along side the results yielded by the derived mathematical model of equation 4.42. Unlike the fundamental model for reliability (equation 4.4) the fundamental model for availability does not model the remaining time after the last

maintenance stage. Therefore the two evaluations of unavailability are presented at PM stages. Table 5.3 is divided into two main headers; first principle analysis and derived model. Many of the sub headers are also same as those described for Table 5.1 and Table 5.2. Therefore only those that are unique to the unavailability evaluation are described below.

Number of Minimal Failures - This gives the number of failures which has occurred resulting into minimal repair between the previous effective age and the age at the current PM stage. It is symbolised by # *Minimal Failures*.

Minimal Failure Downtime - This is the length of the component downtime due to minimal repair. It is the product of the number of minimal failures and the time required to carrying out the repair.

Uptime - This sub header gives the uptime between the previous effective age and the age at the current PM stage.

Downtime - This gives the downtime between the previous effective age and the age at the current PM stage.

The *availability* and *unavailability* sub headers give the availability and unavailability of the component at the given PM stage. The preceding sub headers (or columns) establish the value of the parameters required for the evaluation of the component availability. This makes it possible to evaluate the component availability using the fundamental model shown in equation 4.31. The value of the unavailability is one less the value of the availability.

It can be observed from Table 5.3 that the two evaluations produce the same or almost the same values for the component unavailability. This then suggests the validity of the derived Weibull component unavailability model of equation 4.42.

Table 5.3 - First principle and derived model evaluations of component unavailability

First Principle Analysis										**Derived Model**
Time (t)	*j*	W_{j-1}^{+}	W_j	*# Minimal Failure*	*Minimal Failure Downtime*	*Uptime*	*Downtime*	*Availability*	*Unavailability*	*Unavailability*
0	0	0	0	0	0	0	0	1	0	0
30	1	0	30	0.01	0.00083333	29.99916667	0.167500	0.994447503	0.005552497	0.0055525
60	2	3.75	33.75	0.0125	0.001041663	59.99812501	0.335209	0.99444405	0.00555595	0.00555595
90	3	7.5	37.5	0.015	0.001249995	89.99687501	0.503126	0.994440597	0.005559403	0.0055594
120	4	11.25	41.25	0.0175	0.001458328	119.9954167	0.671251	0.994437144	0.005562856	0.00556286
150	5	15	45	0.02	0.00166666	149.99375	0.839585	0.994433691	0.005566309	0.00556631
180	6	18.75	48.75	0.0225	0.001874993	179.991875	1.008127	0.994430238	0.005569762	0.00556976
210	7	22.5	52.5	0.025	0.002083325	209.9897917	1.176877	0.994426785	0.005573215	0.00557322
240	8	26.25	56.25	0.0275	0.002291658	239.9875001	1.345836	0.994423332	0.005576668	0.00557667
270	9	30	60	0.03	0.00249999	269.9850001	1.515003	0.994419879	0.005580121	0.00558012
300	10	33.75	63.75	0.0325	0.002708323	299.9822917	1.684378	0.994416426	0.005583574	0.00558357

5.3 Chapter Summary

The derived Weibull model for component reliability and unavailability of chapter 4 were in this chapter numerically validated. The technique used was a comparison between the results returned by the theoretical models derived in chapter 4 with results returned by calculations of reliability and unavailability from first principles. Approximation in values retunred by the two processes suggests the validity of the models drived in chapter 4.

The evaluations of the reliability model was performed for a given time scale and at unit time intervals. The approximations as shown in Table 5.1 (as summary for quick view) and Table 5.2 (more detailed evaluation) suggest the validity of the derived reliability model.

The evaluations of the unavailability model was also performed and presented in Table 5.3. The evaluations were done at preventive maintenance stages. As it is the case with the reliability evaluations, the approximation in evaluations suggest the validity of the derived unavailability model. Hence, both derived models are used as evaluation functions for optimising preventive maintenance schedules under assumptions of imperfect preventive maintenance as defined in the next chapter.

6 THE APPROACH TO OPTIMISATION OF PREVENTATIVE MAINTENANCE

The purpose of PM optimisation is to derive maintenance schedules for a system that represent optimal or near optimal trade-offs among predefined objective functions. The strategy for optimisation plays an important role in ensuring that the derived PM schedules meet design requirements. This implies that objective functions must be carefully selected. In this work, system unavailability (failure probability) and cost have been selected as the objective functions of the optimisation. Cost is a key requirement and failure probability can support evaluation of other requirements including reliability and availability. However, it should be stressed that the novel PM optimisation approach developed here (in particular this chapter) is generic and more objectives could be added, for example safety, weight or other attributes that may be important in the design of a particular system for a particular application.

Approaches to maintenance optimisation differ in terms of maintenance models and the optimisation strategy adopted. A system PM schedule comprises of PM schedules of its respective constituent components that have been identified for PM.

As mentioned in the introduction, HiP-HOPS is a technique that offers mature dependability analysis and optimisation options (Parker, 2010) which enable optimisation of the architecture of a system model through replication of components in the model. It therefore provides a good basis upon which to build further work on optimising maintenance. For the purpose of optimisation, HiP-HOPS implements two genetic algorithms; PESA II and NSGA II. Following the comparison between these algorithms in chapter 3, NSGA II was selected as a basis for adaptation and for building the work developed in this thesis. Not only has NSGA II demonstrated several advantages in the general literature, but according to Parker (2010) who applied both algorithms to a dependability optimisation problem, the inability to intuitively set a good hyper-grid size for the PESA II GA represented a significant problem for its use. Even after a good hyper-grid size was found using costly trial and error, PESA II did

not achieve good solutions when compared to NSGA II which in this case produced excellent results.

The general procedure for NSGA II has been described in chapter 3. The specific adaptation of the algorithm for solving the PM optimisation problem is given in this chapter. Specifically, this chapter describes the approach that has been devised to perform the following.

- Encoding of a system model for PM
- Generating population of encodings
- Evaluation of solutions
- Breeding from potential solutions
- Defining the PM scheduling optimisation problem and production of schedules

6.1 PM Encoding

A PM schedule is a set of time intervals T_{pi} where each T_{pi} is the time at which PM activities are performed on the *i-th* component of the system model under consideration. For a system model with *m* number of components identified for PM, the following is a typical representation of PM schedule (PMS).

$$PMS = \{T_{p1}, T_{p2}, T_{p3}, T_{p4}, T_{p5}, T_{p6}, .. , T_{pm}\}$$

Each T_{pi} is a two parameter function consisting of (i) the shortest PM interval T, and (ii) the CoMI α_i of the *i-th* component. This relationship is already shown in equation 4.1 (on page 59). Since T is a constant whereas the CoMI α_i is an integer variant, PMS can be represented in a simple form as follows.

$$PMS = \{\alpha_1, \alpha_2, \alpha_3, \alpha_4, \alpha_5, \alpha_6, .. , \alpha_m\}$$

Hence the type of encoding adopted for this work is an integer encoding of CoMIs of the constituent components of the system model. Representing PMS in the above format

ensures that PM is carried out at an interval that is a multiple of *T* which is a simplification but a reasonable assumption to make. The format also reduces the number of digits required compared to encoding the PM time and hence gives a simplified representation for encoding the problem. An example of a typical PM encoding for a system model, where *m = 10* is shown below.

PMS = {3, 6, 12, 4, 7, 9, 1, 3, 5, 2}

In genetic algorithm terms, the above representation is referred to as individual, and in this work the term "PM individual" will be used from now onwards. Each component of the system is mapped onto its CoMI implying a one to one relation existing between each CoMI and a component. To illustrate this, let Figure 6.1 be a representation for a component-CoMI relation. Formally, the list of components in Figure 6.1 is referred to as *domain*, while the CoMIs as *range*. A close look at Figure 6.1 shows that the range contains two occurrences of CoMI value 3; however domain elements C_1 and C_8 can not be mapped unto the common or single range element 3. This is because their respective range elements are generated from two different failure data (such as the MTTF or MTBF of their domain elements). Also, these respective range elements are likely to differ in other variants of the system model. This is to say that the elements of the range are specific to a particular PM individual.

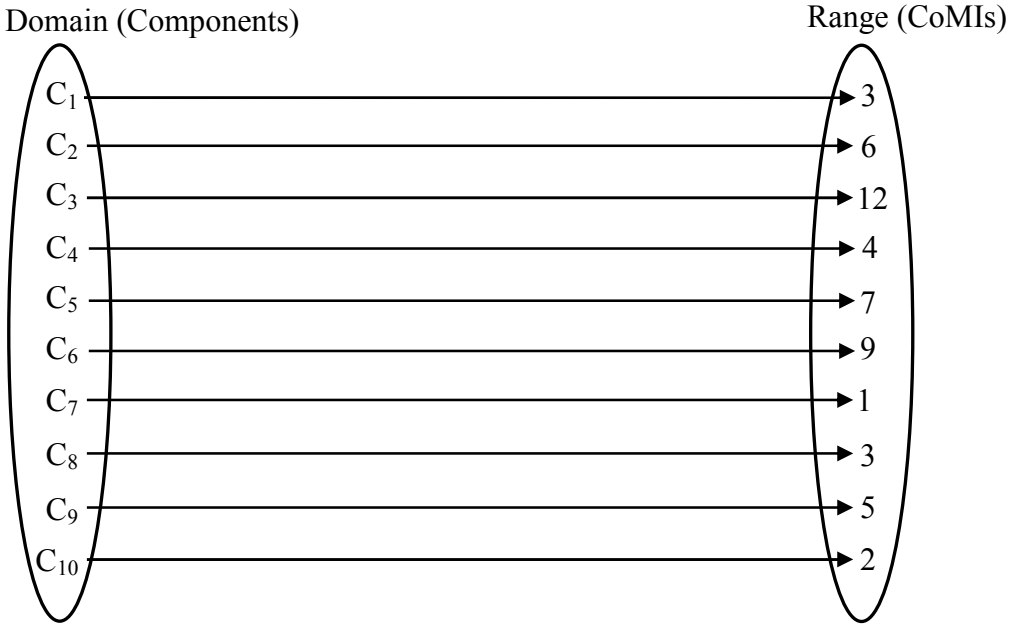

Figure 6.1 - One to one mapping between components and CoMIs

6.2 Generating a PM Individual

The initial PM individual population is generated randomly, as is the norm in optimisations of this type. Constraints may be used to define a feasible region ***f*X** within the total solution space **X,** and to guide population generation towards this feasible region which contains solutions that meet the constraints. In this work two categories of constraints are identified and developed; *primary* and *secondary*. The primary constraints consist of those constraints that are necessary to ensure that PM times for all components do not occur too early or late. Secondary constraints consist of those additional and optional constraints that are defined to further improve design objectives or to simply meet the requirements of a given PM policy.

6.2.1 Primary Constraints

One of the challenges of scheduling PM is to ensure that the maintenance does not occur too late, when the reliability of components has dropped too much. An opposite challenge is also not to schedule the PM too early, when the reliability of components is high and maintenance simply means incurring unnecessary cost. In order to ensure that these do not happen, the following primary constraints are imposed on the value of the shortest PM interval and CoMIs respectively.

$$T < \frac{1}{\lambda_H} \qquad (C1)$$

$$\alpha_i T \leq \begin{cases} \frac{1}{\lambda_i} & ; ATF_i \leq RT \\ RT & ; ATF_i > RT \end{cases} \qquad (C2)$$

Where: λ_H is the average failure rate of the component that fails most frequently

λ_i is the average failure rate of the *i-th* component

RT is the system risk time as mentioned in chapter 4

ATF in this work is the average time failure for the *i-th* component. For a PPM policy, *ATF* is synonymous to MTTF, and to MTBF for IPM.

The first constraint (C1) ensures that the shortest PM interval T is smaller than the ATF of the component that fails most frequently in the system. This ensures that maintenance is not carried out too late. The second constraint (C2) is an umbrella to two conditional constraints and ensures that the PM time (T_{pi}) for the *i-th* component is (i) not greater than its average time failure (ATF_i); if ATF_i is less or equal to the system risk time, or (ii) not greater than the system risk time; if ATF_i is greater than the system risk time.

Constraint C1 is fairly straightforward. C2 actually ensures that the genetic representation of a PM schedule falls within the feasible region $\boldsymbol{fX}$ of the solution space **X**. This is necessary to efficiently guide the encoding of PM individuals. Enforcing C2 requires further step; the maximum CoMI α_{max} for any given component must be determined. The CoMI α_i for the *i-th* component should be a value such that $1 \leq \alpha_i \leq \alpha_{imax}$, α_{imax} being the maximum CoMI for the *i-th* component which is evaluated as expressed in equation 6.1.

$$\alpha_{imax} = \begin{cases} Q\left(\frac{ATF}{T}\right) & ;\ ATF_i \leq RT \\ Q\left(\frac{RT}{T}\right) & ;\ ATF_i > RT \end{cases} \tag{6.1}$$

Where: Q is the integer quotient of the division

Hence in generating a PM individual, each constituent CoMI is a random integer between 1 and α_{imax}, inclusive.

6.2.2 Secondary Constraints

Secondary constraints can be seen as non mandatory constraints and are therefore optional. The number of secondary constraints used in this work is limited to two and their activation is based on a Boolean flag. These two constraints are *expert judgement* and *architecture modification through component substitution* (or simply component substitution).

6.2.2.1 Expert Judgement

The pattern of operation, degradation and ultimately failure of a component may become familiar over a period of exposure to operation. This knowledge and, or experience could sometimes be preferred over a stochastic process that uses probabilities to determine the PM time for the component in consideration. In principle, system components are designed to operate under specified conditions, in practice however not all of these conditions may be met while in operation and this can affect the reliability of the component. Expert judgement therefore relies on the knowledge and experience of the system engineer on a given system and its specified constituent component(s). Another scenario where expert judgement plays an important role in determining component PM time is when the required elements of the failure data that are necessary for the specified PM policy are unavailable. These required elements could be MTTF or MTBF and, or useful life.

When both the required elements of a component failure data and expert judgement exist, the latter overrides the former, and this characteristic will be considered in defining the PM optimisation. Each component identified for PM is initialised with expert time ET = 0, which implies that as long as this value remains unchanged, expert judgement is inactive on such component. The expert judgement constraint is therefore defined as a bi-implication (also known as biconditional) connective. A bi-implication connective is in the form $a \Leftrightarrow b$, meaning "a is true if and only if b is true" where a and b are expressions (Diller, 1999). The secondary constraint is hence defined as follows.

$$\left(\alpha_{ie}T \leq \begin{cases} ET_i & ; ET_i \leq RT \\ RT & ; ET_i > RT \end{cases}\right) \Leftrightarrow \left((expert\_judgement_i = true) \wedge (ET_i \geq T)\right) \quad (C3)$$

Where: α_{ie} is the CoMI of the *i-th* component derived from the expert judgement time

ET_i is the expert judgement time for the *i-th* component

$expert\_judgement_i$ is a Boolean variable that is flagged true for a component that is identified for expert judgement and false otherwise

When an expert specifies a PM time, equation 6.2 ensures that this time is a multiple of T, and if not then converted to such. The CoMI α_{ie} for the *i-th* component under expert judgement is evaluated as expressed in equation 6.2.

$$\alpha_{ie} = \begin{cases} Q\left(\frac{ET_i}{T}\right) & ; ET_i \leq RT \\ Q\left(\frac{RT}{T}\right) & ; ATF_i > RT \end{cases} \quad (6.2)$$

6.2.2.2 Architecture Modification through Component Substitution

Each component within a system model can have several design or implementation options; for example, a sensor may be procured from two different suppliers and the two versions will have different failure and cost characteristics. The set of component design options are here referred to as *implementation options*. Architecture modification through component substitution simply refers to modifying a system model by substituting implementations of components with alternatives. Implementation *x* is an alternative of *y* if $x \neq y$ and both *x* and *y* are members of same basic (component) type. This concept is illustrated in Figure 6.2 where component C_4 can be seen to have three possible implementations. Allowing component substitution in the context of PM optimisation enables one to determine which of the respective implementation options will better meet design requirements with respect to the given PM policy. This is precisely the reason that this option is included in this work.

In Figure 6.2 each component is represented in the form C_{i,k_i}, where *i* is the index of the *i-th* component, and is unique for all components. Every component comprises of an active implementation in the design, k_i is the index of the active implementation of component *i* among its available options (implementation options); k_i will here be referred to as *implementation option index*. Where a component has no alternatives, the value of k_i is *0* ($k_i = 0$); for instance components $C_{1,0}$, $C_{2,0}$ and $C_{3,0}$ in Figure 6.2.

The *4-th* component in Figure 6.2 has three implementation options, i.e. $C_{4,1}$, $C_{4,2}$ and $C_{4,3}$ implying that implementation options $IO_4 = \{C_{4,1}, C_{4,2}, C_{4,3}\}$, among these the

active implementation is $C_{4,1}$ and hence $AI_4 = \{C_{4,1}\}$. The set of alternatives to the active implementation consist of all those in the implementation options with the exclusion of the active one; hence, $AL_4 = \{C_{4,2}, C_{4,3}\}$. Connections to non active implementations are indicated in Figure 6.2 by doted lines. Only one out of the implementation options can be active.

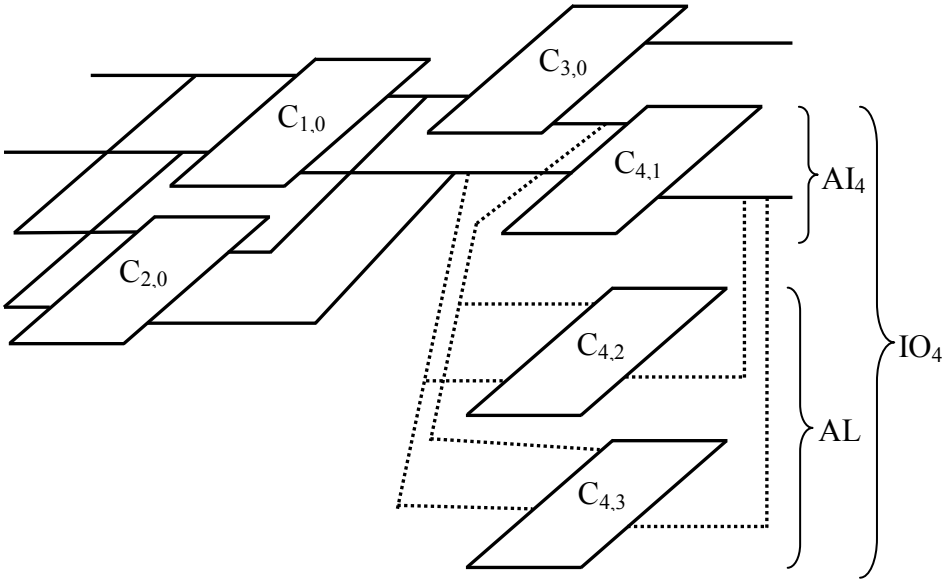

Figure 6.2 - Concept of architecture modification under PM

When optimising PM schedules, an active implementation for the *i-th* component with implementation option index greater than 0 (i.e. $k_i > 0$) implies that its alternatives should be considered for PM optimisation. This is possible if the system engineer has flagged the component for substitution. The value of $k_i \neq 0$ is true if the *i-th* component has implementation options. To substitute a component, one out of the implementation options is randomly selected and the current active is moved from set of active implementation to the set of alternatives. If the selection is restricted to the set of alternatives, the optimisation in its entirety will excluded the first implementation that has been manually denoted by the engineer as active. The purpose of the optimisation is to infuse diversity as much as possible in the variants of the system model. Therefore the current active implementation is granted chance to being selected, by choosing from the set of implementation options as opposed to the alternatives only. The selected

implementation is then flagged active and moved to the set of active implementation. Hence the following is established.

Let IO_i be the set consisting of implementation options for the *i-th* component within the system model

h_i be the number of implementations in IO_i

AI_i be the set of active implementation within IO_i

AL_i be the set of all implementations in IO_i excluding the element in AI_i

Then:

$$IO_i = \bigcup_{k_i=1}^{h_i} C_{i,k_i} \qquad ; defines\ set\ of\ implementation\ options$$

$$AI_i \subset IO_i \mid \exists_1 x \in IO_i \bullet x \qquad ; defines\ the\ set\ of\ active\ alternatives$$

$$AL_i \subset IO_i \mid \forall x \in IO_i \wedge x \notin AI_i \bullet x \qquad ; defines\ the\ set\ of\ alternatives$$

The process of substitution can be defined using a procedure in the form of a pseudo-code, such that it can be referenced in the definition of the PM optimisation problem and the modified NSGA II. Similar approaches of using pseudo-code for specific processes which are later invoked in algorithms have been used by Deb et al (2000), Favuzza et al (2006) and, MacFarlane and Tuson (2009). The pseudo-code is identified as *substitute_component* and receives two arguments. The first is the index *i* of the component under consideration, followed by the index k_i of the current active implementation of the *i-th* component. Within the pseudo-code, the index (*new_k_i*) for the new active implementation is randomly selected and its CoMI also randomly selected in-between 1 and its maximum value α_{imax,new_k_i} and returned.

substitute_component(i, k_i)	
new_k_i = random(1..h_i)	randomly selects the index of one of the implementations within the implementation options
AL_i = AL_i ∪ {C_{i,k_i}}	adds current active implementation to the set of alternatives
AI_i = AI_i \ {C_{i,k_i}}	removes current active implementation from the set of active implementation
AI_i = AI_i ∪ {C_{i,new_k_i}}	adds the new active implementation to the set of active implementation
AL_i = AL_i \ {C_{i,new_k_i}}	removes new active implementation from the set of alternatives
α_{i,new_k_i} = random(1.. α_{imax,new_k_i})	obtains a new CoMI from substituted implementation
return α_{i,new_k_i}	returns the new CoMI

Having established the pseudo-code, the constraint is expressed as seen in C4. The substitution is performed when two conditions are met; (i) if the system engineer has flagged the *i-th* component for substitution (i.e. $substitute_i = true$), and (ii) if the *i-th* component has implementation options.

$$substitute\_component(i, k_i) \Leftrightarrow \big((substitute_i = true) \wedge (k_i > 0)\big) \qquad (C4)$$

The assumption here is that in designing the system, the engineer has a first hand preference on the choice of active implementations for respective components of the system. Where possible, the engineer may however like to explore within the space of available implementation options, which ones are likely to better improve the design.

6.3 Defining the PM Optimisation Problem

Having established an appropriate PM encoding and constraints, the PM optimisation problem can now be defined. The definition is done in two phases; phase 1 is defined with respect to the primary constraints and therefore forms the fundamental definition, while phase two includes both the primary and secondary constraints. These separate definitions are necessary in order to show and to understand how the format of the decision variables (CoMI) changes when using the secondary constraints. Phase two therefore is a composite definition of the problem which could be applied to a problem with either primary or both primary and secondary constraints.

In addition, it is possible to optimise the PM schedules of a system model under combination of primary and one of secondary constraints (expert judgement or component substitution) using the composite definition. This can be done by suppressing either of the secondary constraints as desired; for instance setting its Boolean flag to false.

6.3.1 Definition of the PM Scheduling Optimisation from Fundamentals

The definition of PM scheduling optimisation using the primary constraints, also here referred to as the definition of PM optimisation from fundamentals is expressed as follows.

$$min\ \mathbf{F}(\boldsymbol{\alpha}) = \{ U(\boldsymbol{\alpha}), C(\boldsymbol{\alpha}) \}$$

such that:

$$\boldsymbol{\alpha} \in \mathbf{A},$$

$$T < \frac{1}{\lambda_H},$$

$$\alpha_i T \leq \begin{cases} \frac{1}{\lambda_i} & ; ATF_i \leq RT \\ RT & ; ATF_i > RT \end{cases}$$

Where: **A** is the solution space of all CoMIs.

$\boldsymbol{\alpha} = \{\alpha_1, \alpha_2, \alpha_3, .. \alpha_{m-2}, \alpha_{m-1}, \alpha_m\}$

m is the number of components identified for PM

U is the system unavailability

C is the system cost

The objective functions (U and C) are detailed as follows.

$$U(\boldsymbol{\alpha}) = O_s(\boldsymbol{\alpha}) = \begin{cases} \{U_{pc}(\alpha_1), U_{pc}(\alpha_2), .. U_{pc}(\alpha_m),\} & ; under\ PPM \\ \{U_{ic}(\alpha_1), U_{ic}(\alpha_2), .. U_{ic}(\alpha_m),\} & ; underIPM \end{cases}$$

Where: U_{pc} is the component unavailability under PPM as expressed in equation 4.16

U_{ic} is the component unavailability under IPM as expressed in equation 4.42

O_s is the system unavailability as expressed in equation 4.46

$$C(\boldsymbol{\alpha}) = \begin{cases} C_{ps}(\boldsymbol{\alpha}) = \{C_{pc1}(\alpha_1), C_{pc2}(\alpha_2), .. C_{pcm}(\alpha_m),\} & ; under\ PPM \\ C_{is}(\boldsymbol{\alpha}) = \{C_{c1}(\alpha_1), C_{c2}(\alpha_2), .. C_{cm}(\alpha_m),\} & ; underIPM \end{cases}$$

Where: C_{ps} is the system cost under PPM as expressed in equation 4.18

C_{pci} is the cost for the *i-th* component under PPM as expressed in equation 4.17

C_{is} is the system cost under IPM as expressed in equation 4.45

C_{ci} is the cost for the *i-th* component under IPM as expressed in equation 4.44

6.3.2 Composite Definition of the PM Scheduling Optimisation

The composite however encapsulates both primary and secondary constraints, and therefore represents CoMIs in the secondary constraints format i.e. α_{i,k_i}, where *i* refers to the *i-th* component and k_i the index value of the current active implementation. The composite definition of the PM scheduling optimisation problem is expressed as follows.

min $\mathbf{F}(\boldsymbol{\alpha}) = \{ U(\boldsymbol{\alpha}), C(\boldsymbol{\alpha}) \}$

such that:

$\boldsymbol{\alpha} \in \mathbf{A}$,

$T < \frac{1}{\lambda_H}$,

$$\left[\begin{array}{l} \left(\left(\alpha_{ie} T \leq \begin{cases} ET_i & ; ET_i \leq RT \\ RT & ; ET_i > RT \end{cases} \right) \Leftrightarrow \left(\left(expert_{judgement_i} = true \right) \wedge (ET_i \geq T) \right) \right) \vee \\ \left(\alpha_{i,k_i} T \leq \begin{cases} \frac{1}{\lambda_{i,k_i}} & ; ATF_{i,k_i} \leq RT \\ RT & ; ATF_{i,k_i} > RT \end{cases} \right) \end{array}\right],$$

$$substitute\_component(i, k_i) \Leftrightarrow \left((substitute_i = true) \wedge (k_i > 0) \right)$$

Where: $\boldsymbol{\alpha} = \{\alpha_{1,k_1}, \alpha_{2,k_2}, \alpha_{3,k_3}, .. \alpha_{m-2,k_{m-2}}, \alpha_{m-1,k_{m-1}}, \alpha_{m,k_m}\}$

i,k_i:$\mathbb{N}$ • $k_i = 0 \vee k_i$ = random(*1..*h_i) ; each k_i for the *i-th* component is either 0 or a random number between 1 and the number of its implementations

The specified constraints for the composite definition can be written concisely as C1, (C3 ∨ C2), C4. As mentioned earlier, an expert judgement overrides the PM time obtained from component failure data through probabilistic analysis, and this is why constraints C2 and C3 are specified as logical combinations i.e. (C3 ∨ C2).

The objective functions (*U* and *C*) are then detailed as follows.

$$U(\boldsymbol{\alpha}) = O_s(\boldsymbol{\alpha}) = \begin{cases} \{U_{pc}(\alpha_{1,k_1}), U_{pc}(\alpha_{2,k_2}), .., U_{pc}(\alpha_{m,k_m})\} & ; under\ PPM \\ \{U_{ic}(\alpha_{1,k_1}), U_{ic}(\alpha_{2,k_2}), .. , U_{ic}(\alpha_{m,k_m})\} & ; underIPM \end{cases}$$

$$C(\boldsymbol{\alpha}) = \begin{cases} C_{ps}(\boldsymbol{\alpha}) = \{C_{pc1}(\alpha_{1,k_1}), C_{pc2}(\alpha_{2,k_2}), .. , C_{pcm}(\alpha_{m,k_m})\} & ; under\ PPM \\ C_{is}(\boldsymbol{\alpha}) = \{C_{c1}(\alpha_{1,k_1}), C_{c2}(\alpha_{2,k_2}), .. , C_{ci}(\alpha_{m,k_m})\} & ; underIPM \end{cases}$$

6.4 Diversity in PM Encoding

Diversity in the PM encoding is introduced via recombination which is performed through classic genetic operators: crossover and mutation. This way, more variants of the PM encoding are created and evolved. The aim of diversifying the encoding is to progressively populate an evenly spread of the Pareto front by widening the search. In the course of injecting diversity in PM encoding, constraints imposed on the optimisation may be taken into account, and this is addressed in this section.

6.4.1 Evolving a New PM Encoding from Existing Ones

New PM individuals are created when all PM individuals have been evaluated against unavailability and cost, and then ranked according to NSGA II ranking scheme described in chapter 3. A tournament selection is used and the lowest PM individual in rank emerges as the winner for recombination. Two PM individuals are required for the recombination and therefore tournament selection is performed twice with each returning a PM individual. The returned PM individuals are referred to as parents.

The recombination process is described next in terms of the imposed constraints on the optimisation.

6.4.1.1 Based on Primary Constraints

Under primary constraints, the recombination of PM individuals is fairly simple and straightforward. Two parents are recombined using uniform crossover, with each child locus offering each of the parents the chance to contribute corresponding gene to that location. The gene contribution is randomly selected at each locus; with crossover probability $P_{ci} < 0.5$ parent one (P_1) contributes its corresponding *i-th* gene (CoMI) to locus i of the child. Alternatively, with probability $P_{ci} \geq 0.5$ parent two (P_2) does the contribution.

The standard notation for CoMI in this work has been the Greek alphabet α (alpha) however, in order to exemplify genetic operations on the two parents, it is necessary to draw clear distinction between CoMIs of P_1 and P_2. Therefore the English alphabets "a" and "b" are here used to represent CoMIs from P_1 and P_2 respectively. Figure 6.3 is the illustration of this example for a system with 12 components.

P_1	$a_{1,0}$	$a_{2,0}$	$a_{3,0}$	$a_{4,0}$	$a_{5,0}$	$a_{6,0}$	$a_{7,0}$	$a_{8,0}$	$a_{9,0}$	$a_{10,0}$	$a_{11,0}$	$a_{12,0}$
P_2	$b_{1,0}$	$b_{2,0}$	$b_{3,0}$	$b_{4,0}$	$b_{5,0}$	$b_{6,0}$	$b_{7,0}$	$b_{8,0}$	$b_{9,0}$	$b_{10,0}$	$b_{11,0}$	$b_{12,0}$

Figure 6.3 - Typical CoMIs from parents to be recombined

Assuming no parent genes are returned in succession by the random selection, then the resultant child C beginning with contribution from P_1 following uniform crossover is as shown in Figure 6.4.

C	$a_{1,0}$	$b_{2,0}$	$a_{3,0}$	$b_{4,0}$	$a_{5,0}$	$b_{6,0}$	$a_{7,0}$	$b_{8,0}$	$a_{9,0}$	$b_{10,0}$	$a_{11,0}$	$b_{12,0}$

Figure 6.4 - Child CoMIs following uniform crossover

With probability P_m the child C is mutated by randomly selecting a locus i. The current value of CoMI $\alpha_{i,0}$ is substituted with a value v; $1 \leq v \leq \alpha_{imax,0}$, where v is randomly selected.

6.4.1.2 Based on Secondary Constraints

Under secondary constraints, recombination takes into account two cases; (i) expert judgement, and (ii) architecture modification through component substitution. These two are here discussed.

Case i - Expert judgement

The assumption under expert judgement is that the system engineer has a clear judgement on the best PM time for a given component based on previous knowledge and experience. Crossover under expert judgement is not different from what is obtained in the primary constraints. However, mutation is disallowed on a locus of a component that is subjected to expert judgement; this is to maintain the integrity of such judgement. If such locus is selected, then no mutation is performed on the PM individual.

Case ii - Architecture modification through component substitution

Recombination involving component substitution among alternative implementations of components is performed in similar way as under primary constraints. This is further illustrated as seen in Figure 6.5. Assuming components 4, 7 and 8 shaded in Figure 6.5 have been flagged for component substitution and that each has three implementation options, Figure 6.6 is the resultant child obtained from recombining P_1 and P_2 using uniform crossover beginning with gene contribution from P_1.

P_1	$a_{1,0}$	$a_{2,0}$	$a_{3,0}$	$a_{4,2}$	$a_{5,0}$	$a_{6,0}$	$a_{7,2}$	$a_{8,3}$	$a_{9,0}$	$a_{10,0}$	$a_{11,0}$	$a_{12,0}$
P_2	$b_{1,0}$	$b_{2,0}$	$b_{3,0}$	$b_{4,1}$	$b_{5,0}$	$b_{6,0}$	$b_{7,3}$	$b_{8,1}$	$b_{9,0}$	$b_{10,0}$	$b_{11,0}$	$b_{12,0}$

Figure 6.5 - Typical CoMIs from parents under component substitution

C	$a_{1,0}$	$b_{2,0}$	$a_{3,0}$	$b_{4,1}$	$a_{5,0}$	$b_{6,0}$	$a_{7,2}$	$b_{8,1}$	$a_{9,0}$	$b_{10,0}$	$a_{11,0}$	$b_{12,0}$

Figure 6.6 - Child CoMIs from uniform crossover under component substitution

With probability P_m the child C is also mutated by randomly selecting a locus i. The current value of CoMI α_{i,k_i} is substituted with a value v; $1 \leq v \leq \alpha_{imax,k_i}$, where v as mentioned earlier is randomly selected.

6.5 NSGA II for PM Scheduling

To optimise the PM schedules of a system, a variant of the NSGA II is developed. This variant takes into account the identified constraints and objective functions. The mechanics of the adapted algorithm using HiP-HOPS are here discussed. The algorithm first generates a random initial population P of N number of PM individuals, with each individual represented as p. The following steps are then executed:

i. Set population index t = 1.

ii. Set front index i = 1.

iii. Randomly generate an initial population P_t of N number of PM individuals. This is performed in two steps as follows (1) If a given component i qualifies for component substitution, then $\alpha_{i,k_i} = substitute\_component(i, k_i)$ (2) if the component qualifies for expert judgement then $\alpha_{i,k_i} = \alpha_{ie,k_i}$ else $\alpha_{i,k_i} = random(1..\alpha_{imax,k_i})$.

iv. $\forall p \in P_t$, configure the variant of the system model with p by using the encoding to set the CoMI of each component and then evaluate the unavailability and cost (objective functions) of the system by calling the automatic fault tree synthesis and analysis functions of HiP-HOPS.

v. $\forall p \in P$, find n_p number of solutions that dominate p, and S_p set of solutions for which p dominates.

vi. Add all p with $n_p = 0$ into the set F_i (the i-th front) and assign domination rank R_p = i.

vii. For each $p \in F_i$ assign crowding distance to p.

viii. Increment front index by 1; i.e. i = i + 1.

ix. For each $p \in F_{i-1}$, visit each $q \in S_p$ and decrement n_q by 1, if by doing so, n_q becomes 0 then add q into the set F_i (q belonging to front i, R_q = i).

x. Repeat step viii to find subsequent fronts.

xi. Perform recombination as follows ("a – j" below) taking constraints into account; the process has already been described in details in section 5.4.1 "evolving a new PM encoding from existing ones."

(a) Set child population $Q_t = \emptyset$.

(b) Use binary tournament selection to select two parents from population P_t; as described in section 5.4.1.

(c) With probability P_c, perform uniform crossover on the selected parents to evolve with a child p.

(d) With probability P_m, perform mutation in one of the following ways; (1) if the selected locus *i* corresponds to a component that has been flagged for expert judgement (i.e. $expert\_judgement_i = true$) and $ET_i \geq T$ then exit to step "e" below, else (2) for a component under primary constraints or does not qualify for expert judgement perform normal mutation.

(e) Add p to Q_t; i.e. $Q_t = Q_t \cup p$.

(f) If the size of Q_t is not equal to N, then go to step "b".

(g) $\forall p \in Q_t$, configure the variant of the system model with p. The values of objective functions (unavailability and cost) are also calculated.

(h) P_t and Q_t are combined into B_t; i.e. $B_t = P_t \cup Q_t$ and B_t is sorted based on non-domination.

(i) From 2N solutions (combination of P_t and Q_t) in B_t, N best solutions are selected using the crowding calculation and comparison to form P_{t+1}.

(j) Increment population index by 1; i.e. t = t + 1.

xii. If maximum generation is not reached then go to iv else terminate giving the set of PM individuals in the first front F_1 as the solution.

6.6 PM Optimisation Space

The optimisation space in preventive maintenance defines the number of all potential PM individuals, within which lies the subspace of feasible PM individuals that meet constraints. The space occupied by all potential scheduled PM individuals is termed as PM solution space **X** while the subspace of feasible PM individuals is termed as feasible PM region $f\mathbf{X}$. The relationship between the two is defined as $f\mathbf{X} \subset \mathbf{X}$.

6.6.1 PM Solution Space

The solution space in a PM scheduling problem can be large and can contain an enormous number of potential PM schedules. Not all these schedules are valid though if they violate some of the constraints of the optimisation.

6.6.2 Feasible PM Region

The feasible PM region defines the space containing the population of scheduled PM individuals that are feasible solutions. The size of the feasible PM region is influenced by the constraints imposed. To illustrate the evaluation for the number of feasible PM individuals, Figure 6.7 is considered. For simplicity, Figure 6.7 is a system model of 3 components, C_1, C_2 and C_3, with the assumption that all the components have a maximum CoMI of 3 each.

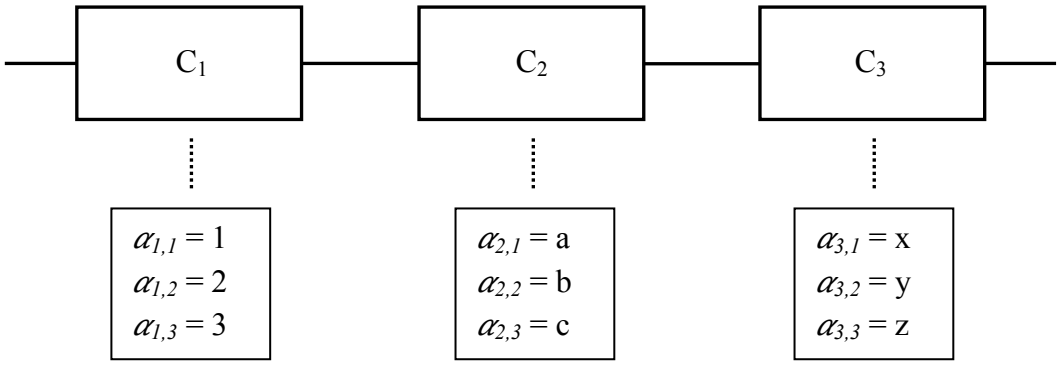

Figure 6.7 - Illustration of a system model with its CoMIs

For easy identification of the parent component of a CoMI, the CoMIs in Figure 6.7 are distinct in representation, such that no two or more components have common CoMI. The feasible PM individuals in Figure 6.7 through enumeration are:

{1, a, x}, {1, a, y}, {1, a, z}, {1, b, x}, {1, b, y}, {1, b, z}, {1, c, x}, {1, c, y}, {1, c, z}
{2, a, x}, {2, a, y}, {2, a, z}, {2, b, x}, {2, b, y}, {2, b, z}, {2, c, x}, {2, c, y}, {2, c, z}
{3, a, x}, {3, a, y}, {3, a, z}, {3, b, x}, {3, b, y}, {3, b, z}, {3, c, x}, {3, c, y}, {3, c, z}

The above reveals that the number of PM individuals #($\boldsymbol{f}\mathbf{X}$) within the feasible PM region $\boldsymbol{f}\mathbf{X}$ is 18. A given PM individual, say {a, 1, x} with respect to Figure 6.7 is an invalid individual. This is because C_1 is not the parent of "a" and likewise C_2 is not the parent of "1". Therefore {a, 1, x} is a non-feasible PM individual although it may be a member of the PM solution space, sequence is of importance.

Let $\rho(\alpha)$ be a parent function for a given CoMI α, and hence $\rho(\alpha)$ reads parent of α, PM schedules are therefore arranged such that $\rho(\alpha_{i,j}) = C_i$. The number of scheduled PM individuals within a given feasible PM region is therefore by induction given by equation 6.3.

$$\#(\boldsymbol{f}\mathbf{X}) = \prod_{i=1}^{m} \alpha_{imax} \tag{6.3}$$

Where: α_{imax} is the maximum CoMI value for component *i* and is evaluated using equation 6.1.

m is the number of components identified for PM.

Equation 6.3 is sufficient for evaluating the number of PM individuals within a given feasible PM region under *primary constraints* and only *expert judgement* under secondary constraints. The evaluation of the number of scheduled PM individuals within the feasible region under *component substitution* is a complex procedure requiring further work. Therefore equation 6.3 is also used in this work to provide a base value for the number of PM individuals within the feasible PM region under component substitution.

To evaluate the number of PM individuals within the solution space **X**, equation 6.3 is also used. However, α_{imax} for all components is calculated using the system risk time as shown in equation 6.4. The solution space **X** for a system model should be bounded by the system under consideration.

$$\alpha_{imax} = Q\left(\frac{RT}{T}\right) \tag{6.4}$$

Where Q is the integer quotient of the division.

6.7 Chapter Summary

To optimise a system, its model must be represented (or encoded) in a form that can be used by search algorithms such as genetic algorithm. An integer encoding of CoMIs for system components has been established. The mechanism by which a population of PM individuals is created has also been addressed where constraints have been established. In addition to guiding the search towards the region of potential feasible PM schedules, the constraints further improve system design from an engineering perspective. Also, in order to infuse diversity into the population, the process of recombination through crossover and mutation has also been addressed with respect to the established constraints.

The PM scheduling optimisation problem has been defined with unavailability and cost being the objective functions. Following this definition, a variant of NSGA II has been established to accommodate system optimisation with the assumptions of PM capability. A calculation for the number of potential PM individuals within PM solution space and feasible region has also been addressed. The calculation is sufficient for PM scheduling optimisations under primary and expert judgement constraints. It is only used as base value evaluation for the component substitution constraint. The absolute approximation value for component substitution is left for further work.

Hence, this chapter concludes the presentation of the approach for optimising PM schedules that represents the key intellectual contribution of this work. Evaluation of this approach follows in chapter 7.

7 EVALUATION

The aim of this chapter is to evaluate the achievement of this work against the research hypothesis and objectives set out in the introduction. To do this, the established models for component reliability, availability and cost in conditions of maintenance (chapter 4) are first demonstrated on a case study performed on a simplified model of the fuel oil service system of a container ship. The calculation of the same attributes at system level and the evaluation of the overall PM optimisation approach presented in chapter 6 are performed on the fuel system and on a second case study applied on a model of an aircraft wheel brake system. The second study shows repeatability of the approach and because of its larger size illustrates to some extent the scalability of the proposed concepts.

Additionally, a set of manually enumerated PM schedules that are totally according to expert judgement is compared with those obtained automatically via application of the optimisation algorithms presented in the thesis. These schedules were derived in consultation with experts in the University of Hull and Germanischer Lloyd and express ad hoc schedules that are expected to yield good performance in terms of unavailability and cost. Both processes (manual and automated) are applied on the fuel oil service system and subjected to primary constraints under PPM policy. The aim is to investigate the benefit of the automated process developed in this work.

7.1 Fuel Oil Service System (FOSS)

The *fuel oil service system* supplies fuel oil to the main engine of a ship.

7.1.1 FOSS Description

The fuel oil service system is illustrated in Figure 7.1. It consists of a *Service tank* in which the fuel oil is stored after it has been purified from water and debris. Within the *Service tank* exists a heating coil to heat up the fuel oil and to also maintain viscosity, thereby making the fuel oil easy to be pumped. The *Service tank* is connected to a *Booster pump* which pumps or transports the fuel oil to a *Mixing tank* through an

Automatic filter and a *Flow meter*. The filter further purifies the fuel oil while the *Flow meter* ensures that there is flow of fuel oil to the *Mixing tank*.

The *Mixing tank* is also connected to the *Service tank* and this is useful in scenario where the pressure in the *Mixing tank* exceeds a defined value. If this is the case, then the excess fuel oil is released to the *Service tank*. From the *Mixing tank*, fuel oil is transported to the *Main engine* through a *Circulation pump*. This pump is further connected to a *Heater* which makes viscosity of the fuel oil easy. The *Heater* in turn is connected to a *Viscosimeter* which regulates the heat of the working heater.

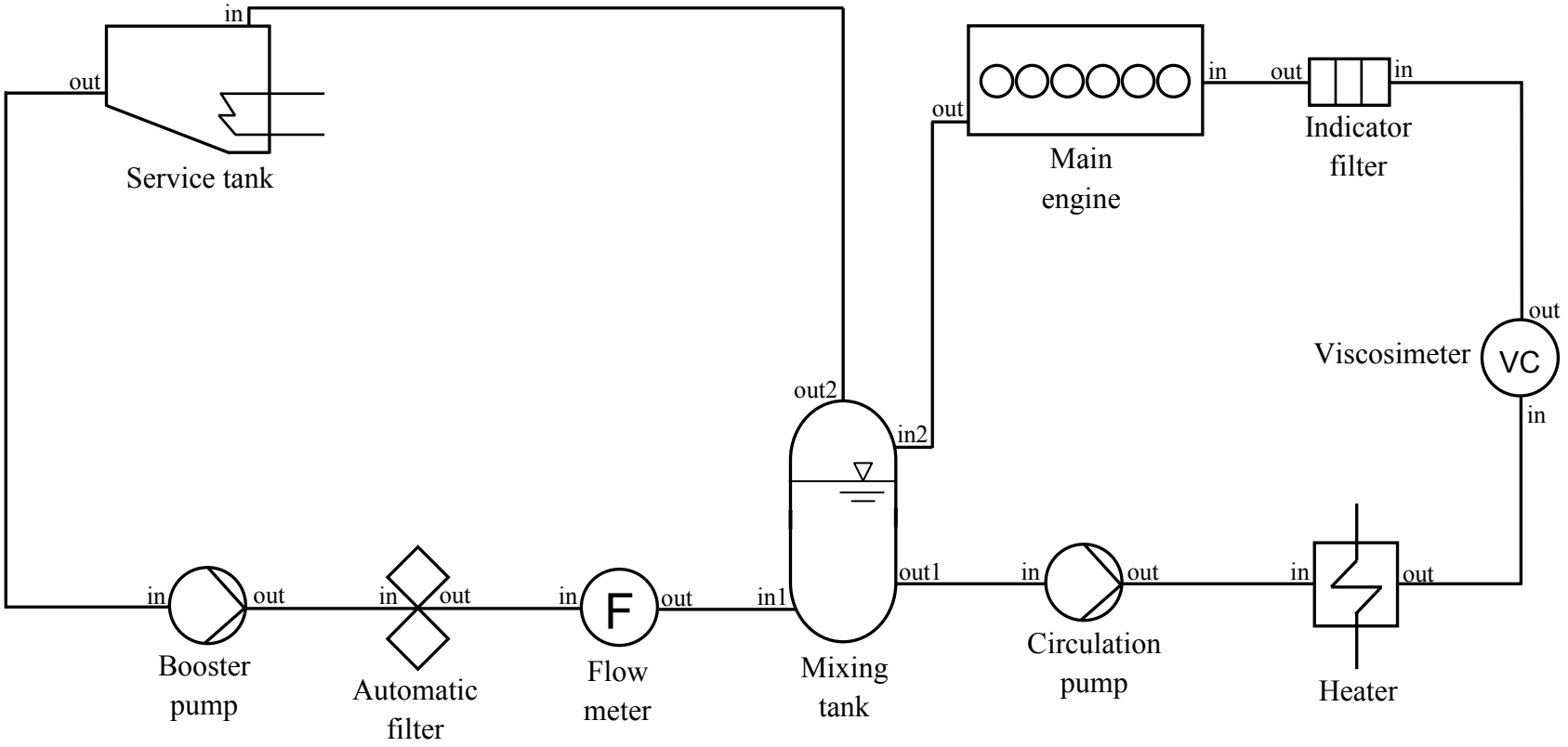

Figure 7.1 - Fuel oil service system (FOSS)

An *Indicator filter* which takes its supply from the *Viscosimeter* further again purifies the fuel oil before reaching the *Main engine*. Excess amount of fuel oil in the *Main engine* is flown back to the *Mixing tank* through the connecting pipe between them.

In order to analyse the model of the fuel oil service system, its constituent components were annotated with their respective failure data in accordance with HiP-HOPS as described in chapter 2 (section 2.1.2.1 on page 15). The failure expression for each component is as shown in Table 7.1. These failure expressions model the failure behaviour of the respective components and are basically specified as lists of internal failure modes of components and lists of deviations of parameters as they can be observed at component outputs with corresponding failure logic connecting these output failures to internal failures and malfunctions of inputs.

Table 7.1 - Components failure expression for the fuel oil service system

Component	**Failure Expression**
Main engine	`O-in or mainEngineFailure`
Indicator filter	`O-in or filter Failure`
Viscosimeter	`O-in or viscosimeterFailure`
Heater	`O-in or heaterFailure`
Circulation pump	`O-in or circulationPumpFailure`
Mixing tank	`O-in1 or O-in2 or mixtankFailure`
Flow meter	`O-in or flowmeterFailure`
Automatic filter	`O-in or automaticFilterFailure`
Booster pump	`O-in or boosterPumpFailure`
Service tank	`O-in or serviceTankFailure`

Table 7.2 identifies the evaluation functions used in evaluating the reliability models while Table 7.3 shows those used for optimising the PM schedules for the FOSS under PPM and IPM. A detailed form for each has already been discussed in chapters 4 and 6 respectively.

Table 7.2 - Summary of evaluation functions used for reliability

	Reliability Functions	
PM Policy	*Each Component of FOSS is Evaluated Using*	*FOSS Evaluated Using*
PPM	equation 4.15	equation 4.46
IPM	equation 4.31	equation 4.46
No PM	equation 4.9	equation 4.46

Table 7.3 - Summary of evaluation functions used for optimising PM schedules

	Objective Functions			
	Unavailability		**Cost**	
PM Policy	*Each Component of FOSS is Evaluated Using*	*FOSS Evaluated Using*	*Each Component of FOSS is Evaluated Using*	*FOSS Evaluated Using*
PPM	equation 4.16	equation 4.46	equation 4.17	equation 4.18
IPM	equation 4.42	equation 4.46	equation 4.44	equation 4.45

In addition to HiP-HOPS annotations with failure data on the constituent components of the FOSS, the following parameter values for both PPM and IPM were used.

Weibull shape parameter $\beta = 2$
Weibull scale parameter $\theta = 1500$
FOSS shortest PM interval $T = 180$
Improvement factor $f = 0.875$

Parameters were given typical values seen in the literature. These values are arbitrary but this is not an issue since the particular shape of the Weibull distribution neither determines the validity nor the approach developed here.

7.2 Component Evaluation Models

The first step in the evaluation was to explore the effect of maintenance on the reliability of a component according to the theoretical models developed in chapter 4.

7.2.1 FOSS Single-Component Evaluation

The *Service tank* component in Figure 7.1 was selected and evaluated for reliability. The effect of PPM and IPM on the reliability of the *Service tank* is in each case first compared to that of a non-PM policy and then all three compared together. In order to have a precise interpretation for the effect of IPM on the component, it is useful to consider two different improvement factors $f_1 = 0.875$ and $f_2 = 0.375$. It is impossible to present in a tabular form all the reliability values from time $t = 0$ to $t = 1500$, therefore a time step of 60 units is considered, which implies time sequence as 0, 60, 120, … , 1380, 1440, 1500.

7.2.1.1 PPM Evaluation

Table 7.4 gives the reliability of the *Service tank* under PPM policy and compares this to corresponding reliability under No PM policy. In this way it is possible to tell if the effect of PPM on the reliability of the *Service tank* is *improved*, as *bad as old*, or even *worse than old*.

Table 7.4 assumes that a component returns to a new age only after its PM actions are completed. This is very obvious especially at and before the first PM stage; i.e. the reliability of the component under No PM and PPM policies are same for $t \leq 180$. It is also evident that for $t > 180$, the reliability of the component is improved under PPM policy compared to the scenario where no PM policy exists.

These two observations are as expected and reveal that the reliability of the *Service tank* improves under PPM policy. It also infers that, unlike the exponential model (appendix A), the Weibull reliability model under PPM is sufficient to modelling a real life PM problem using the age reduction model.

Table 7.4 - *Service tank* reliability under No PM and under PPM

Time (t)	Reliability	
	Under No PM	Under PPM
0	1	1
60	0.998401	0.998401
120	0.99362	0.99362
180	0.985703	0.985703
240	0.974725	0.984127
300	0.960789	0.979415
360	0.944027	0.971611
420	0.924595	0.970057
480	0.902668	0.965412
540	0.878447	0.95772
600	0.852144	0.956189
660	0.823987	0.95161
720	0.794216	0.944027
780	0.763074	0.942518
840	0.730811	0.938005
900	0.697676	0.930531
960	0.663916	0.929043
1020	0.62977	0.924595
1080	0.595473	0.917227
1140	0.561244	0.915761
1200	0.527292	0.911376
1260	0.493812	0.904114
1320	0.46098	0.902668
1380	0.428956	0.898346
1440	0.397882	0.891188
1500	0.367879	0.889763

7.2.1.2 IPM Evaluation

The evaluation of the reliability $R_{ic}(t)$ of the *Service tank* under IPM policy was performed in similar way. The evaluation involves investigating (i) the reliability of the *Service tank* under no PM policy, and (ii) the reliability of the *Service tank* under IPM policy, taking into account two different improvement factor values. Under IPM, a comparison involving one improvement factor value could result into premature

conclusions. Two improvements factor within the domain of the variable [0..1] were therefore tested and the results presented in Table 7.5.

Table 7.5 - *Service tank* reliability under No PM and under IPM

	Reliability		
		Under IPM	
Time (t)	Under No PM	$f = f_1 = 0.875$	$f = f_2 = 0.375$
0	1	1	1
60	0.998401	0.998401	0.998401
120	0.99362	0.99362	0.99362
180	0.985703	0.985703	0.985703
240	0.974725	0.982948	0.978282
300	0.960789	0.977069	0.967833
360	0.944027	0.968123	0.954455
420	0.924595	0.964262	0.941849
480	0.902668	0.95735	0.926512
540	0.878447	0.947455	0.908586
600	0.852144	0.942551	0.891822
660	0.823987	0.934682	0.872729
720	0.794216	0.923925	0.851482
780	0.763074	0.918052	0.831813
840	0.730811	0.909312	0.810279
900	0.697676	0.897789	0.787068
960	0.663916	0.89103	0.765798
1020	0.62977	0.881513	0.743138
1080	0.595473	0.869328	0.719278
1140	0.561244	0.861773	0.697601
1200	0.527292	0.851577	0.674983
1260	0.493812	0.838837	0.651603
1320	0.46098	0.830582	0.630501
1380	0.428956	0.819811	0.60885
1440	0.397882	0.806625	0.586812
1500	0.367879	0.79777	0.567001

As in the case of PPM, the reliability of the *Service tank* under No PM and IPM (for both improvement factor values) policies are also same for $t \leq 180$. Similarly, for both improvements factor under IPM, the reliability of the *Service tank* is seen to improve for

$t > 180$. Interestingly, the reliability of the *Service tank* under IPM improves more with the higher improvement factor f_1 than with f_2 as expected.

It is expected that as $f \rightarrow 0$ the reliability of a component decreases, whereas as $f \rightarrow 1$ the reliability improves. Similarly, the reliability of a component under IPM with improvement factor $f > 0$ is an improvement to the scenario where no PM policy exists. The results indicate that the established reliability model under IPM provides a good frame upon which to model the general effects of IPM on component reliability.

7.2.1.3 Composite Evaluation of PPM and IPM Models

The results of the application of both the models developed in chapter 4 (PPM and IPM) for the *Service tank* are illustrated together in Figure 7.2, which is effectively a graphic representation of Table 7.4 and Table 7.5. The green plot or R_1 is the reliability of the *Service tank* under PPM, the red plot or R_2 is the reliability under IPM with improvement factor $f = f_1 = 0.875$, the deep-red (dark-red) plot or R_3 is the reliability under IPM with improvement factor $f = f_2 = 0.375$, while the black plot or R_4 is the reliability under no PM policy.

From Figure 7.2, it is obvious that reliability is best improved under PPM. The figure also shows that under IPM, reliability is better improved as the improvement factor f value approaches 1. Over all, the figure reveals that reliability is improved under some kind of PM policy which is what one would expect in theory.

These results show that the two models developed in chapter 4 are complimentary (actually the PPM model is a case of IPM with $f = 1$), and together can represent the spectrum from no maintenance to perfect maintenance. Thus, the choice of the Weibull distribution together with the age reduction model that have underpinned the reliability models developed in chapter 4 are deemed appropriate, as the derived models seem capable of logically representing the effects of various types of maintenance at component level.

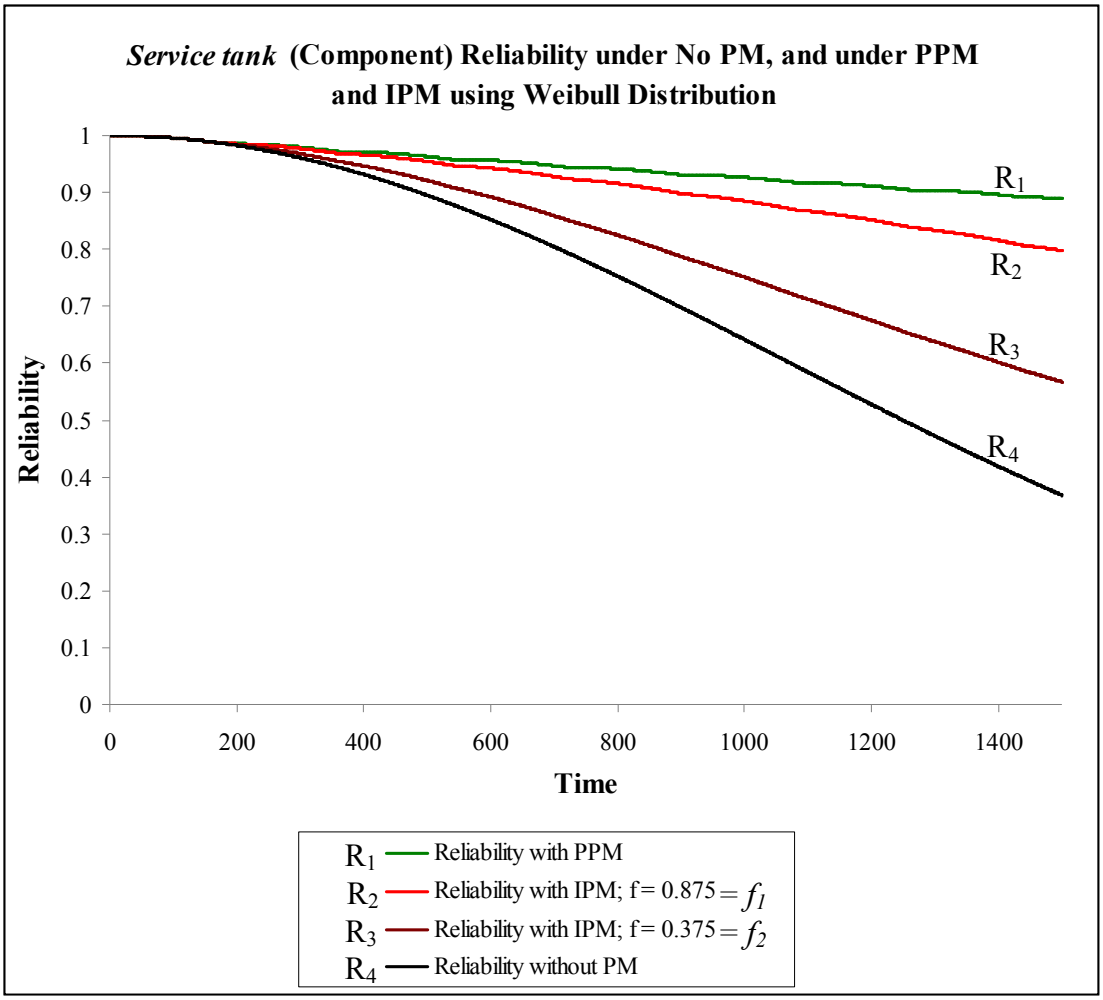

Figure 7.2 - *Service tank* (component) reliability under No PM, and under PPM and IPM

7.2.2 FOSS Reliability Evaluation

The evaluation of the effect of PM policies on the FOSS was performed by HiP-HOPS and the results are presented here in a composite form, i.e. in a single presentation which encompasses No PM and both PM policies (PPM and IPM). The same improvement factor values were set for all components of the FOSS under IPM evaluation; $f_1 = 0.875$ and $f_2 = 0.375$. For the purpose of this evaluation, the same PM time T_p was used for all the components of the FOSS and this value was set to 180 (i.e. it was set to the shortest PM interval for the FOSS, symbolising that CoMIs for all components equal unity).

The results obtained using a time step of 60 units is shown in Table 7.6 while the graphic representation for the entire time scale considered is shown in Figure 7.3.

Table 7.6 - FOSS reliability under No PM, and under PPM and IPM

	Reliability			
			Under IPM	
Time	**Under No PM**	**Under PPM**	$f = f_1 = 0.875$	$f = f_2 = 0.375$
0	1	1	1	1
60	0.985703	0.985703	0.985703	0.985703
120	0.944027	0.944027	0.944027	0.944027
180	0.878447	0.878447	0.878447	0.878447
240	0.794216	0.865888	0.856591	0.820686
300	0.697676	0.829278	0.811572	0.745081
360	0.595473	0.771669	0.747095	0.657351
420	0.493812	0.760636	0.720701	0.583217
480	0.397882	0.728476	0.67552	0.503104
540	0.311486	0.67787	0.615213	0.421982
600	0.236928	0.668178	0.58714	0.356863
660	0.1751	0.639928	0.544473	0.293708
720	0.125732	0.595473	0.490602	0.235276
780	0.0877205	0.586959	0.463238	0.190647
840	0.0594631	0.562142	0.425028	0.15056
900	0.0391639	0.523091	0.37894	0.115908
960	0.0250621	0.515612	0.354024	0.0905786
1020	0.0155826	0.493812	0.321409	0.069123
1080	0.0094136	0.459508	0.283565	0.0515313
1140	0.0055254	0.452938	0.262141	0.0391258
1200	0.00315111	0.433788	0.235513	0.0290828
1260	0.00174605	0.403653	0.205635	0.0211763
1320	0.000940029	0.397882	0.188123	0.015746
1380	0.000491721	0.381059	0.167271	0.0114973
1440	0.000249912	0.354588	0.144559	0.00825071
1500	0.00012341	0.349518	0.130887	0.00605695

As expected, the reliability of the FOSS under No PM, and under PPM and IPM policies are the same for $t \leq 180$. At $t > 180$ reliability is seen to best improve under PPM, followed by IPM with improvement factor f_1 and finally f_2. This relationship can further be seen in Figure 7.3.

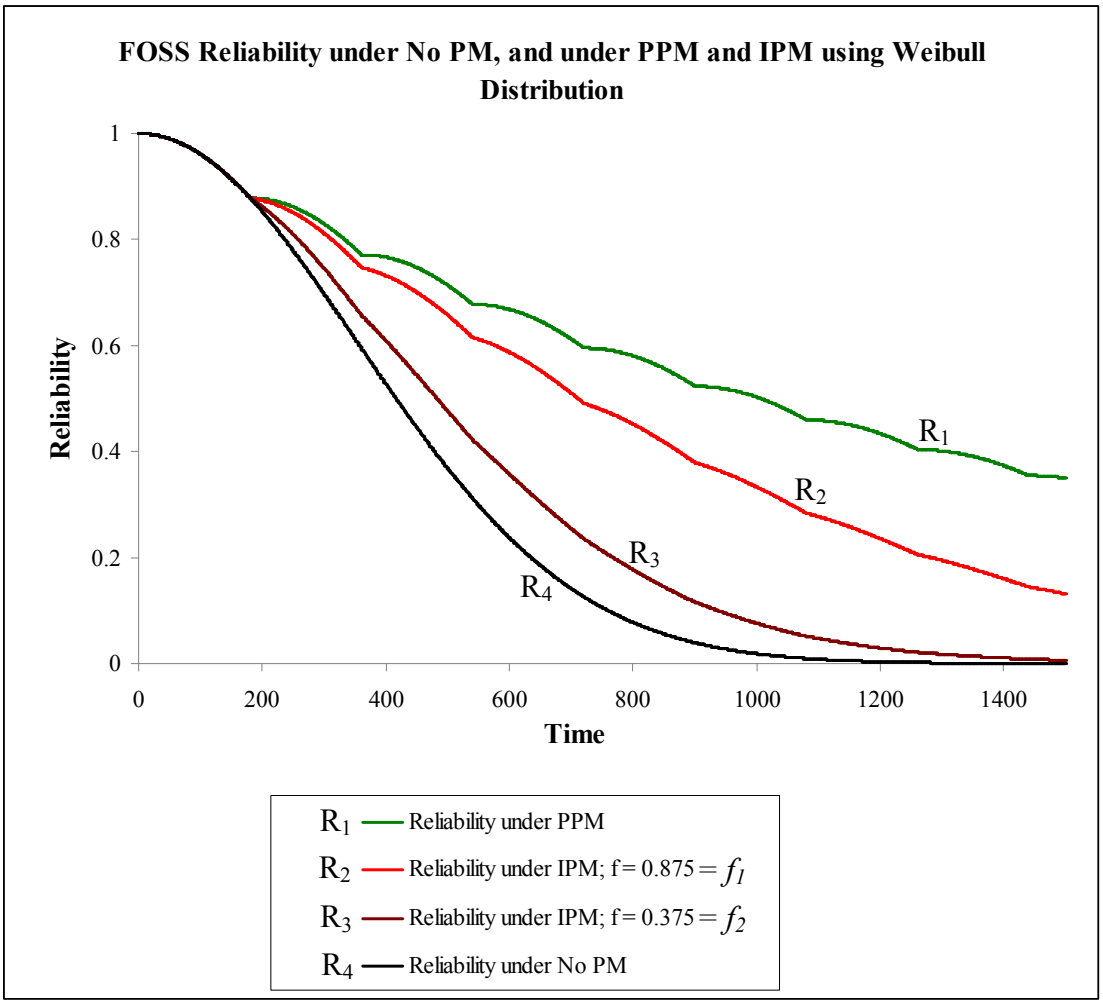

Figure 7.3 - FOSS reliability under No PM, and under PPM and IPM

7.3 Evaluation of PM Scheduling Optimisation on FOSS

The optimisation approach developed in chapter 6 was evaluated on the FOSS under PPM and IPM policies. PPM evaluation is thus first discussed followed by IPM.

7.3.1 Evaluation of PPM Scheduling Optimisation on FOSS

The result of a PPM evaluation on the FOSS is a set of optimal PPM schedules. Each evaluation performed in this work is based on identified constraints and hence it is performed in the following order.

(i) Applying only the primary constraints on the optimisation

(ii) Applying only the secondary constraints on the optimisation where (a) expert judgement is first evaluated separately and (b) architecture modification of the FOSS through component substitution is also evaluated separately.

(iii) Applying all the identified constraints on respective selected components in a single run.

The results are discussed next.

7.3.1.1 Through Primary Constraints

The evaluation of the PPM scheduling optimisation was performed by using constraints C1 (ensuring that PM is not carried out too late) and C2 (ensuring that PM is not carried out too early) as established in chapter 6 (specifically on page 96). PM optimisation under primary constraints is basically an optimisation from fundamentals as also established in chapter 6. The Pareto frontier of results obtained is shown in Figure 7.4. All the PM individuals forming part of the Pareto frontier are simply those that are best trade-offs between FOSS unavailability and cost. None of them can in total be said to be better than the other and their selection may largely depend on system design requirements. The result summary of Figure 7.4 is as shown in Table 7.7.

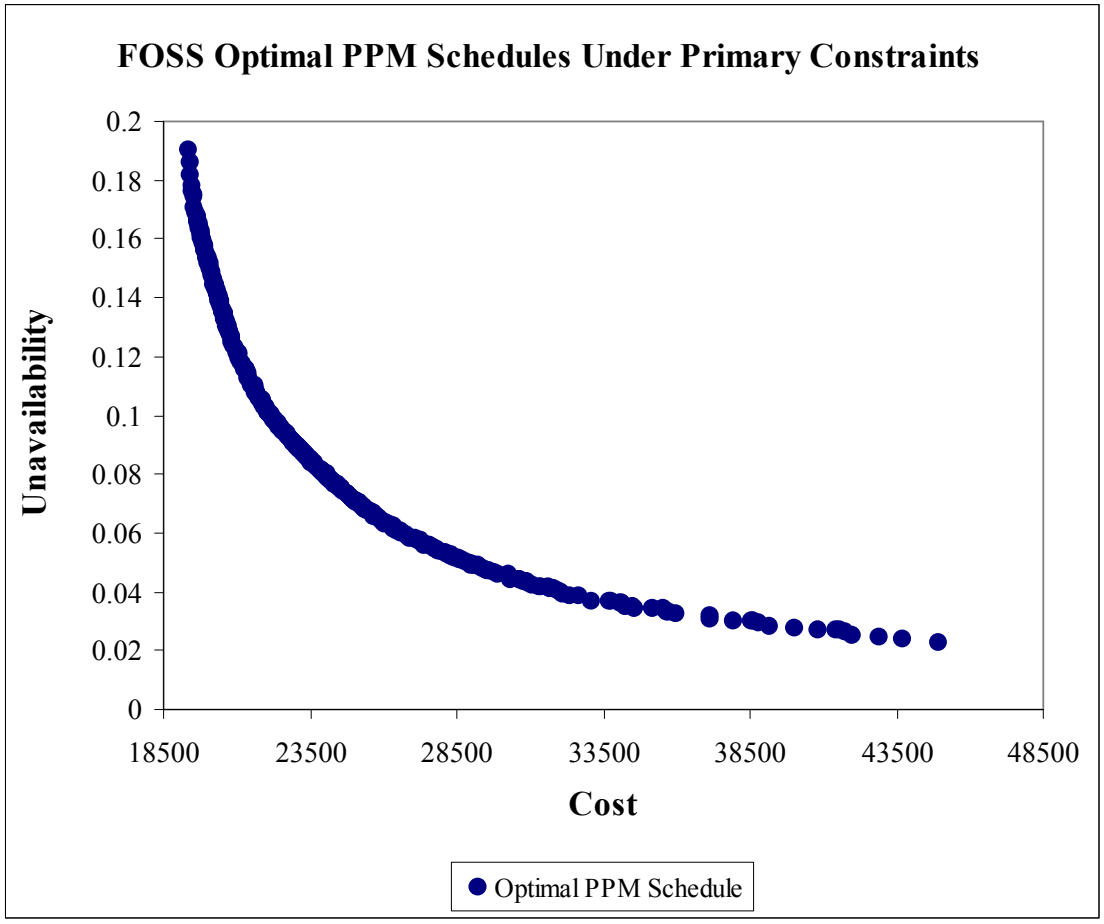

Figure 7.4 - Pareto frontier of PPM schedules under primary constraints

Table 7.7 - Results summary for the Pareto frontier of PPM schedules

Optimisation Indicators	**Value**
PPM Solution Space **X**	15,516,041,187,205,900,000
Feasible PPM Region ***f*X**	2,310,448,250,880
Number of solutions (optimal PPM schedules) found	349
Total number of generations the search was subjected	5120
Generation for which last solution(s) was/were found	3087

The HiP-HOPS tabular output format for PM schedule (PMSS) of a system from where Figure 7.4 and subsequent ones were obtained is as follows.

PMSS = C_1-name($CoMI_1$) C_2-name($CoMI_2$) C_3-name($CoMI_3$) ... C_m-name($CoMI_m$)

Where C_i-*name* is the name of the *i-th* component.

$CoMI_i$ is the CoMI for the *i-th* component.

Due to space limitation, short names are used here to represent the actual component names for the FOSS. These short names are as shown in Table 7.8.

Table 7.8 - Short name representation of actual components name of the FOSS

Actual Component Name	**Short Name**
Automatic filter	af
Booster pump	bp
Circulation pump	cp
Flow meter	fm
Heater	ht
Indicator filter	if
Main engine	me
Mixing tank	mt
Service tank	st
Viscosimeter	vm

It is also impossible to present the entire tabular format of the 349 optimal PPM schedules seen in Figure 7.4, therefore only the first and last 10 of the PPM schedules found by the GA are shown in Table 7.9 and the components short name representation established in Table 7.8 is used. The table shows PPM schedules against their respective evaluated objective functions (*cost* and *unavailability*) and the generations for which they were found. The PM time for any of the components is simple, and is the product of the shortest PM interval (T = 180) and the CoMI of the component in question.

Table 7.9 - A subset of optimal PPM schedules; a tabular representation

Optimal PPM Schedule	Cost	Unavailability	Generation
af(11) bp(5) cp (9) fm(6) ht(3) if(3) me(17) mt(7) st(8) vm(9)	20448	0.137757	50
af(8) bp(4) cp(9) fm(6) ht(3) if(6) me(17) mt(7) st(8) vm(6)	20776	0.12792	51
af(8) bp(4) cp(6) fm(6) ht(2) if(3) me(17) mt(7) st(8) vm(6	21359	0.113982	52
af(6) bp(4) cp(6) fm(6) ht(2) if(3) me(17) mt(7) st(6) vm(6)	21884	0.103924	53
af(11) bp(5) cp(9) fm(6) ht(4) if(3) me(11) mt(7) st(8) vm(6)	20582	0.133858	53
af(11) bp(5) cp(9) fm(9) ht(5) if(7) me(13) mt(7) st(8) vm(9)	19923	0.156719	53
af(6) bp(4) cp(6) fm(3) ht(2) if(3) me(17) mt(5) st(4) vm(3)	23619	0.083703	53
af(6) bp(4) cp(6) fm(5) ht(2) if(3) me(17) mt(5) st(6) vm(6)	22248	0.098694	54
af(4) bp(2) cp(3) fm(4) ht(2) if(2) me(16) mt(5) st(4) vm(3)	25506	0.067579	54
af(11) bp(5) cp(9) fm(6) ht(3) if(6) me(13) mt(7) st(8) vm(9)	20238	0.144436	54
af(11) bp(5) cp(9) fm(6) ht(3) if(3) me(13) mt(7) st(8) vm(6)	20623	0.13236	1441
af(1) bp(1) cp(2) fm(1) ht(1) if(1) me(17) mt(2) st(1) vm(1)	41731	0.026365	2906
af(1) bp(1) cp(2) fm(2) ht(1) if(1) me(17) mt(1) st(1) vm(1)	41470	0.026992	2913
af(11) bp(10) cp(9) fm(9) ht(5) if(7) me(13) mt(7) st(12) vm(9)	19467	0.177714	2953
af(1) bp(1) cp(1) fm(1) ht(1) if(1) me(17) mt(2) st(1) vm(1)	43731	0.024119	2983
af(1) bp(1) cp(1) fm(1) ht(1) if(1) me(17) mt(1) st(1) vm(1)	44937	0.022431	2984
af(1) bp(1) cp(2) fm(1) ht(1) if(1) me(17) mt(1) st(1) vm(1)	42937	0.024681	2992
af(4) bp(2) cp(6) fm(3) ht(2) if(2) me(16) mt(5) st(4) vm(3)	24919	0.071305	2993
af(1) bp(1) cp(2) fm(1) ht(1) if(1) me(17) mt(1) st(1) vm(2)	41537	0.026925	3079
af(8) bp(5) cp(9) fm(6) ht(3) if(3) me(13) mt(7) st(8) vm(6)	20888	0.123454	3087

7.3.1.2 Through Secondary Constraints

The evaluation of the PPM scheduling optimisation under secondary constraints was performed by applying constraints C3 (on page 98, expert judgement on component PM time) and C4 (on page 102, architecture modification through component substitution) to the optimisation. Unlike the primary constraints where C1 and C2 are inseparable, C3 and C4 as seen in chapter 6 are two options that are independent, though their combination in a single optimisation is possible. Hence, to evaluate the PPM scheduling optimisation through secondary constraints, two cases are considered; (i) expert judgement on component PM time, and (ii) architecture modification through component substitution. These are discussed as follows.

Case i - Expert Judgement on Component PM Time

Under the expert judgement, three components were selected and in HiP-HOPS annotated with an expert PM time. The implication on the optimisation is that these components will have fixed CoMIs through out the optimisation. The selected components are shown in Table 7.10.

Table 7.10 - Selected components for expert judgement with their respective expert PM times

Components	**Expert PM Time**
Main engine	1500
Service tank	1260
Viscosimeter	870

The Pareto frontier of results obtained under the expert judgement is shown in Figure 7.5.

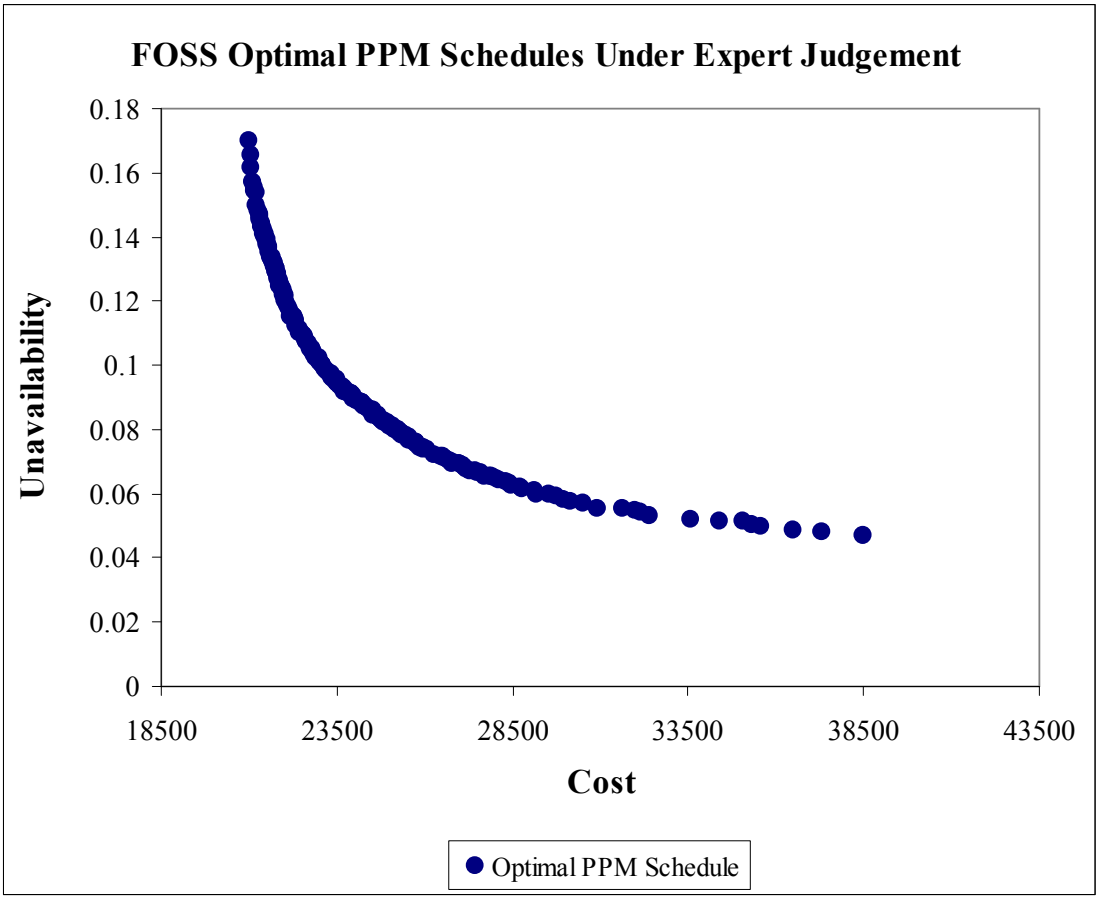

Figure 7.5 - Pareto frontier of PPM schedules under expert judgement

A summary of the Pareto frontier seen in Figure 7.5 is shown in Table 7.11.

Table 7.11 - Results summary for the Pareto frontier of PPM schedules under expert judgement

Optimisation Indicators	**Value**
PPM Solution Space **X**	15,516,041,187,205,900,000
Feasible PPM Region ***f*X**	81,682,513,920
Number of solutions (optimal PPM schedules) found	214
Total number of generations the search was subjected	5120
Generation for which last solution(s) was/were found	810

Table 7.12 - A subset of optimal PPM schedules under expert judgement; a tabular representation

Optimal PPM Schedule	Cost	Unavailability	Generation
af(8) bp(5) cp(6) fm(6) ht(3) if(3) me(8) mt(7) st(7) vm(4)	22358	0.112255	12
af(8) bp(4) cp(9) fm(6) ht(3) if(3) me(8) mt(7) st(7) vm(4)	22206	0.11542	12
af(8) bp(5) cp(6) fm(5) ht(3) if(3) me(8) mt(7) st(7) vm(4)	22521	0.109769	12
af(11) bp(5) cp(9) fm(6) ht(3) if(3) me(8) mt(7) st(7) vm(4)	21843	0.126741	12
af(8) bp(5) cp(9) fm(9) ht(3) if(3) me(8) mt(7) st(7) vm(4)	21945	0.124033	12
af(8) bp(4) cp(6) fm(6) ht(3) if(3) me(8) mt(7) st(7) vm(4)	22456	0.109883	12
af(8) bp(5) cp(9) fm(6) ht(4) if(3) me(8) mt(7) st(7) vm(4)	22067	0.1193	12
af(11) bp(5) cp(9) fm(6) ht(4) if(3) me(8) mt(7) st(7) vm(4)	21802	0.128248	12
af(8) bp(5) cp(9) fm(5) ht(3) if(3) me(8) mt(7) st(7) vm(4)	22271	0.115307	12
af(11) bp(5) cp(9) fm(6) ht(3) if(5) me(8) mt(7) st(7) vm(4)	21703	0.131867	13
af(11) bp(5) cp(9) fm(6) ht(5) if(3) me(8) mt(7) st(7) vm(4)	21761	0.130101	687
af(1) bp(1) cp(2) fm(1) ht(1) if(1) me(8) mt(2) st(7) vm(4)	35301	0.050541	729
af(1) bp(1) cp(1) fm(1) ht(1) if(1) me(8) mt(1) st(7) vm(4)	38507	0.046705	736
af(11) bp(5) cp(9) fm(6) ht(3) if(7) me(8) mt(7) st(7) vm(4)	21563	0.136522	737
af(8) bp(4) cp(9) fm(6) ht(4) if(3) me(8) mt(7) st(7) vm(4)	22165	0.116947	762
af(1) bp(1) cp(2) fm(1) ht(1) if(1) me(8) mt(1) st(7) vm(4)	36507	0.048899	771
af(1) bp(1) cp(1) fm(1) ht(1) if(1) me(8) mt(2) st(7) vm(4)	37301	0.048351	805
af(1) bp(1) cp(2) fm(2) ht(1) if(1) me(8) mt(1) st(7) vm(4)	35040	0.051152	806
af(11) bp(5) cp(9) fm(9) ht(3) if(7) me(8) mt(7) st(7) vm(4)	21400	0.142644	806
af(8) bp(5) cp(9) fm(6) ht(3) if(3) me(8) mt(7) st(7) vm(4)	22108	0.117777	810

It is not surprising that the feasible PPM region under expert judgement is smaller compared to that under primary constraints. This is because the expert PM time imposes a fixed CoMI on the component for which it applies, and this is achieved through the use of equation 6.2. Under primary constraints, a component has a chance of being allocated a CoMI between *1* and α_{max} inclusive for every variant of the FOSS PM schedule. However, under expert judgement, this value does not vary and remains fixed for every variant of the FOSS and hence the smaller size of the feasible PPM region.

Table 7.12 shows a subset of the optimal PPM schedules under expert judgement in a tabular form. It comprises of the first and last 10 of the 214 optimal PPM schedules of Figure 7.5. The table shows that the components subjected to expert judgement have fixed CoMIs throughout the variant FOSS PPM schedules. These components are *Main engine*, *Service tank* and *Viscosimeter*. The fixed CoMIs as a result of the expert PM time are *Main engine (8)*, *Service tank (7)* and *Viscosimeter (4)*. This characteristic is as expected.

Case ii - Architecture Modification through Component Substitution

This second evaluation of the secondary constraints enables the PPM schedule optimisation to substitute implementations of components flagged for substitution. Every implementation of a flagged component has equal chance of being selected for substitution and hence creates a much larger feasible PPM region. The flagged components of the FOSS for substitution alongside their implementations are as shown in Table 7.13. The table also contain short names for the respective component implementations. The short names are similar to their counterparts in Table 7.8, with the exception that those in Table 7.13 are postfixed with a concatenation of underscore character and an index number.

Table 7.13 shows that 3 components were flagged for substitutions; these are *Heater*, *Mixing tank* and *Flow meter*. *Heater* and *Mixing tank* have 4 implementations each while *Flow meter* has 5. The result of the PPM schedule optimisation is as shown in Figure 7.6 and the summaries in Table 7.14 and Table 7.15.

Table 7.13 - Components with their implementations and short name representation

Component	**Implementations**	**Short Name**
Heater	Heater_1	ht_1
	Heater_2	ht_2
	Heater_3	ht_3
	Heater_4	ht_4
Mixing tank	Mixing_tank_1	mt_1
	Mixing_tank_2	mt_2
	Mixing_tank_3	mt_3
	Mixing_tank_4	mt_4
Flow meter	Flow_meter_1	fm_1
	Flow_meter_2	fm_2
	Flow_meter_3	fm_3
	Flow_meter_4	fm_4
	Flow_meter_5	fm_5

Table 7.15 shows the implementations that form part of the PPM schedule solution vector. It can be seen in Table 7.14, that the PPM schedule solution vector consists of 321 optimal PPM schedules. Table 7.15 also shows the implementations and their number of occurrences found to be part of the total 321 optimal PPM schedules found. Also, the value of the PM solution space **X** and PM feasible region ***fX*** as seen in Table 7.14 are in accordance with base value evaluation as established in chapter 6 (section 6.6.2), and hence the indication of greater than (>).

Some component implementations of the FOSS were found fitter than others, while some were found to be completely weak with respect to the set FOSS design objectives. For instance, for component *Heater*, implementation *heater_2* was found most suitable for the design objectives and occurred in all the 321 optimal PPM schedules found. For component *Mixing tank*, two of its implementations dominate the PPM solution vector, *mixing_tank_3* forms part of 192 optimal PPM schedules while *Mixing_tank_4* forms part of the remaining 129 optimal PPM schedules. Component *Flow meter* has

implementation *flow_meter_3* dominating the entire 321 PPM optimal schedules. Thus it is likely that architecture modification could result into solution vectors with diverse implementations of same component. This however will depend on the failure data of the respective implementations.

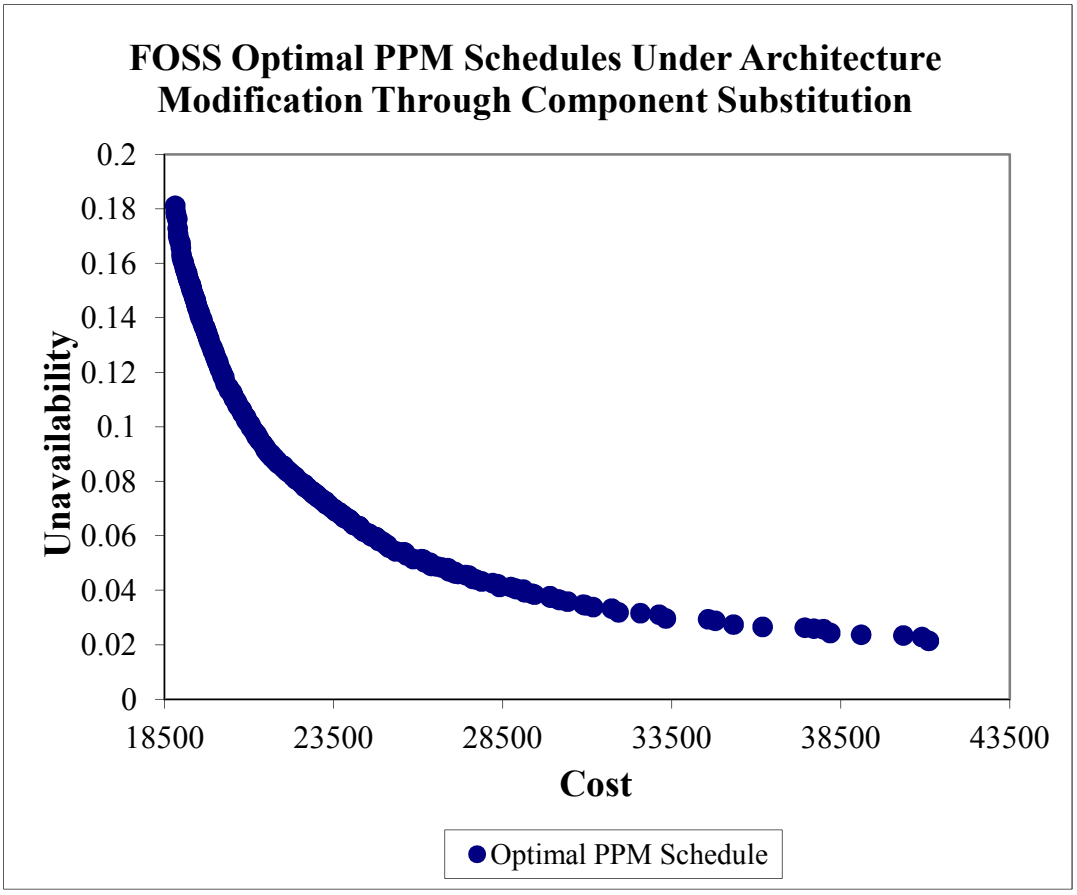

Figure 7.6 - Pareto frontier of PPM schedules under component substitution

Table 7.14 - Results summary for the Pareto frontier of PPM schedules under component substitution

Optimisation Indicators	**Value**
PPM Solution Space **X**	> 15,516,041,187,205,900,000
Feasible PPM Region ***f*X**	> 2,310,448,250,880
Number of solutions (optimal PPM schedules) found	321
Total number of generations the search was subjected	5120
Generation for which last solution(s) was/were found	2388

Table 7.15 - Number of occurrences of component implementations forming part of the FOSS PPM optimisation solution vector under component substitution

Component	**Implementations**	**Number of Occurrence**
Heater	Heater_1	0
	Heater_2	321
	Heater_3	0
	Heater_4	0
Mixing tank	Mixing_tank_1	0
	Mixing_tank_2	0
	Mixing_tank_3	192
	Mixing_tank_4	129
Flow meter	Flow_meter_1	0
	Flow_meter_2	0
	Flow_meter_3	321
	Flow_meter_4	0
	Flow_meter_5	0

The tabular representation of a subset of Figure 7.6 is as shown in Table 7.16. It shows the first and last 10 of the 321 optimal PPM schedules found. The representation of the PM schedule of the *i-th* component ($PMSC_i$) under component substitution in HiP-HOPS is as follows.

$PMSC_i = C_i\text{-}name.C_{i,k}\text{-}name(CoMI_{i,k})$

Where $C_i\text{-}name$ is the name of the *i-th* component

$C_{i,k}\text{-}name$ is the name of the *k-th* (active) implementation of the *i-th* component

$CoMI_{i,k}$ is the CoMI of the *k-th* (active) implementation of the *i-th* component

The component and the active implementation names are separated by a dot (.). Within the PM schedule of the system (i.e. FOSS), the PM schedule representation for the *i-th* component not flagged for substitution remains $PMSC_i = C_i\text{-}name(CoMI_i)$.

Table 7.16 - A subset of optimal PPM schedules under component substitution; a tabular representation

Optimal PPM Schedule	Cost	Unavailability	Generation
af(8) bp(4) cp(6) fm.fm_3(4) ht.ht_2(1) if(3) me(11) mt.mt_4(7) st(6) vm(6)	21011	0.10112	25
af(6) bp(4) cp(6) fm.fm_3(4) ht.ht_2(2) if(3) me(11) mt.mt_3(4) st(6) vm(6)	21456	0.092533	25
af(6) bp(4) cp(6) fm.fm_3(4) ht.ht_2(2) if(3) me(16) mt.mt_3(3) st(6) vm(6)	21496	0.091414	25
af(6) bp(3) cp(6) fm.fm_3(4) ht.ht_2(1) if(3) me(11) mt.mt_3(2) st(6) vm(3)	22387	0.080732	26
af(6) bp(4) cp(6) fm.fm_3(4) ht.ht_2(1) if(3) me(11) mt.mt_3(2) st(6) vm(3)	22191	0.083251	27
af(8) bp(5) cp(6) fm.fm_3(4) ht.ht_2(2) if(3) me(16) mt.mt_3(3) st(6) vm(6)	21133	0.098672	27
af(6) bp(3) cp(6) fm.fm_3(4) ht.ht_2(1) if(3) me(11) mt.mt_3(4) st(6) vm(6)	21742	0.088699	27
af(8) bp(4) cp(6) fm.fm_3(4) ht.ht_2(1) if(3) me(11) mt.mt_3(3) st(6) vm(6)	21321	0.094933	27
af(8) bp(4) cp(6) fm.fm_3(4) ht.ht_2(1) if(3) me(11) mt.mt_3(2) st(6) vm(6)	21401	0.093817	28
af(6) bp(4) cp(6) fm.fm_3(4) ht.ht_2(1) if(3) me(11) mt.mt_3(4) st(6) vm(6)	21546	0.091196	28
af(6) bp(3) cp(6) fm.fm_3(4) ht.ht_2(2) if(3) me(11) mt.mt_3(4) st(6) vm(6)	21652	0.090039	1460
af(1) bp(1) cp(1) fm.fm_3(1) ht.ht_2(1) if(1) me(17) mt.mt_3(1) st(1) vm(1)	41104	0.021441	2236
af(6) bp(4) cp(6) fm.fm_3(4) ht.ht_2(1) if(3) me(11) mt.mt_3(2) st(6) vm(6)	21666	0.088954	2238
af(6) bp(3) cp(6) fm.fm_3(4) ht.ht_2(2) if(3) me(11) mt.mt_3(2) st(6) vm(6)	21772	0.087795	2245
af(1) bp(1) cp(1) fm.fm_3(1) ht.ht_2(1) if(1) me(17) mt.mt_3(2) st(1) vm(1)	40904	0.022849	2279
af(1) bp(1) cp(1) fm.fm_3(2) ht.ht_2(1) if(1) me(17) mt.mt_3(1) st(1) vm(1)	40352	0.023412	2283
af(8) bp(4) cp(6) fm.fm_3(4) ht.ht_2(2) if(3) me(14) mt.mt_3(3) st(6) vm(6)	21231	0.096264	2300
af(1) bp(1) cp(2) fm.fm_3(1) ht.ht_2(1) if(1) me(17) mt.mt_3(1) st(1) vm(2)	37704	0.02594	2347
af(6) bp(3) cp(6) fm.fm_3(4) ht.ht_2(1) if(3) me(11) mt.mt_3(2) st(6) vm(6)	21862	0.086451	2349
af(1) bp(1) cp(2) fm.fm_3(1) ht.ht_2(1) if(1) me(17) mt.mt_3(1) st(1) vm(1)	39104	0.023693	2388

At the early stages (i.e. early generations) of PPM optimisation, the solution search has a much higher possibility of finding as part of the optimal PPM schedules different implementations of any given component that is flagged for substitution. This means that at early generations, the optimal PPM schedules will consist of diverse implementations for any given component. This is because at such generations, only fewer PPM schedules have been searched within the feasible PPM region, and therefore the optimal PPM schedules may only have dominated fewer potential solutions. As the search progresses through generations, previously found optimal PPM schedules may as well be dominated by newly found ones. Hence, under component substitution the Pareto frontier converges to PPM schedules consisting of component implementation(s) that is or are best suitable for the given design objectives, while weaker implementation(s) is or are eliminated. To view this transformation, Tables B.1 – B.3 in appendix B show the optimal PPM schedules under component substitution that were obtained in generations 1 – 3 respectively.

7.3.1.3 FOSS PPM Optimisation from Fundamentals versus Component Substitution

The architecture modification through component substitution allows for exploration of various available implementations of a given component with respect to the design objectives. It is therefore worth comparing Figure 7.6 with Figure 7.4 (PPM optimisation under primary constraints otherwise referred to as PPM optimisation from fundamentals). A Pareto frontier for an optimisation problem with two objective functions infers that an optimal solution with the highest value in one of the objective functions also has the lowest value in the other objective function value, and vice versa. Hence the comparison reveals the following.

Under primary constraints, the two optimal PPM schedules at the two ends of the Pareto frontier has the values (i) unavailability = 0.189988, cost = 19356; found in generation 676, and (ii) unavailability = 0.0224311, cost = 44937; found in generation 2984. These give differences in objective space as unavailability = 0.1675569 and cost = 25581.

Under component substitution the values are (i) unavailability = 0.181133, cost = 18819; found in generation 551, and (ii) unavailability = 0.0214408, cost = 41104; found in generation 2236. The differences in objective space are unavailability = 0.1596922 and cost = 22285.

It is then clear that under primary constraints, the FOSS has greater differences in objective function space values compared to when subjected to component substitution. This explains the existence of more optimal PPM schedules found under primary constraints (349) than component substitution (321). The wider the difference in objective function space, the more likely it is for NSGA II to crowd the space with optimal solutions through genetic operators. Although more optimal PPM solutions were found under primary constraints, the component substitution was able to explore the wider feasible PPM region to evolve with an over-all optimal PPM schedules with the least unavailability and the least cost values. For instance the two optimal PPM schedules at both ends of the Pareto frontier under component substitution; the one found in generation 551 constituting implementations *Flow_meter_3*, *Heater_2* and *Mixing_tank_4*, and that found in generation 2236 constituting implementations *Flow_meter_3*, *Heater_2* and *Mixing_tank_3*.

7.3.1.4 Composite Evaluation

The composite PPM evaluation on the FOSS is based on the composite definition of the PM optimisation problem which encompasses both the primary and secondary constraints. As in section 6.3.1.2, same components were here subjected to expert judgement with same expert PM times as seen in Table 7.10. Also same components were subjected to component substitution with same respective implementations as seen in Table 7.13.

The Pareto frontier of the composite PPM optimisation is shown in Figure 7.7 and the summary of the optimisation in Table 7.17. Table 7.17 reveals that a total of 206 optimal PPM schedules were found, with the last found in generation 1722. In a real life

engineering optimisation problem, a given system model may be subjected to both primary and secondary constraints in a single instance of PPM optimisation run.

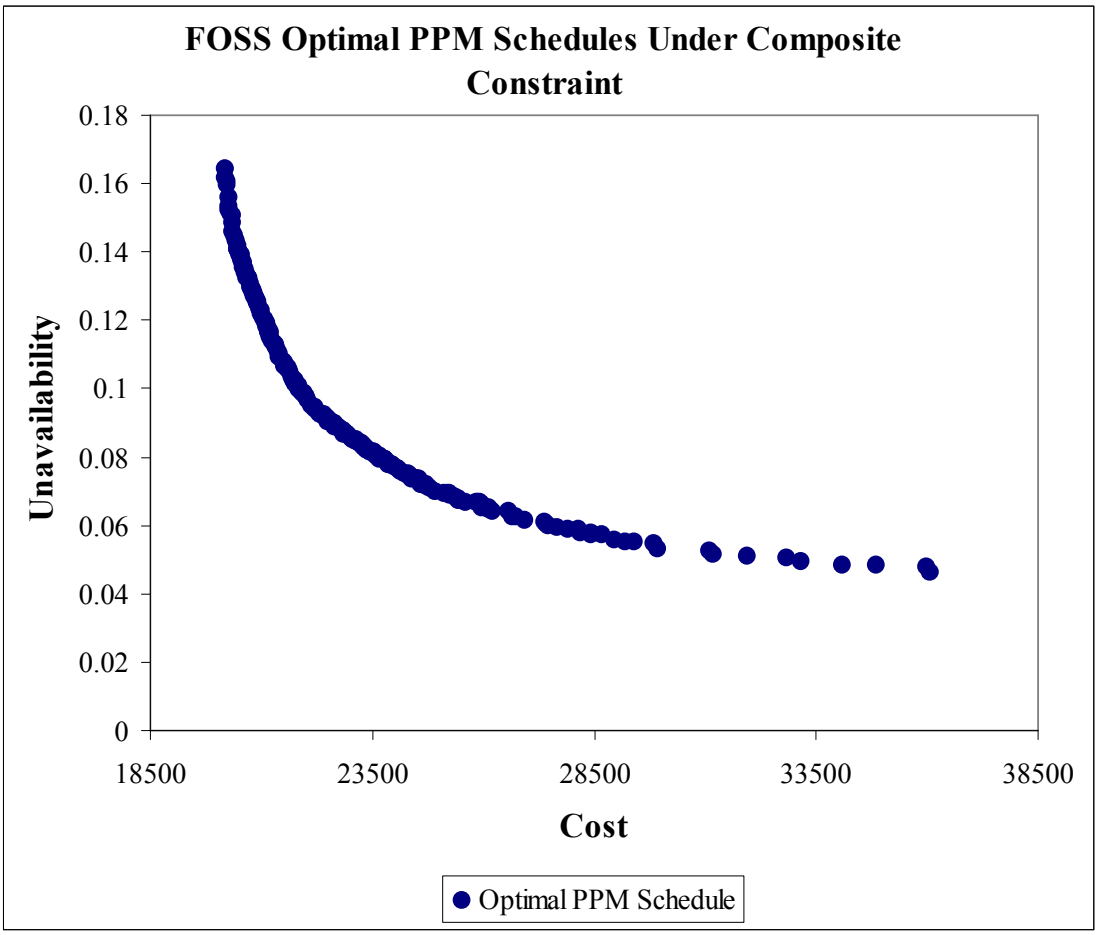

Figure 7.7 - Pareto frontier of PPM schedules under composite constraint

Table 7.17 - Results summary for the Pareto frontier of PPM schedules under composite constraint

Optimisation Indicators	**Value**
PPM Solution Space **X**	> 15,516,041,187,205,900,000
Feasible PPM Region ***f*X**	> 2,310,448,250,880
Number of solutions (optimal PPM schedules) found	206
Total number of generations the search was subjected	5120
Generation for which last solution(s) was/were found	1722

Table 7.18 shows the number of occurrences of component implementations forming part of the FOSS composite PPM optimisation solution vector. The component implementations summary in Table 7.18 is similar to that of Table 7.15 (number of occurrences of component implementations forming part of the FOSS PPM optimisation solution vector under component substitution). The difference is that in the composite PPM optimisation, the expert judgement has an effect on the objective function evaluations, and as seen in Table 7.18, component *Mixing tank*, has implementation *Mixing_tank_2* dominating the entire optimal PPM schedules, unlike in Table 7.15 where it is shared between *Mixing_tank_3* and *Mixing_tank_4*.

Table 7.18 - Number of occurrences of component implementations forming part of the FOSS composite PPM optimisation solution vector

Component	**Implementations**	**Number of Occurrence**
Heater	Heater_1	0
	Heater_2	206
	Heater_3	0
	Heater_4	0
Mixing tank	Mixing_tank_1	0
	Mixing_tank_2	206
	Mixing_tank_3	0
	Mixing_tank_4	0
Flow meter	Flow_meter_1	0
	Flow_meter_2	0
	Flow_meter_3	206
	Flow_meter_4	0
	Flow_meter_5	0

Table 7.19 shows the first and last 10 out of the 206 composite PPM schedules. Similar to the evaluation of the FOSS under expert judgement, the table shows that the components *Main engine*, *Service tank* and *Viscosimeter* subjected to expert judgement has fixed CoMIs in all the optimal PPM schedules.

The first three generations of the composite PPM scheduling optimisation are shown in Tables C.1 – C.3 in appendix C. The tables show the diverse component implementations existing in early generations.

Table 7.19 - A subset of optimal PPM schedules under composite constraint; a tabular representation

Optimal PPM Schedule	Cost	Unavailability	Generation
af(8) bp(4) cp(6) ft.ft_3(4) ht.ht_2(1) if(3) me(8) mt.mt_2(5) st(7) vm(4)	21866	0.099808	26
af(8) bp(4) cp(6) ft.ft_3(4) ht.ht_2(3) if(3) me(8) mt.mt_2(5) st(7) vm(4)	21746	0.102454	26
af(6) bp(4) cp(6) ft.ft_3(4) ht.ht_2(2) if(3) me(8) mt.mt_2(4) st(7) vm(4)	22217	0.094253	29
af(5) bp(3) cp(6) ft.ft_3(4) ht.ht_2(2) if(3) me(8) mt.mt_2(4) st(7) vm(4)	22678	0.089516	29
af(6) bp(3) cp(6) ft.ft_3(4) ht.ht_2(2) if(3) me(8) mt.mt_2(5) st(7) vm(4)	22237	0.093825	30
af(5) bp(3) cp(6) ft.ft_3(4) ht.ht_2(1) if(3) me(8) mt.mt_2(4) st(7) vm(4)	22768	0.088175	30
af(6) bp(4) cp(6) ft.ft_3(4) ht.ht_2(3) if(3) me(8) mt.mt_2(5) st(7) vm(4)	22011	0.097638	31
af(6) bp(3) cp(6) ft.ft_3(4) ht.ht_2(1) if(3) me(8) mt.mt_2(5) st(7) vm(4)	22327	0.092491	31
af(8) bp(7) cp(9) ft.ft_3(4) ht.ht_2(2) if(6) me(8) mt.mt_2(8) st(7) vm(4)	20944	0.124657	31
af(6) bp(3) cp(6) ft.ft_3(2) ht.ht_2(1) if(3) me(8) mt.mt_2(5) st(7) vm(4)	22703	0.088824	31
af(6) bp(4) cp(6) ft.ft_3(4) ht.ht_2(1) if(2) me(8) mt.mt_2(4) st(7) vm(4)	22587	0.090419	1093
af(2) bp(1) cp(1) ft.ft_3(1) ht.ht_2(1) if(1) me(8) mt.mt_2(2) st(7) vm(4)	31944	0.051043	1356
af(8) bp(4) cp(6) ft.ft_3(4) ht.ht_2(2) if(3) me(8) mt.mt_2(5) st(7) vm(4)	21776	0.101132	1365
af(1) bp(1) cp(1) ft.ft_3(1) ht.ht_2(1) if(1) me(8) mt.mt_2(1) st(7) vm(4)	36091	0.046305	1399
af(2) bp(1) cp(1) ft.ft_3(1) ht.ht_2(1) if(1) me(8) mt.mt_2(1) st(7) vm(4)	33176	0.049128	1401
af(6) bp(4) cp(6) ft.ft_3(4) ht.ht_2(1) if(3) me(8) mt.mt_2(4) st(7) vm(4)	22307	0.092919	1439
af(1) bp(1) cp(1) ft.ft_3(1) ht.ht_2(1) if(1) me(8) mt.mt_2(2) st(7) vm(4)	34859	0.048226	1453
af(8) bp(4) cp(6) ft.ft_3(4) ht.ht_2(2) if(3) me(8) mt.mt_2(8) st(7) vm(4)	21600	0.105751	1463
af(1) bp(1) cp(1) ft.ft_3(1) ht.ht_2(2) if(1) me(8) mt.mt_2(1) st(7) vm(4)	36001	0.047708	1631
af(8) bp(3) cp(6) ft.ft_3(4) ht.ht_2(2) if(3) me(8) mt.mt_2(5) st(7) vm(4)	21972	0.098662	1722

7.3.2 Evaluation of IPM Scheduling Optimisation on FOSS

The IPM evaluation on the FOSS produces a set of optimal IPM schedules. The evaluation was carried out in same manner as the PPM evaluation, i.e. by first applying only primary constraints on the optimisation, followed by secondary constraints where *expert judgement* and *component substitution* were considered separately and finally by performing a composite evaluation on the FOSS.

7.3.2.1 Through Primary Constraints

Similar to PPM, the IPM scheduling optimisation on the FOSS was performed using constraints C1 and C2 as established in chapter 6 (specifically on page 96). The Pareto frontier of results obtained from the IPM scheduling optimisation is shown in Figure 7.8.

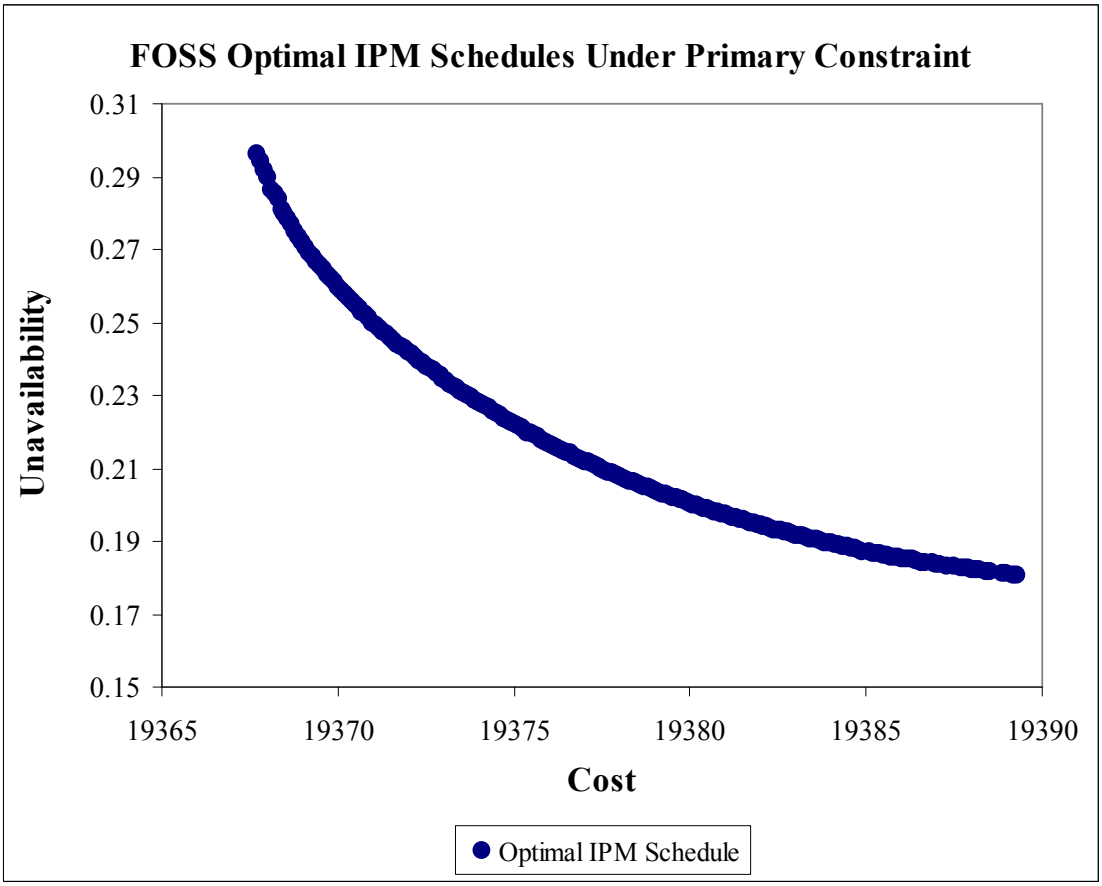

Figure 7.8 - Pareto frontier of IPM schedules under primary constraints

The summary of Figure 7.8 is as seen in Table 7.20 and shows that 206 optimal IPM schedules were found in the search that progressed through 5120 generations. However, the last found optimal IPM schedule was in generation 2160. A tabular representation of the first and last 10 of the 206 optimal IPM schedules is shown in Table 7.21.

Table 7.20 - Results summary for the Pareto frontier of IPM schedules under primary constraints

Optimisation Indicators	**Value**
IPM Solution Space **X**	15,516,041,187,205,900,000
Feasible IPM Region ***f*X**	2,568,674,820,096
Number of solutions (optimal IPM schedules) found	206
Total number of generations the search was subjected	5120
Generation for which last solution(s) was/were found	2160

It may be useful to contrast the PPM and IPM schedules obtained under primary constraints in order to have a better perception of what went on in optimisation the IPM schedules of the FOSS. The contrast is largely based on how the Pareto frontier is populated. To do this, the distance in objective function space for both the PPM and IPM needs to be established.

The two optimal PPM schedules at the two ends of the PPM Pareto frontier (Figure 7.4) has the values (i) unavailability = 0.189988, cost = 19356; found in generation 676, and (ii) unavailability = 0.0224311, cost = 44937; found in generation 2984. These give differences in objective function space as unavailability = 0.1675569 and cost = 25581.

Under IPM, these values are (i) unavailability = 0.296192, cost = 19367.7; found in generation 761, and (ii) unavailability = 0.18051, cost = 19389.3; found in generation 680. The differences in objective function space are unavailability = 0.115682 and cost = 21.6.

The FOSS PM optimisation has greater differences in objective function space values under PPM. This explains why there are more optimal PM schedules under PPM (349)

compared to that obtained under IPM (206). As mentioned earlier, the wider the difference in objective function space, the more likely it is for NSGA II to crowd the space with optimal solutions through genetic operators.

However, the difference in object function space for cost under IPM is marginal. For a closer analysis, Table 7.22, Table 7.23 and Table 7.24 shows the optimal IPM schedules obtained in generations 1, 2 and 3 respectively. The total number of optimal IPM schedules found in generations 1, 2 and 3 are 3, 4 and 21 respectively.

The objective functions value of the two optimal IPM schedules at both ends of the Pareto frontier in generation 1 (Table 7.22) are (i) unavailability = 0.249952, cost = 20248.1, and (ii) unavailability = 0.235774, cost = 30300.8. The differences in objective function space are unavailability = 0.014178 and cost = 10052.7. It can be seen that at generation 1 the distance in cost space is wider than it is in Figure 7.8 (the final optimal IPM schedules obtained for the FOSS).

In generation 2 (Table 7.23), the objective function values of the two optimal IPM schedules at both ends of the Pareto frontier are (i) unavailability = 0.242588, cost = 19828.6; found in generation 2, and (ii) unavailability = 0.215759, cost = 20030.4; found in generation 2. The differences in objective function space are unavailability = 0.026829 and cost = 201.8. At this generation, the distance in cost space seems to have drastically shrunk. In optimisation terms, this implies that better IPM schedules with less cost values were found which replaced those found in generation 1.

In generation 3 (Table 7.24), the two optimal IPM schedules at both ends of the Pareto frontier has objective functions value as (i) unavailability = 0.249186, cost = 19817.3; found in generation 3, and (ii) unavailability = 0.212386, cost = 20055.9; found in generation 3. The differences in objective function space are unavailability = 0.0368 and cost = 238.6.

In general, as the IPM optimisation progresses over generations, IPM schedules of better unavailability and more cost effective are found, which eventually dominate several others found previously.

Table 7.21 - A subset of optimal IPM schedules under primary constraints; a tabular representation

Optimal IPM Schedule	Cost	Unavailability	Generation
af(12) bp(12) cp(10) fm(14) ht(14) if(16) me(10) mt(12) st(18) vm(16)	19375.5	0.219477	28
af(12) bp(12) cp(10) fm(13) ht(14) if(16) me(10) mt(12) st(19) vm(16)	19375.3	0.220431	28
af(13) bp(12) cp(11) fm(14) ht(14) if(16) me(10) mt(12) st(19) vm(16)	19376.6	0.21398	29
af(12) bp(12) cp(11) fm(13) ht(14) if(16) me(10) mt(12) st(18) vm(16)	19375.4	0.219768	30
af(13) bp(14) cp(12) fm(15) ht(14) if(18) me(10) mt(12) st(22) vm(16)	19378.8	0.204644	33
af(13) bp(13) cp(12) fm(14) ht(14) if(18) me(10) mt(12) st(20) vm(16)	19377.6	0.209385	34
af(12) bp(12) cp(11) fm(14) ht(14) if(16) me(10) mt(12) st(19) vm(16)	19376.1	0.216327	35
af(12) bp(12) cp(11) fm(14) ht(13) if(16) me(10) mt(12) st(19) vm(16)	19376	0.216792	36
af(13) bp(13) cp(12) fm(15) ht(14) if(18) me(10) mt(12) st(21) vm(16)	19378.3	0.206428	43
af(12) bp(12) cp(11) fm(14) ht(14) if(17) me(10) mt(12) st(20) vm(16)	19376.4	0.214741	44
af(11) bp(12) cp(10) fm(13) ht(12) if(14) me(10) mt(12) st(17) vm(15)	19373.7	0.230092	1135
af(12) bp(12) cp(11) fm(13) ht(13) if(16) me(10) mt(12) st(20) vm(16)	19375.8	0.217873	1158
af(11) bp(12) cp(10) fm(13) ht(12) if(15) me(10) mt(12) st(18) vm(15)	19374	0.22813	1183
af(11) bp(12) cp(10) fm(12) ht(12) if(14) me(10) mt(12) st(17) vm(15)	19373.3	0.23261	1194
af(12) bp(12) cp(11) fm(13) ht(13) if(16) me(10) mt(12) st(19) vm(16)	19375.6	0.218991	1382
af(11) bp(10) cp(9) fm(10) ht(9) if(11) me(10) mt(9) st(14) vm(12)	19369.2	0.269171	1567
af(11) bp(10) cp(9) fm(11) ht(9) if(12) me(10) mt(10) st(15) vm(13)	19370.3	0.25688	1718
af(13) bp(13) cp(12) fm(15) ht(14) if(17) me(10) mt(12) st(19) vm(16)	19377.7	0.209003	1727
af(11) bp(10) cp(9) fm(10) ht(8) if(11) me(10) mt(8) st(13) vm(11)	19368.5	0.27966	1924
af(13) bp(12) cp(12) fm(15) ht(14) if(17) me(10) mt(12) st(21) vm(16)	19378	0.207834	2160

Table 7.22 - Optimal IPM schedules under primary constraints in generation 1

Optimal IPM Schedule	Cost	Unavailability	Generation
af(16) bp(16) cp(6) fm(17) ht(6) if(5) me(18) mt(11) st(16) vm(16)	20579.9	0.248151	1
af(21) bp(13) cp(14) fm(10) ht(8) if(4) me(12) mt(11) st(17) vm(13)	20248.1	0.249952	1
af(15) bp(13) cp(12) fm(18) ht(4) if(14) me(1) mt(10) st(15) vm(13)	30300.8	0.235774	1

Table 7.23 - Optimal IPM schedules under primary constraints in generation 2

Optimal IPM Schedule	Cost	Unavailability	Generation
af(15) bp(13) cp(12) fm(18) ht(6) if(14) me(12) mt(11) st(15) vm(13)	19969.2	0.226341	2
af(21) bp(13) cp(12) fm(18) ht(8) if(14) me(12) mt(11) st(15) vm(13)	20030.4	0.215759	2
af(21) bp(13) cp(14) fm(10) ht(8) if(11) me(12) mt(11) st(17) vm(13)	19966.4	0.230068	2
af(15) bp(13) cp(12) fm(10) ht(8) if(14) me(12) mt(10) st(15) vm(13)	19828.6	0.242588	2

Table 7.24 - Optimal IPM schedules under primary constraints in generation 3

Optimal IPM Schedule	Cost	Unavailability	Generation
af(21) bp(13) cp(12) fm(18) ht(8) if(14) me(12) mt(11) st(15) vm(13)	20030.4	0.215759	2
af(15) bp(13) cp(12) fm(10) ht(8) if(14) me(12) mt(10) st(15) vm(13)	19828.6	0.242588	2
af(15) bp(13) cp(12) fm(10) ht(8) if(14) me(12) mt(10) st(17) vm(13)	19838.5	0.239359	3
af(15) bp(13) cp(12) fm(18) ht(8) if(14) me(12) mt(10) st(15) vm(13)	19918.1	0.225194	3
af(15) bp(13) cp(14) fm(18) ht(6) if(14) me(12) mt(11) st(17) vm(13)	20000.1	0.220557	3
af(15) bp(13) cp(14) fm(10) ht(8) if(14) me(12) mt(11) st(17) vm(13)	19866.1	0.234485	3
af(15) bp(13) cp(12) fm(10) ht(8) if(14) me(12) mt(9) st(15) vm(13)	19822.7	0.245533	3
af(15) bp(13) cp(12) fm(18) ht(8) if(18) me(12) mt(11) st(15) vm(13)	19933.9	0.220725	3
af(15) bp(13) cp(12) fm(18) ht(8) if(14) me(12) mt(11) st(15) vm(13)	19924.7	0.222715	3
af(21) bp(13) cp(12) fm(18) ht(8) if(14) me(12) mt(10) st(15) vm(13)	20023.9	0.218259	3
af(15) bp(13) cp(12) fm(10) ht(8) if(11) me(12) mt(10) st(15) vm(13)	19823.2	0.244976	3
af(15) bp(13) cp(12) fm(17) ht(8) if(14) me(12) mt(10) st(15) vm(13)	19904.1	0.226489	3
af(15) bp(13) cp(12) fm(10) ht(8) if(14) me(12) mt(8) st(15) vm(13)	19817.3	0.249186	3
af(15) bp(13) cp(12) fm(10) ht(8) if(14) me(12) mt(11) st(15) vm(13)	19835.2	0.240166	3
af(15) bp(13) cp(12) fm(10) ht(8) if(14) me(12) mt(11) st(17) vm(13)	19845.1	0.236926	3
af(15) bp(13) cp(12) fm(10) ht(8) if(15) me(12) mt(10) st(15) vm(13)	19830.7	0.242005	3
af(15) bp(13) cp(12) fm(13) ht(8) if(14) me(12) mt(11) st(17) vm(13)	19872.7	0.227896	3
af(21) bp(13) cp(14) fm(18) ht(8) if(11) me(12) mt(11) st(17) vm(13)	20055.9	0.212386	3
af(15) bp(13) cp(12) fm(10) ht(8) if(11) me(12) mt(10) st(17) vm(13)	19833.1	0.241757	3
af(21) bp(13) cp(12) fm(18) ht(8) if(11) me(12) mt(10) st(17) vm(13)	20028.4	0.217401	3
af(21) bp(13) cp(14) fm(16) ht(8) if(14) me(12) mt(11) st(17) vm(13)	20034.1	0.2127	3

7.3.2.2 Through Secondary Constraints

The evaluation of the IPM scheduling optimisation on the FOSS under secondary constraints was carried out in similar manner as its PPM equivalent. Two cases were also considered; (i) expert judgement on component PM time, and (ii) architecture modification through component substitution.

Case i - Expert Judgement on Component PM Time

Under the expert judgement, same components selected under PPM policy were also selected and in HiP-HOPS annotated with same expert PM times. These components are *Main engine* with *1500* time units, *Service tank* with *1260* time units and *Viscosimeter* with *870* time units. The notion behind the use of same data in this work is to maintain consistency and ensure fair comparison where necessary. The Pareto frontier of the IPM schedules found is shown in Figure 7.9 and the summary in Table 7.25.

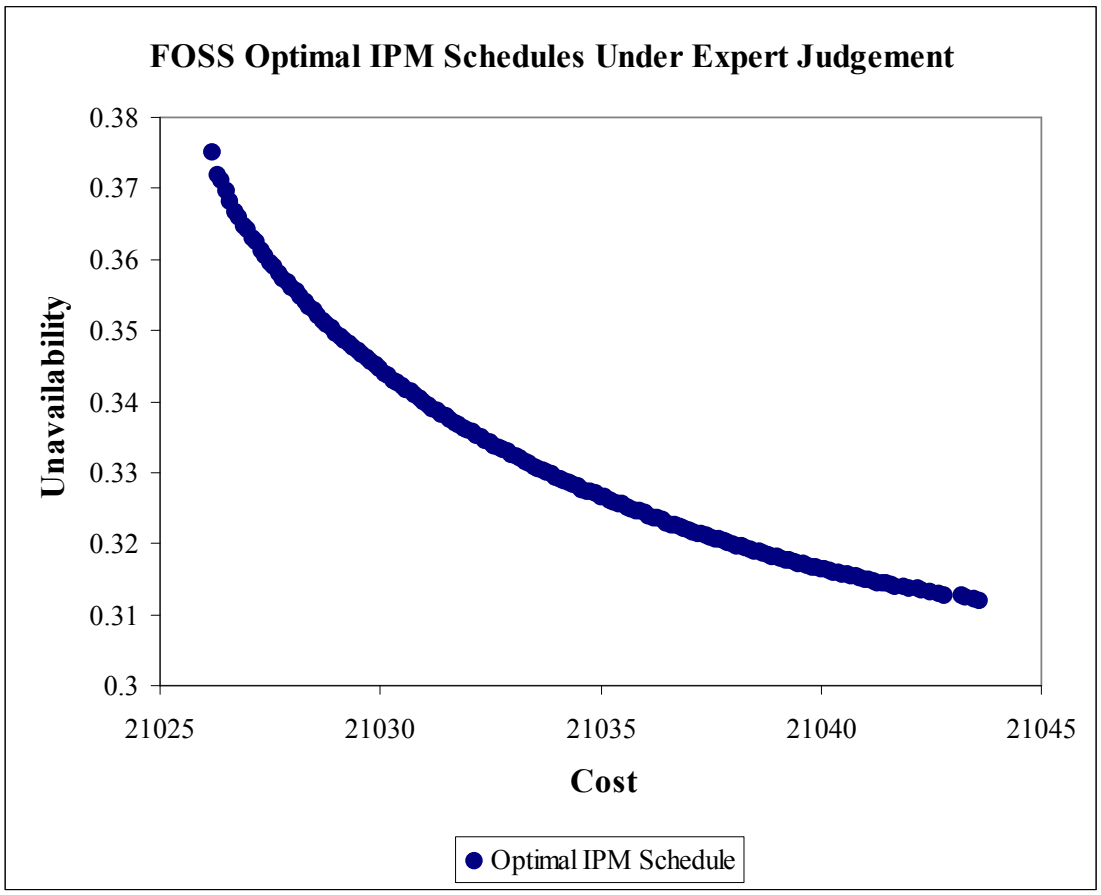

Figure 7.9 - Pareto frontier of IPM schedules under expert judgement

Table 7.25 - Results summary for the Pareto frontier of IPM schedules under expert judgement

Optimisation Indicators	**Value**
IPM Solution Space **X**	15,516,041,187,205,900,000
Feasible IPM Region ***f*X**	90,811,736,064
Number of solutions (optimal IPM schedules) found	167
Total number of generations the search was subjected	5120
Generation for which last solution(s) was/were found	830

Table 7.25 shows that a total of 167 optimal IPM schedules were found through a search that progressed through to 5120 generations, where the last optimal IPM schedule was found in generation 830. In contrast, 214 PM schedules were found under PPM policy as seen in Table 7.11.

Table 7.26 shows a subset of the optimal IPM schedules under expert judgement in a tabular form. The subset consists of the first and last 10 of the 167 optimal IPM schedules seen in Figure 7.9. The table also shows that the components subjected to expert judgement have fixed CoMIs throughout the variant FOSS IPM schedules. These fixed CoMIs are not in any way different from those obtained under PPM policy seen in Table 7.12. The fixed CoMIs as a result of the expert PM time are *Main engine (8)*, *Service tank (7)* and *Viscosimeter (4)*.

Table 7.26 - A subset of optimal IPM schedules under expert judgement; a tabular representation

Optimal PPM Schedule	Cost	Unavailability	Generation
af(16) bp(16) cp(14) fm(18) ht(14) if(21) me(8) mt(12) st(7) vm(4)	21038.1	0.319637	5
af(11) bp(11) cp(10) fm(13) ht(12) if(15) me(8) mt(12) st(7) vm(4)	21029.4	0.34745	7
af(11) bp(12) cp(10) fm(13) ht(12) if(15) me(8) mt(12) st(7) vm(4)	21029.6	0.346486	11
af(14) bp(13) cp(12) fm(15) ht(14) if(18) me(8) mt(12) st(7) vm(4)	21033.4	0.331193	15
af(14) bp(14) cp(13) fm(15) ht(14) if(20) me(8) mt(12) st(7) vm(4)	21034.2	0.32873	17
af(14) bp(13) cp(12) fm(14) ht(14) if(18) me(8) mt(12) st(7) vm(4)	21032.9	0.332823	17
af(14) bp(14) cp(13) fm(16) ht(14) if(20) me(8) mt(12) st(7) vm(4)	21034.7	0.327295	17
af(13) bp(13) cp(11) fm(14) ht(14) if(18) me(8) mt(12) st(7) vm(4)	21032	0.335885	18
af(14) bp(14) cp(13) fm(16) ht(14) if(19) me(8) mt(12) st(7) vm(4)	21034.6	0.327578	19
af(14) bp(14) cp(12) fm(15) ht(14) if(20) me(8) mt(12) st(7) vm(4)	21033.8	0.329884	20
af(11) bp(10) cp(10) fm(11) ht(11) if(14) me(8) mt(11) st(7) vm(4)	21028	0.355986	372
af(12) bp(11) cp(10) fm(12) ht(14) if(16) me(8) mt(12) st(7) vm(4)	21029.7	0.346006	388
af(12) bp(11) cp(10) fm(13) ht(14) if(16) me(8) mt(12) st(7) vm(4)	21030.1	0.34386	418
af(11) bp(10) cp(9) fm(11) ht(10) if(13) me(8) mt(10) st(7) vm(4)	21027.3	0.361147	454
af(11) bp(10) cp(9) fm(11) ht(11) if(13) me(8) mt(12) st(7) vm(4)	21027.9	0.356759	464
af(11) bp(10) cp(9) fm(10) ht(10) if(13) me(8) mt(10) st(7) vm(4)	21027	0.364095	468
af(12) bp(12) cp(10) fm(13) ht(14) if(17) me(8) mt(12) st(7) vm(4)	21030.4	0.342502	502
af(12) bp(11) cp(10) fm(13) ht(14) if(17) me(8) mt(12) st(7) vm(4)	21030.2	0.343472	502
af(11) bp(10) cp(9) fm(10) ht(10) if(13) me(8) mt(8) st(7) vm(4)	21026.5	0.369634	676
af(11) bp(11) cp(9) fm(10) ht(11) if(13) me(8) mt(10) st(7) vm(4)	21027.2	0.362338	830

Case ii - Architecture Modification through Component Substitution

This evaluation enables the IPM scheduling optimisation to substitute implementations of components flagged for substitution. The components of the FOSS flagged for substitution under IPM are same as those considered for PPM as shown in Table 7.13. These are *Heater*, *Mixing tank* and *Flow meter* with *Heater* and *Mixing tank* having 4 implementations each while *Flow meter* having 5 implementations. The result of the IPM scheduling optimisation is as shown in Figure 7.10 and the summaries in Table 7.27 and Table 7.28.

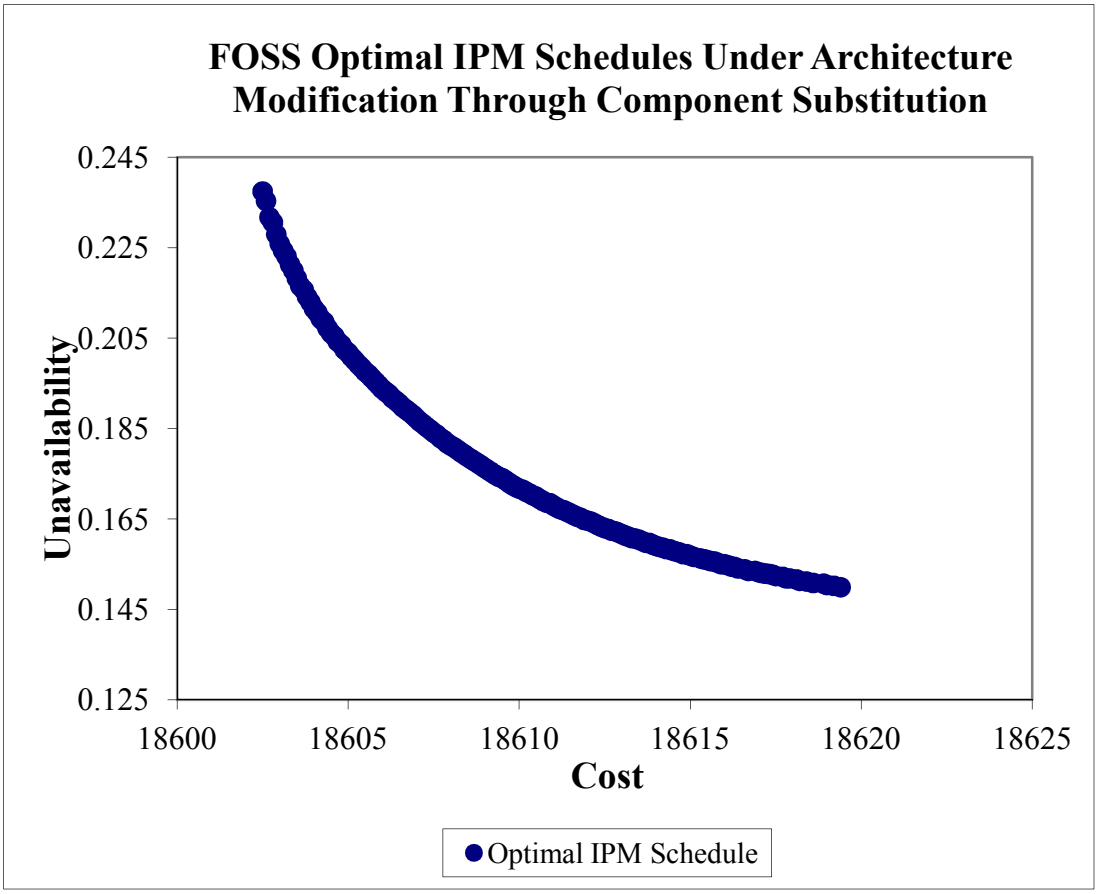

Figure 7.10 - Pareto frontier of IPM schedules under component substitution

Table 7.27 shows that 159 optimal IPM schedules were found through a search that stretched through 5120 generations. However, the last optimal IPM schedule was found in generation 2446.

Table 7.27 - Results summary for the Pareto frontier of IPM schedules under

Optimisation Indicators	**Value**
IPM Solution Space **X**	> 15,516,041,187,205,900,000
Feasible IPM Region $f\mathbf{X}$	> 2,568,674,820,096
Number of solutions (optimal IPM schedules) found	159
Total number of generations the search was subjected	5120
Generation for which last solution(s) was/were found	2446

Table 7.28 - Number of occurrences of component implementations forming part of the FOSS IPM optimisation solution vector under component substitution

Component	**Implementations**	**Number of Occurrence**
Heater	Heater_1	159
	Heater_2	0
	Heater_3	0
	Heater_4	0
Mixing tank	Mixing_tank_1	0
	Mixing_tank_2	0
	Mixing_tank_3	0
	Mixing_tank_4	159
Flow meter	Flow_meter_1	0
	Flow_meter_2	0
	Flow_meter_3	159
	Flow_meter_4	0
	Flow_meter_5	0

Table 7.28 shows the implementations of the respective components that were found fitter for the IPM policy. For component *Heater*, implementation *heater_1* was found most suitable for the design objectives of the FOSS compared to *heater_2* which was found most suitable under PPM policy. For component *Mixing tank*, only

implementation *Mixing_tank_4* was found most suitable which in contrast to PPM policy, implementations *mixing_tank_3* and *mixing_tank_4* dominated the PM solution vector. For component *Flow meter*, implementation *flow_meter_3* was found most suitable, same as found under PPM policy.

The tabular representation of a subset of Figure 7.10 is shown in Table 7.29 and consists of the first and last 10 optimal IPM schedules of the 159 found. Interestingly in all the 159 optimal IPM schedules, *Main engine* and *Flow_meter_3* has CoMIs 10 and 1 respectively. This implies that those are the most suitable CoMIs for the respective components for the specified PM policy.

Similar to the evaluation of component substitution under PPM, at the early generations of PM optimisation under IPM, the optimal IPM schedule vector will consist of different implementations of any given component that is flagged for substitution. To view this, Tables D.1 – D.3 in appendix D show the optimal IPM schedules under component substitution that were obtained in generations 1 – 3 respectively.

7.3.2.3 FOSS IPM Optimisation from Fundamentals versus Component Substitution

Similar to the comparison made between PPM scheduling optimisation under architecture modification through component substitution and primary constraint, the comparison of these two constraints under IPM scheduling optimisation is here discussed. It is therefore worth comparing Figure 7.10 with Figure 7.8 (IPM scheduling optimisation under primary constraints, also referred to as from fundamentals). This comparison is as follows.

Under primary constraints, the two optimal IPM schedules at the two ends of the Pareto frontier has the values (i) unavailability = 0.296192, cost = 19367.7; found in generation 761, and (ii) unavailability = 0.18051, cost = 19389.3; found in generation 680. These give differences in objective function space as unavailability = 0.115682 and cost = 21.6.

Under component substitution the values are (i) unavailability = 0.23746, cost = 18602.5; found in generation 505, and (ii) unavailability = 0.149943, cost = 18619.4; found in generation 116. The differences in objective function space are unavailability = 0.087517 and cost = 16.9.

As is the case under PPM, the IPM scheduling optimisation under primary constraints has greater differences in objective function space values compared to that subjected to component substitution. It also explains why there are more optimal IPM schedules found under primary constraints (206) than component substitution (159). It is obvious that more optimal IPM schedules were found under primary constraints, however the component substitution was able to explore the wider feasible IPM region and to evolve with an over all optimal IPM schedule with the least unavailability and cost values.

Table 7.29 - A subset of optimal IPM schedules under component substitution; a tabular representation

Optimal PPM Schedule	Cost	Unavailability	Generation
af(16) bp(15) cp(13) ft.ft_3(1) ht.ht_1(14) if(19) me(10) mt.mt_4(1) st(22) vm(16)	18612.9	0.162048	25
af(15) bp(16) cp(13) ft.ft_3(1) ht.ht_1(14) if(21) me(10) mt.mt_4(1) st(22) vm(16)	18612.8	0.162336	26
af(15) bp(14) cp(12) ft.ft_3(1) ht.ht_1(14) if(18) me(10) mt.mt_4(1) st(22) vm(16)	18611.5	0.166281	28
af(15) bp(15) cp(13) ft.ft_3(1) ht.ht_1(14) if(21) me(10) mt.mt_4(1) st(22) vm(16)	18612.5	0.163013	28
af(16) bp(16) cp(15) ft.ft_3(1) ht.ht_1(14) if(21) me(10) mt.mt_4(1) st(22) vm(16)	18614.3	0.158389	30
af(15) bp(14) cp(13) ft.ft_3(1) ht.ht_1(14) if(18) me(10) mt.mt_4(1) st(22) vm(16)	18611.9	0.164845	30
af(15) bp(15) cp(12) ft.ft_3(1) ht.ht_1(14) if(21) me(10) mt.mt_4(1) st(22) vm(16)	18612.1	0.164452	30
af(15) bp(15) cp(13) ft.ft_3(1) ht.ht_1(14) if(19) me(10) mt.mt_4(1) st(22) vm(16)	18612.3	0.163681	30
af(19) bp(16) cp(16) ft.ft_3(1) ht.ht_1(14) if(21) me(10) mt.mt_4(1) st(22) vm(16)	18616.9	0.15355	30
af(16) bp(15) cp(13) ft.ft_3(1) ht.ht_1(14) if(21) me(10) mt.mt_4(1) st(22) vm(16)	18613.1	0.161379	31
af(11) bp(10) cp(9) ft.ft_3(1) ht.ht_1(9) if(11) me(10) mt.mt_4(1) st(12) vm(10)	18602.8	0.230611	714
af(11) bp(10) cp(9) ft.ft_3(1) ht.ht_1(9) if(11) me(10) mt.mt_4(1) st(13) vm(10)	18602.9	0.227947	722
af(11) bp(10) cp(9) ft.ft_3(1) ht.ht_1(9) if(12) me(10) mt.mt_4(1) st(12) vm(9)	18602.6	0.235368	731
af(13) bp(13) cp(12) ft.ft_3(1) ht.ht_1(14) if(18) me(10) mt.mt_4(1) st(21) vm(16)	18609.9	0.17213	738
af(11) bp(10) cp(9) ft.ft_3(1) ht.ht_1(9) if(12) me(10) mt.mt_4(1) st(13) vm(12)	18603.5	0.218261	747
af(13) bp(13) cp(12) ft.ft_3(1) ht.ht_1(14) if(19) me(10) mt.mt_4(1) st(21) vm(16)	18610	0.171743	759
af(12) bp(12) cp(11) ft.ft_3(1) ht.ht_1(13) if(15) me(10) mt.mt_4(1) st(20) vm(16)	18608.2	0.180325	762
af(11) bp(10) cp(9) ft.ft_3(1) ht.ht_1(9) if(11) me(10) mt.mt_4(1) st(12) vm(11)	18603	0.225891	850
af(11) bp(10) cp(9) ft.ft_3(1) ht.ht_1(9) if(12) me(10) mt.mt_4(1) st(14) vm(11)	18603.4	0.219942	1014
af(15) bp(15) cp(13) ft.ft_3(1) ht.ht_1(14) if(19) me(10) mt.mt_4(1) st(21) vm(16)	18612	0.164664	2446

7.3.2.4 Composite Evaluation

Under the composite IPM evaluation on the FOSS, the same components as under PPM were subjected to expert judgement with same expert PM times. This same principle also applies to components that were flagged for component substitution. Components subjected to expert judgement are as specified in Table 7.10 (section 7.3.1.2), whereas those flagged for substitution are as specified in Table 7.13 (also in section 7.3.1.2).

The composite IPM evaluation portrays a real life engineering optimisation problem where the given system model is subjected to both primary and secondary constraints in a single instance of IPM optimisation run. The Pareto frontier of the composite IPM optimisation is shown in Figure 7.11. The summary of the optimisation is shown in Table 7.30 and shows that a total of 121 optimal IPM schedules were found. Although the search progressed through to 5120 generations, the last optimal IPM schedule was found in generation 1041.

Table 7.31 shows the number of occurrences of component implementations forming part of the FOSS composite IPM optimisation solution vector. The component implementation summary in Table 7.31 is similar to that of Table 7.28 (number of occurrences of component implementations forming part of the FOSS IPM optimisation solution vector under component substitution). However, the expert judgement has an effect on the objective function evaluations under the composite IPM optimisation, and as seen in Table 7.31, component *Mixing tank* has implementation *mixing_tank_2* dominating the entire optimal IPM schedules.

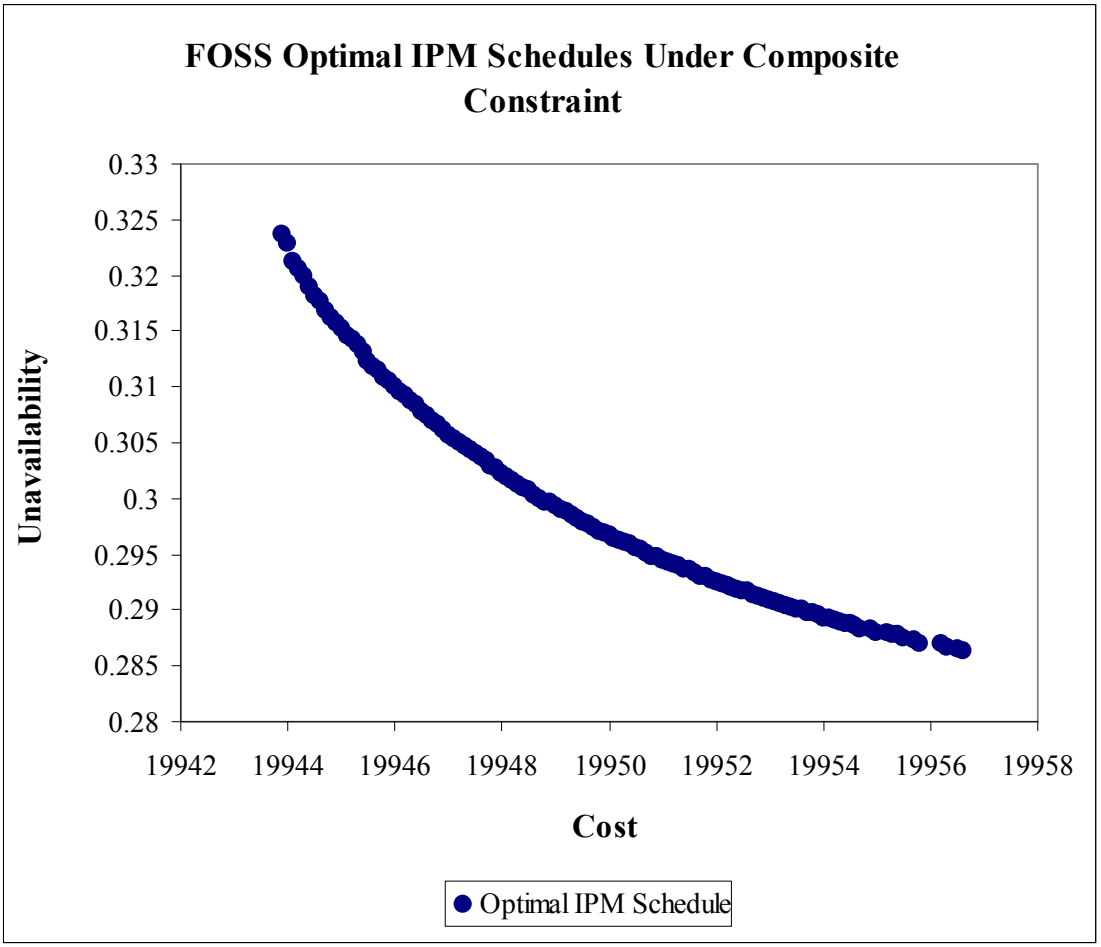

Figure 7.11 - Pareto frontier of IPM schedules under composite constraint

Table 7.30 - Results summary for the Pareto frontier of IPM schedules under composite constraint

Optimisation Indicators	**Value**
IPM Solution Space **X**	> 15,516,041,187,205,900,000
Feasible IPM Region ***f*X**	> 2,568,674,820,096
Number of solutions (optimal IPM schedules) found	121
Total number of generations the search was subjected	5120
Generation for which last solution(s) was/were found	1041

Table 7.31 - Number of occurrences of component implementations forming part of the FOSS composite IPM optimisation solution vector

Component	Implementations	Number of Occurrence
Heater	Heater_1	121
	Heater_2	0
	Heater_3	0
	Heater_4	0
Mixing tank	Mixing_tank_1	0
	Mixing_tank_2	121
	Mixing_tank_3	0
	Mixing_tank_4	0
Flow meter	Flow_meter_1	0
	Flow_meter_2	0
	Flow_meter_3	121
	Flow_meter_4	0
	Flow_meter_5	0

Table 7.32 shows the first and last 10 out of the 121 composite IPM schedules found. The components subjected to expert judgement have fixed CoMIs as expected. These components are *Main engine*, *Service tank* and *Viscosimeter* having 8, 7 and 4 CoMIs respectively in all the optimal IPM schedules.

The first three generations of the composite IPM scheduling optimisation are shown in Tables E.1 – E.3 in appendix E. The tables show the diverse component implementations existing in early generations of the composite IPM scheduling optimisation.

Table 7.32 - A subset of optimal IPM schedules under composite constraints; a tabular representation

Optimal PPM Schedule	Cost	Unavailability	Generation
af(14) bp(14) cp(12) fm.fm_3(1) ht.ht_1(14) if(18) me(8) mt.mt_2(1) st(7) vm(4)	19948.2	0.301541	48
af(14) bp(14) cp(11) fm.fm_3(1) ht.ht_1(14) if(17) me(8) mt.mt_2(1) st(7) vm(4)	19947.7	0.303325	50
af(12) bp(11) cp(10) fm.fm_3(1) ht.ht_1(14) if(17) me(8) mt.mt_2(1) st(7) vm(4)	19945.7	0.31149	51
af(14) bp(14) cp(12) fm.fm_3(1) ht.ht_1(14) if(17) me(8) mt.mt_2(1) st(7) vm(4)	19948.1	0.301907	51
af(14) bp(14) cp(11) fm.fm_3(1) ht.ht_1(14) if(18) me(8) mt.mt_2(1) st(7) vm(4)	19947.8	0.30296	53
af(12) bp(12) cp(11) fm.fm_3(1) ht.ht_1(14) if(16) me(8) mt.mt_2(1) st(7) vm(4)	19946.2	0.309199	55
af(14) bp(14) cp(12) fm.fm_3(1) ht.ht_1(14) if(19) me(8) mt.mt_2(1) st(7) vm(4)	19948.3	0.301214	55
af(12) bp(12) cp(12) fm.fm_3(1) ht.ht_1(14) if(16) me(8) mt.mt_2(1) st(7) vm(4)	19946.5	0.307793	55
af(12) bp(12) cp(10) fm.fm_3(1) ht.ht_1(14) if(17) me(8) mt.mt_2(1) st(7) vm(4)	19945.9	0.310474	55
af(12) bp(11) cp(10) fm.fm_3(1) ht.ht_1(14) if(16) me(8) mt.mt_2(1) st(7) vm(4)	19945.6	0.311898	56
af(12) bp(12) cp(11) fm.fm_3(1) ht.ht_1(13) if(16) me(8) mt.mt_2(1) st(7) vm(4)	19946.1	0.309609	382
af(11) bp(10) cp(9) fm.fm_3(1) ht.ht_1(9) if(11) me(8) mt.mt_2(1) st(7) vm(4)	19943.9	0.323618	419
af(12) bp(13) cp(11) fm.fm_3(1) ht.ht_1(14) if(16) me(8) mt.mt_2(1) st(7) vm(4)	19946.4	0.308336	425
af(12) bp(13) cp(11) fm.fm_3(1) ht.ht_1(13) if(16) me(8) mt.mt_2(1) st(7) vm(4)	19946.3	0.308747	452
af(13) bp(13) cp(11) fm.fm_3(1) ht.ht_1(13) if(16) me(8) mt.mt_2(1) st(7) vm(4)	19946.8	0.306676	464
af(11) bp(10) cp(9) fm.fm_3(1) ht.ht_1(11) if(14) me(8) mt.mt_2(1) st(7) vm(4)	19944.3	0.319987	553
af(11) bp(11) cp(11) fm.fm_3(1) ht.ht_1(13) if(16) me(8) mt.mt_2(1) st(7) vm(4)	19945.4	0.313054	554
af(11) bp(12) cp(10) fm.fm_3(1) ht.ht_1(13) if(16) me(8) mt.mt_2(1) st(7) vm(4)	19945.3	0.313716	560
af(11) bp(10) cp(9) fm.fm_3(1) ht.ht_1(11) if(13) me(8) mt.mt_2(1) st(7) vm(4)	19944.2	0.320591	676
af(11) bp(10) cp(9) fm.fm_3(1) ht.ht_1(10) if(13) me(8) mt.mt_2(1) st(7) vm(4)	19944.1	0.321262	1041

7.4 Further Evaluation of PM Scheduling Optimisation

To demonstrate the scalability of the defined scheduled preventive maintenance optimisation problem, the approach was further evaluated on a larger model, an *aircraft wheel brake system* (AWBS). This evaluation was performed under the composite constraint and also under PPM policy only.

7.4.1 Further Case Study – Aircraft Wheel Brake System

The model of the AWBS found in Sharvia (2010) is here adapted and it is as shown in Figure 7.12. Due to the nature of the AWBS design diagram, the *in* and *out* ports of its constituent components unlike those of the FOSS are here labelled within the components.

The wheel brake system provides safe baking for the aircraft during taxiing. Safe braking implies the supply of correct pressure to the brake actuator or the *wheel brake system (WBS*) seen in Figure 7.12. This way, skidding or taxiing beyond or before expected location could be prevented. The wheel brake system also prevents the occurrence of unintended aircraft motion, especially when parked.

The AWBS consists of two primary hydraulic pumps; *Green* and *Blue*. In normal mode of braking, the *Green pump* provides the required hydraulic pressure while the *Blue pump* provides pressure in alternative mode. The alternative mode becomes active when failure occurs in normal mode. The *Green valve* and *Blue valve* control pressure from the *Green pump* and *Blue pump* respectively. In normal mode the *Green valve* and *Blue valve* are both opened to provide constant stream of pressure to the *Selector valve*. However, only one of the two redundant hydraulic lines is selected by the *Selector valve* to prevent a scenario where both provide braking pressure.

In a normal braking mode, the *brake system control unit* (*BSCU*) can receive as input the *Brake pedal position* which it processes to produce control signals for braking. The *BSCU* also receives several other input signals which are continuously being monitored. These inputs indicate certain critical aircraft and system (AWBS) states so that the

correct braking function is achieved to improve fault tolerance mechanism. The *BSCU* basically computes braking and anti-skid commands and transmits the signal to the appropriate braking channel. The *Aircraft speed* and *Deceleration rate* are used when *Auto brake* is true.

The *Selector valve* receives braking pressure from the *Green pump* and in addition to the control signal received from the *BSCU*, brake pressure is further transmitted to the *CMD/AS meter valveG* which measures the amount of brake pressure and adjusts the valve position to output the required amount of pressure based on the command issued by the *BSCU*. The brake pressure is further transmitted to the *WBS* through normal pressure *NormalP*.

Should failure exist in control pressure which emanates from *CMD/AS meter valveG*, a signal is sent to the *BSCU* to put the AWBS into alternative mode and the braking process continues in this mode. In addition, the AWBS enters alternative mode when (i) *Green pump* produces pressure below threshold or pressure is omitted, or (ii) when any other failure occur along the *Green pump* line. Once an alternative mode is activated, an *OnAlternative* signal is sent to inform the *Selector valve* to ignore any pressure from the *Green valve*. Once the AWBS goes into alternative mode, reverting to normal mode is impossible during the mission time of the aircraft.

The *Selector valve* in alternative mode receives braking control pressure from the *Blue pump* in addition to brake control signal from the *BSCU*. Brake pressure is then transmitted to the *CMD/AS meter valveB* which also measures the amount of brake pressure and adjusts the valve position to output the required amount of pressure based on the command issued by the *BSCU*. The brake pressure is further transmitted to the *WBS* through alternative pressure *AlternativeP*.

As an increased safety measure, the AWBS comprise of an *Accumulator valve* which continuously receives pressure from *Accumulator pump*. The *Accumulator valve* also receives control signal from the *BSCU* in order to be informed of what mode is in force. In *Alternative* mode primary channel of output is given to the *CMD/AS meter valveB*

and the *Accumulator valve* is left redundant. However when there is no pressure from the *Selector valve* or the pressure falls under threshold when the AWBS is in *Alternative mode*, the AWBS enters emergency mode and pressure from the *Accumulator pump* is released to the *Accumulator valve* and the braking pressure is transmitted to the *Manual meter valve*. The *Manual meter valve* also receives as input, pressure from *Mechanical pedal* which serves as an extra safety measure. The *Manual meter valve* regulates pressure from the *Selector valve* and the *Accumulator pump*. The pressure is further transmitted to the *WBS* through emergency pressure *EmergencyP*.

In order to analyse the model of the AWBS, its constituent components were as in the case of the FOSS annotated in HiP-HPS with their respective failure data. The failure expression for each component is as shown in Table 7.33. These failure expressions model the failure behaviour of the respective components and are here defined similar to that found in Sharvia (2010). They are specified as lists of internal failure modes of the components and lists of deviations of parameters at component outputs. The *BSCU* has two types of internal failure modes, which are related to the monitor and command units. The analysis is also focused on omission of function as is the case for the FOSS.

Table 7.33 - Components failure expression for the aircraft wheel brake system

Component	**Failure Expression**
Green pump	`greenPumpFailure`
Green valve	`O-in or greenValveFailure`
Blue pump	`bluePumpFailure`
Blue valve	`O-in or blueValveFailure`
Selector valve	**`At out1:`** `O-in1 or O-in2 or selectorValveFailure`
	`At out3:` `O-in3 or O-in2 or selectorValveFailure`

CMD/AS meter valveG	O-in1 or O-in2 or CMDASMeterValveGFailure
CMD/AS meter valveB	O-in1 or O-in2 or CMDASMeterValveBFailure
Accumulator pump	accumulatorPumpFailure
Accumulator valve	(O-in1 and O-in2) or O-in3 or accumulatorValveFailure
Manual meter valve	O-in1 or O-in2 or manualMeterValveFailure
NormalP	O-in or normalPFailure
AlternativeP	O-in or alternativePFailure
EmergencyP	O-in or emergencyPFailure
Mechanical pedal	O-in or mechanicalPedalFailure
WBS	(O-in1 and O-in2 and O-in3) or WBSFailure
BSCU	BSCUCommandFailure
	BSCUMonitorFailure

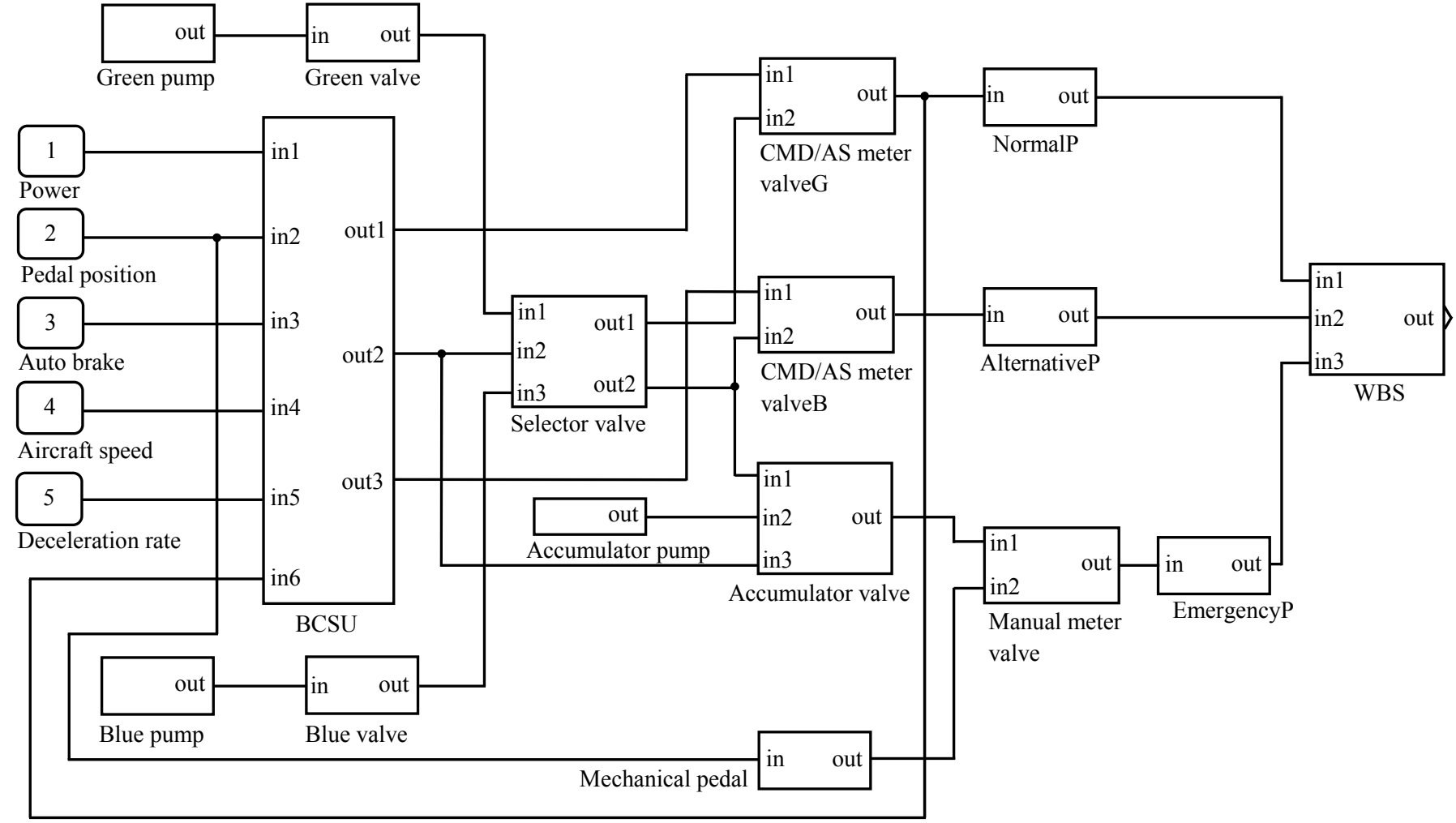

Figure 7.12 - Aircraft Wheel Brake System

7.4.2 PPM Composite Evaluation on AWBS

The composite PPM evaluation on the AWBS was performed and few of its constituent components were selected for expert judgement on PM time while few others were subjected to component substitution. Due to space limitation, Table 7.34 presents a list of the AWBS components and their corresponding short names which are used in the presentation of results obtained. The components that were subjected to expert judgement are *Manual meter valve*, *Selector valve* and *BSCU*, these are presented in Table 7.35 with their corresponding expert judged PM times. Subjected to component substitution are *Accumulator valve*, *WBS*, *CMD/AS meter valveG* and *Blue pump*, presented in Table 7.36 with their implementation options.

Table 7.34 - Short name representation of actual components name of the AWBS

Actual Component Name	**Short Name**
Accumulator pump	ap
Accumulator valve	av
AlternativeP	apr
Blue pump	bp
Blue valve	bv
BSCU	bscu
CMD/AS meter valveB	mvb
CMD/AS meter valveG	mvg
EmergencyP	epr
Green pump	gp
Green valve	gv
Manual meter valve	mmv
Mechanical pedal	mp
NormalP	npr
Selector valve	sv
WBS	wbs

Table 7.35 - Selected components for expert judgement with their respective expert PM times

Components	**Expert PM Time**
Manual meter valve	2030
Selector valve	3060
BSCU	4300

Table 7.36 - Components with their implementations and short name representation

Component	**Implementations**	**Short Name**
Accumulator valve	Accumulator_valve_1	av_1
	Accumulator_valve _2	av _2
	Accumulator_valve _3	av _3
	Accumulator_valve _4	av _4
CMD/AS meter valveG	CMD_AS_meter_valveG_1	mvg_1
	CMD_AS_meter_valveG _2	mvg _2
	CMD_AS_meter_valveG _3	mvg _3
Blue pump	Blue_pump_1	bp_1
	Blue_pump _2	bp _2
	Blue_pump _3	bp _3
WBS	WBS_1	wbs_1
	WBS_2	wbs_2
	WBS_3	wbs_3
	WBD_4	wbs_4
	WBS_5	wbs_5

The search is stretched through to same number of generations - 5120 - as for the FOSS. To show that the system shortest PM interval T can vary depending on the intuition of the engineer and the type of system in question, a shortest PM interval of 210 time units is used for the AWBS.

The Pareto frontier of the results obtained from the composite PPM optimisation is shown in Figure 7.13 while the summary is as shown in Table 7.37. Table 7.37 shows that a total of 1595 optimal PPM schedules were found, with the last one found in generation 5119. With the last optimal PPM schedule found in generation 5119 implies that more optimal PPM schedules were likely to be found beyond generation 5120.

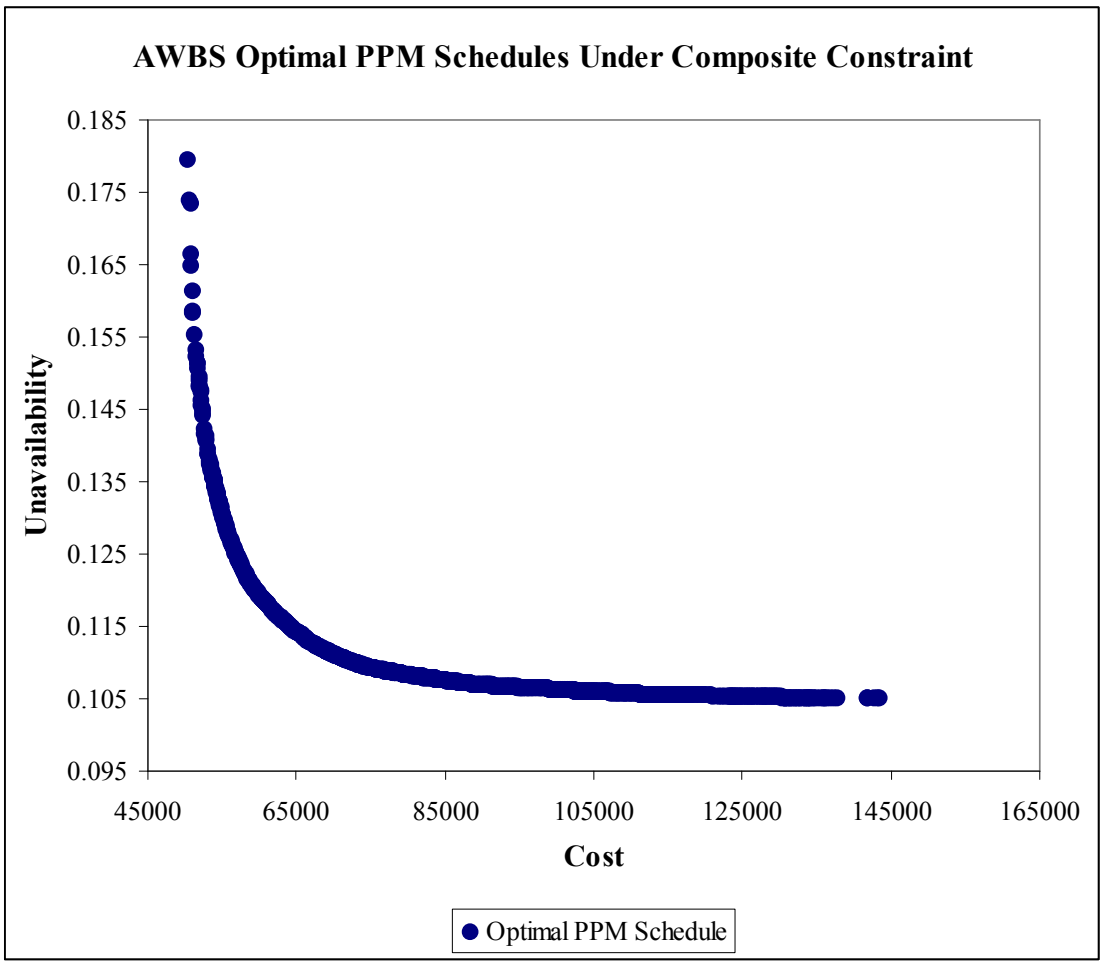

Figure 7.13 - AWBS Pareto frontier of PPM schedules under composite constraint

Table 7.37 - Results summary for the Pareto frontier of PPM schedules under composite constraint

Optimisation Indicators	**Value**
PPM Solution Space **X**	> 1,617,154,011,038,070,000,000,000,000,000,000
Feasible PPM Region ***f*X**	> 4,747,762,302,540,820,000,000,000
Number of solutions (optimal PPM schedules) found	1595
Total number of generations the search was subjected	5120
Generation for which last solution was found	5119

Table 7.38 shows the number of occurrences of component implementations forming part of the AWBS composite PPM optimisation solution vector. Component *Accumulator valve* has implementation *Accumulator_valve_1* dominating the entire optimal PPM schedules, *CMD/AS meter valveG* has implementation *CMD_AS_meter_valveG_1* dominating the entire optimal PPM schedules and so does implementation *Blue_pump_3* for component *Blue pump*. Similarly, component *BWS* has implementation *bws_4* dominating the entire optimal PPM schedules.

Table 7.39 shows the first and last 10 out of the 1595 optimal PPM schedules found. The table shows that the components *Manual meter valve*, *Selector valve* and *BSCU* which were subjected to expert judgement have fixed CoMIs in all the optimal PPM schedules.

The optimal PPM schedules found in the first three generations of the composite PPM scheduling optimisation are shown in Tables Table 7.40, Table 7.41 and Table 7.42. The tables show the diverse component implementations existing in early generations of the composite PPM scheduling optimisation. For instance, in these early generations implementations *Accumulator_valve_1* and *Accumulator_valve_4* were found suitable for component *Accumulator valve* while implementations *wbs_1*, *wbs_2*, *wbs_4* and *wbs_5* were found suitable for component *WBS*. For component *CMD/AS meter valveG*,

implementations *CMD_AS_meter_valveG_1*, *CMD_AS_meter_valveG_2* and *CMD_AS_meter_valveG_3* were found suitable while implementations *Blue_pump_1* and *Blue_pump_3* were found suitable for component *Blue pump*. This characteristic of having several mixtures of a given component implementations was expected at the infancy stage of the optimisation. Through progressive generations, the choice of suitable implementations for a given component is likely to narrow.

Table 7.38 - Number of occurrences of component implementations forming part of the AWBS composite PPM optimisation solution vector

Component	**Implementations**	**Number of Occurrence**
Accumulator valve	Accumulator_valve_1	1595
	Accumulator_valve _2	0
	Accumulator_valve _3	0
	Accumulator_valve _4	0
CMD/AS meter valveG	CMD_AS_meter_valveG_1	1595
	CMD_AS_meter_valveG _2	0
	CMD_AS_meter_valveG _3	0
Blue pump	Blue_pump_1	0
	Blue_pump _2	0
	Blue_pump _3	1595
WBS	WBS_1	0
	WBS_2	0
	WBS_3	0
	WBD_4	1595
	WBS_5	0

Table 7.39 - A subset of AWBS optimal PPM schedules under composite constraint; a tabular representation

Optimal PPM Schedule	Cost	Unavailability	Generation
ap(11) av.av_1(8) apr(13) bp.bp_3(16) bv(20) bscu(20) mvb(23) mvg.mvg_1(15) epr(4) gp(22) gv(20) mmv(9) mp(8) npr(13) sv(14) wbs.wbs_4(2)	59200	0.120365	110
ap(11) av.av_1(8) apr(13) bp.bp_3(16) bv(20) bscu(20) mvb(23) mvg.mvg_1(15) epr(4) gp(22) gv(20) mmv(9) mp(8) npr(10) sv(14) wbs.wbs_4(2)	59350	0.120206	112
ap(11) av.av_1(5) apr(13) bp.bp_3(16) bv(15) bscu(20) mvb(23) mvg.mvg_1(11) epr(4) gp(22) gv(20) mmv(9) mp(8) npr(13) sv(14) wbs.wbs_4(2)	61020	0.118188	112
ap(11) av.av_1(8) apr(13) bp.bp_3(16) bv(20) bscu(20) mvb(23) mvg.mvg_1(15) epr(5) gp(22) gv(20) mmv(9) mp(8) npr(13) sv(14) wbs.wbs_4(2)	58900	0.120814	112
ap(11) av.av_1(8) apr(19) bp.bp_3(16) bv(20) bscu(20) mvb(23) mvg.mvg_1(15) epr(5) gp(22) gv(20) mmv(9) mp(8) npr(13) sv(14) wbs.wbs_4(2)	58750	0.121077	115
ap(11) av.av_1(6) apr(19) bp.bp_3(16) bv(20) bscu(20) mvb(23) mvg.mvg_1(15) epr(4) gp(22) gv(20) mmv(9) mp(8) npr(13) sv(14) wbs.wbs_4(2)	59850	0.119597	115
ap(11) av.av_1(6) apr(13) bp.bp_3(16) bv(15) bscu(20) mvb(23) mvg.mvg_1(15) epr(4) gp(22) gv(20) mmv(9) mp(8) npr(10) sv(14) wbs.wbs_4(2)	60570	0.118688	116
ap(11) av.av_1(6) apr(13) bp.bp_3(16) bv(20) bscu(20) mvb(23) mvg.mvg_1(15) epr(4) gp(22) gv(20) mmv(9) mp(8) npr(13) sv(14) wbs.wbs_4(2)	60000	0.119359	117
ap(11) av.av_1(8) apr(13) bp.bp_3(16) bv(15) bscu(20) mvb(23) mvg.mvg_1(15) epr(5) gp(22) gv(20) mmv(9) mp(8) npr(13) sv(14) wbs.wbs_4(2)	59320	0.120252	120
ap(11) av.av_1(7) apr(13) bp.bp_3(16) bv(15) bscu(20) mvb(23) mvg.mvg_1(15) epr(4) gp(22) gv(20) mmv(9) mp(8) npr(13) sv(14) wbs.wbs_4(2)	60020	0.119279	122
ap(1) av.av_1(1) apr(2) bp.bp_3(4) bv(2) bscu(20) mvb(4) mvg.mvg_1(4) epr(1) gp(6) gv(6) mmv(9) mp(1) npr(4) sv(14) wbs.wbs_4(1)	142760	0.105025	5030
ap(11) av.av_1(8) apr(13) bp.bp_3(24) bv(20) bscu(20) mvb(23) mvg.mvg_1(22) epr(5) gp(22) gv(20) mmv(9) mp(8) npr(13) sv(14) wbs.wbs_4(2)	58250	0.122097	5033

ap(11) av.av_1(8) apr(19) bp.bp_3(16) bv(20) bscu(20) mvb(16) mvg.mvg_1(15) epr(5) gp(22) gv(20) mmv(9) mp(8) npr(13) sv(14) wbs.wbs_4(2)	59050	0.120685	5043
ap(22) av.av_1(22) apr(19) bp.bp_3(24) bv(29) bscu(20) mvb(23) mvg.mvg_1(22) epr(8) gp(33) gv(29) mmv(9) mp(19) npr(19) sv(14) wbs.wbs_4(6)	51610	0.152272	5044
ap(2) av.av_1(1) apr(4) bp.bp_3(4) bv(5) bscu(20) mvb(4) mvg.mvg_1(4) epr(1) gp(6) gv(6) mmv(9) mp(1) npr(3) sv(14) wbs.wbs_4(1)	121920	0.105336	5065
ap(13) av.av_1(10) apr(13) bp.bp_3(24) bv(20) bscu(20) mvb(23) mvg.mvg_1(22) epr(5) gp(33) gv(29) mmv(9) mp(10) npr(13) sv(14) wbs.wbs_4(3)	55680	0.128459	5066
ap(1) av.av_1(1) apr(4) bp.bp_3(4) bv(6) bscu(20) mvb(4) mvg.mvg_1(4) epr(1) gp(5) gv(6) mmv(9) mp(1) npr(4) sv(14) wbs.wbs_4(1)	134230	0.105074	5080
ap(40) av.av_1(22) apr(19) bp.bp_3(24) bv(29) bscu(20) mvb(27) mvg.mvg_1(28) epr(13) gp(33) gv(30) mmv(9) mp(23) npr(19) sv(14) wbs.wbs_4(17)	50610	0.173792	5090
ap(1) av.av_1(1) apr(2) bp.bp_3(6) bv(2) bscu(20) mvb(4) mvg.mvg_1(4) epr(1) gp(6) gv(5) mmv(9) mp(1) npr(4) sv(14) wbs.wbs_4(1)	141800	0.105038	5107
ap(2) av.av_1(1) apr(4) bp.bp_3(5) bv(4) bscu(20) mvb(4) mvg.mvg_1(4) epr(1) gp(6) gv(5) mmv(9) mp(1) npr(4) sv(14) wbs.wbs_4(1)	122670	0.10533	5119

Table 7.40 - AWBS optimal PPM schedules under composite constraint in generation 1

Optimal PPM Schedule	Cost	Unavailability	Generation
ap(29) av.av_3(20) apr(11) bp.bp_1(7) bv(31) bscu(20) mvb(27) mvg.mvg_1(14) epr(4) gp(52) gv(10) mmv(9) mp(4) npr(11) sv(14) wbs.wbs_1(3)	64820	0.136231	1
ap(43) av.av_1(17) apr(32) bp.bp_3(32) bv(23) bscu(20) mvb(9) mvg.mvg_2(27) epr(12) gp(61) gv(23) mmv(9) mp(8) npr(17) sv(14) wbs.wbs_4(8)	54950	0.156958	1
ap(11) av.av_1(22) apr(13) bp.bp_3(18) bv(13) bscu(20) mvb(18) mvg.mvg_3(17) epr(18) gp(27) gv(52) mmv(9) mp(8) npr(5) sv(14) wbs.wbs_5(1)	66290	0.133167	1
ap(13) av.av_1(14) apr(26) bp.bp_3(33) bv(25) bscu(20) mvb(36) mvg.mvg_1(29) epr(2) gp(46) gv(37) mmv(9) mp(7) npr(35) sv(14) wbs.wbs_4(18)	56430	0.149889	1
ap(11) av.av_4(6) apr(22) bp.bp_3(30) bv(30) bscu(20) mvb(30) mvg.mvg_2(19) epr(25) gp(36) gv(37) mmv(9) mp(13) npr(25) sv(14) wbs.wbs_4(11)	55550	0.150233	1
ap(22) av.av_4(23) apr(12) bp.bp_1(45) bv(38) bscu(20) mvb(21) mvg.mvg_1(9) epr(14) gp(37) gv(35) mmv(9) mp(28) npr(20) sv(14) wbs.wbs_2(18)	52610	0.178143	1

Table 7.41 - AWBS optimal PPM schedules under composite constraint in generation 2

Optimal PPM Schedule	Cost	Unavailability	Generation
ap(11) av.av_1(22) apr(12) bp.bp_1(45) bv(13) bscu(20) mvb(21) mvg.mvg_1(9) epr(14) gp(37) gv(52) mmv(9) mp(8) npr(20) sv(14) wbs.wbs_2(4)	56430	0.142736	2
ap(11) av.av_1(14) apr(26) bp.bp_3(33) bv(30) bscu(20) mvb(36) mvg.mvg_1(29) epr(25) gp(46) gv(37) mmv(9) mp(13) npr(25) sv(14) wbs.wbs_4(9)	52960	0.154536	2
ap(22) av.av_1(17) apr(12) bp.bp_3(11) bv(38) bscu(20) mvb(9) mvg.mvg_1(9) epr(12) gp(37) gv(35) mmv(9) mp(8) npr(17) sv(14) wbs.wbs_4(8)	56360	0.143386	2
ap(11) av.av_4(6) apr(22) bp.bp_3(18) bv(30) bscu(20) mvb(30) mvg.mvg_3(17) epr(18) gp(36) gv(37) mmv(9) mp(8) npr(25) sv(14) wbs.wbs_5(1)	65570	0.131412	2
ap(13) av.av_1(14) apr(26) bp.bp_1(45) bv(25) bscu(20) mvb(36) mvg.mvg_1(29) epr(14) gp(46) gv(35) mmv(9) mp(28) npr(20) sv(14) wbs.wbs_2(18)	52815	0.171745	2
ap(13) av.av_4(6) apr(26) bp.bp_3(33) bv(25) bscu(20) mvb(36) mvg.mvg_2(19) epr(2) gp(46) gv(37) mmv(9) mp(7) npr(25) sv(14) wbs.wbs_4(11)	59170	0.137406	2

Table 7.42 - AWBS optimal PPM schedules under composite constraint in generation 3

Optimal PPM Schedule	Cost	Unavailability	Generation
ap(11) av.av_1(22) apr(12) bp.bp_1(45) bv(13) bscu(20) mvb(21) mvg.mvg_1(9) epr(14) gp(37) gv(52) mmv(9) mp(8) npr(20) sv(14) wbs.wbs_2(4)	56430	0.142736	2
ap(11) av.av_1(14) apr(26) bp.bp_3(33) bv(30) bscu(20) mvb(36) mvg.mvg_1(29) epr(25) gp(46) gv(37) mmv(9) mp(13) npr(25) sv(14) wbs.wbs_4(9)	52960	0.154536	2
ap(13) av.av_1(14) apr(26) bp.bp_3(33) bv(25) bscu(20) mvb(36) mvg.mvg_1(29) epr(2) gp(46) gv(37) mmv(9) mp(7) npr(20) sv(14) wbs.wbs_4(11)	56580	0.142196	3
ap(13) av.av_1(17) apr(26) bp.bp_3(33) bv(25) bscu(20) mvb(36) mvg.mvg_1(9) epr(2) gp(46) gv(35) mmv(9) mp(7) npr(17) sv(14) wbs.wbs_4(8)	57330	0.138152	3
ap(11) av.av_4(6) apr(12) bp.bp_3(18) bv(30) bscu(20) mvb(30) mvg.mvg_3(17) epr(18) gp(36) gv(37) mmv(9) mp(8) npr(17) sv(14) wbs.wbs_5(1)	66020	0.130363	3
ap(11) av.av_4(6) apr(22) bp.bp_3(33) bv(30) bscu(20) mvb(36) mvg.mvg_1(29) epr(18) gp(36) gv(37) mmv(9) mp(8) npr(25) sv(14) wbs.wbs_4(9)	55700	0.143943	3
ap(13) av.av_1(17) apr(26) bp.bp_3(33) bv(25) bscu(20) mvb(36) mvg.mvg_2(19) epr(2) gp(46) gv(37) mmv(9) mp(7) npr(25) sv(14) wbs.wbs_4(8)	57280	0.13965	3
ap(11) av.av_1(29) apr(26) bp.bp_3(33) bv(30) bscu(20) mvb(36) mvg.mvg_1(29) epr(25) gp(46) gv(37) mmv(9) mp(13) npr(25) sv(14) wbs.wbs_4(9)	52560	0.161767	3
ap(22) av.av_4(6) apr(22) bp.bp_3(18) bv(30) bscu(20) mvb(9) mvg.mvg_1(9) epr(12) gp(37) gv(35) mmv(9) mp(8) npr(25) sv(14) wbs.wbs_4(8)	56900	0.139971	3
ap(11) av.av_1(17) apr(12) bp.bp_3(11) bv(13) bscu(20) mvb(21) mvg.mvg_1(9) epr(12) gp(37) gv(35) mmv(9) mp(8) npr(20) sv(14) wbs.wbs_2(4)	58080	0.131025	3
ap(13) av.av_1(22) apr(12) bp.bp_1(45) bv(13) bscu(20) mvb(36) mvg.mvg_1(9) epr(14) gp(46) gv(37) mmv(9) mp(7) npr(25) sv(14) wbs.wbs_2(4)	56080	0.143475	3
ap(11) av.av_4(6) apr(22) bp.bp_3(33) bv(25) bscu(20) mvb(30) mvg.mvg_2(19) epr(2) gp(36) gv(37) mmv(9) mp(8) npr(25) sv(14) wbs.wbs_5(1)	67630	0.122529	3

ap(22) av.av_1(17) apr(22) bp.bp_3(11) bv(38) bscu(20) mvb(30) mvg.mvg_1(9) epr(12) gp(36) gv(35) mmv(9) mp(8) npr(17) sv(14) wbs.wbs_4(8)	54860	0.145554	3
ap(22) av.av_1(17) apr(12) bp.bp_3(14) bv(38) bscu(20) mvb(9) mvg.mvg_1(9) epr(12) gp(37) gv(35) mmv(9) mp(8) npr(17) sv(14) wbs.wbs_4(8)	55910	0.143934	3
ap(13) av.av_1(14) apr(26) bp.bp_1(45) bv(30) bscu(20) mvb(36) mvg.mvg_1(29) epr(14) gp(46) gv(35) mmv(9) mp(28) npr(20) sv(14) wbs.wbs_2(18)	52395	0.173307	3
ap(11) av.av_1(14) apr(26) bp.bp_3(33) bv(25) bscu(20) mvb(36) mvg.mvg_1(29) epr(25) gp(46) gv(37) mmv(9) mp(13) npr(25) sv(14) wbs.wbs_4(9)	53380	0.153041	3
ap(11) av.av_1(17) apr(12) bp.bp_3(33) bv(38) bscu(20) mvb(9) mvg.mvg_1(29) epr(12) gp(37) gv(35) mmv(9) mp(13) npr(7) sv(14) wbs.wbs_4(8)	55360	0.144357	3
ap(22) av.av_1(17) apr(22) bp.bp_3(18) bv(30) bscu(20) mvb(30) mvg.mvg_1(9) epr(12) gp(37) gv(35) mmv(9) mp(8) npr(17) sv(14) wbs.wbs_4(8)	53960	0.146165	3
ap(11) av.av_1(17) apr(22) bp.bp_3(18) bv(38) bscu(20) mvb(9) mvg.mvg_1(9) epr(18) gp(36) gv(37) mmv(9) mp(8) npr(17) sv(14) wbs.wbs_4(8)	56210	0.143179	3
ap(13) av.av_1(14) apr(26) bp.bp_3(33) bv(25) bscu(20) mvb(36) mvg.mvg_1(29) epr(14) gp(46) gv(37) mmv(9) mp(28) npr(24) sv(14) wbs.wbs_4(11)	52780	0.159155	3

7.5 Comparison between Manually and Automatically Optimised PM Schedules

Preventive maintenance scheduling is normally performed manually, which may be particularly problematic for complex systems. Since manual PM scheduling is time consuming only few of such schedules may be enumerated, hence restricting the optimisation to a subset of PM solution space that consists of fewer potential PM schedules. In this study a comparison of the manual approach with the techniques developed in the thesis was performed on the fuel oil service system case study.

7.5.1 Manual PPM Scheduling Optimisation

Experts from the shipping industry and the University of Hull were consulted and asked to perform a manual PPM scheduling. HiP-HOPS without the PPM optimisation capability was used only for the purpose of evaluating system objectives under the models and assumptions used in this thesis. The experts considered a shortest PM interval of 90 (T = 90) time units; in days. The experts used the approach and assumptions developed in this work, however since manual PM scheduling is cumbersome the strategy they employed was similar to that of Bris et al (2003). The strategy involves obtaining PM schedules guided by a set maximum value for system unavailability and cost. The values considered by the experts are 0.3 for unavailability and 40000 for cost.

Six PPM schedules were manually obtained which were considered good solutions shown in Table 7.43. The "ID" column in the table implies "identity" and it represents a given PPM schedule. These IDs are used as reference points in comparing the manually obtained PPM schedules with those automatically obtained. Typically, it is difficult to work out from a tabular representation of PM schedules whether the schedules are approximations to the Pareto front. Therefore the graphical representation of Table 7.43 is presented in Figure 7.14.

Table 7.43 - Manually obtained PPM schedules

ID	PPM Schedule	Cost	Unavailability
M1	af(35) bp(29) cp(26) fm(17) ht(6) if(34) me(25) mt(15) st(39) vm(26)	19642	0.235366
M2	af(33) bp(34) cp(2) fm(13) ht(14) if(10) me(31) mt(5) st(22) vm(6)	25083	0.161624
M3	af(14) bp(29) cp(22) fm(8) ht(23) if(10) me(29) mt(9) st(8) vm(1)	27251	0.131278
M4	af(3) bp(7) cp(11) fm(7) ht(24) if(6) me(25) mt(8) st(5) vm(11)	26679	0.088344
M5	af(2) bp(8) cp(2) fm(1) ht(2) if(24) me(7) mt(14) st(15) vm(13)	37487	0.093716
M6	af(5) bp(11) cp(7) fm(3) ht(7) if(1) me(10) mt(3) st(4) vm(16)	32042	0.066594

From Figure 7.14 it can be observed that the manually obtained PPM schedules are not approximations to the Pareto front. M1 is the PPM schedule with the least cost but also results into the highest system unavailability. M5 is the PPM schedule that is most expensive while M6 gives the least system unavailability.

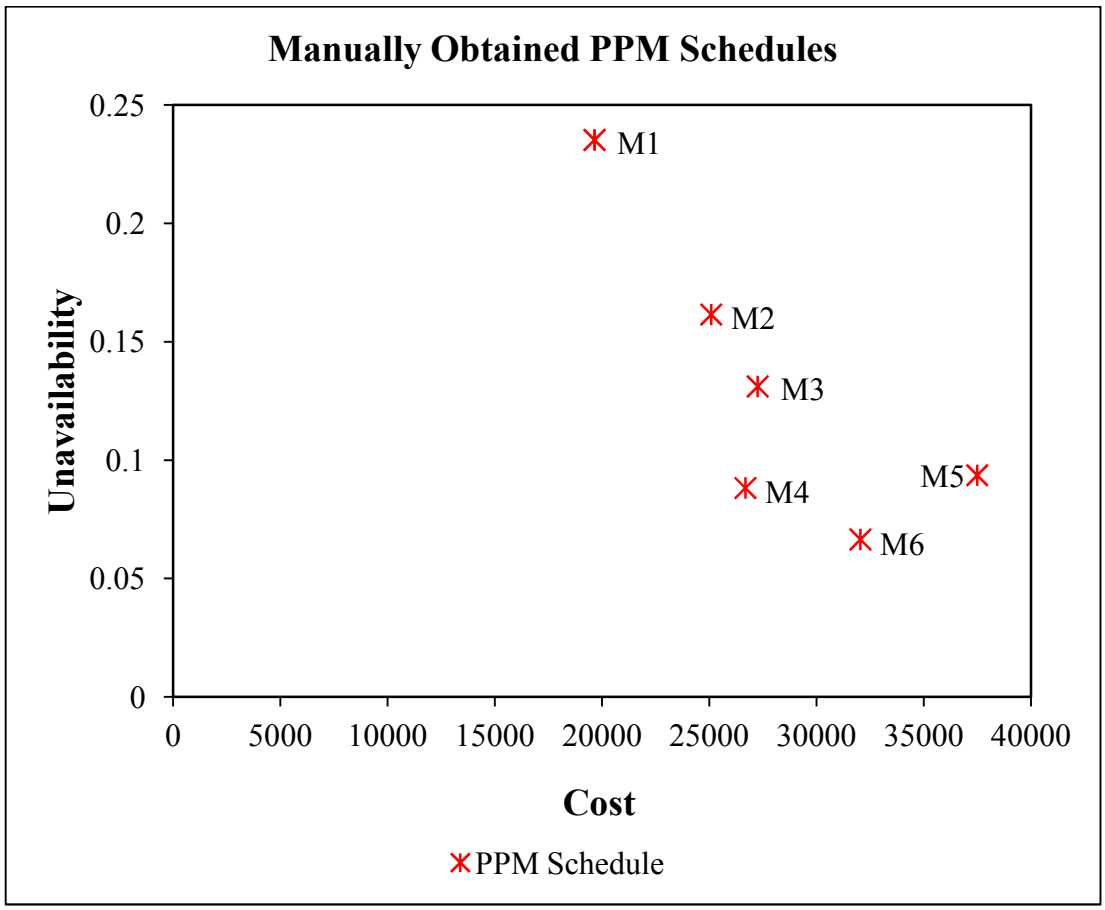

Figure 7.14 - Pareto frontier of manually obtained PPM schedules

7.5.2 Automated PM Scheduling Optimisation

In order to appropriately compare the results of the automated PM scheduling optimisation with those obtained through manual process, the same PM interval of 90 time units was used (i.e. T = 90) for the automated optimisation. The automated optimisation was run through 5120 generations and produced a total of 684 optimal or near optimal PPM schedules with the last schedule found in generation 5052. The results summary is shown in

Table 7.44. The first and last 10 of these PPM schedules are shown in Table 7.45. The graphical representation of the entire 684 PPM schedules obtained is shown in Figure 7.15.

Table 7.44 - Results summary of automated PPM optimisation

Optimisation Indicators	**Value**
PPM Solution Space **X**	15,888,426,175,698,800,000,000
Feasible PPM Region ***f*X**	2,365,899,008,901,120
Number of optimal PPM schedules found	684
Total number of generations the search was subjected	5120
Generation for which last solution(s) was/were found	5052

The summary in

Table 7.44 shows that the solution space is in sextillion while the subset of this population size that are potentially feasible solutions is in quadrillion. It is therefore manually impossible to exhaustively explore the potential PPM schedules in the feasible region.

Table 7.45 - A subset of optimal PPM schedules obtained through automation

Optimal PPM Schedule	Cost	Unavailability	Generation
af(22) bp(10) cp(17) fm(12) ht(6) if(7) me(27) mt(13) st(15) vm(17)	20378	0.138254	51
af(9) bp(6) cp(9) fm(7) ht(3) if(4) me(30) mt(7) st(9) vm(7)	24295	0.073331	58
af(22) bp(13) cp(17) fm(12) ht(6) if(9) me(27) mt(13) st(15) vm(17)	20210	0.144162	59
af(9) bp(6) cp(9) fm(7) ht(4) if(6) me(30) mt(9) st(9) vm(7)	23732	0.07831	62
af(9) bp(6) cp(11) fm(7) ht(4) if(6) me(27) mt(9) st(9) vm(7)	23482	0.080909	63
af(9) bp(5) cp(11) fm(9) ht(5) if(6) me(20) mt(9) st(9) vm(9)	23160	0.08445	64
af(9) bp(5) cp(9) fm(6) ht(3) if(6) me(33) mt(9) st(9) vm(7)	24075	0.074918	65
af(9) bp(5) cp(9) fm(6) ht(5) if(6) me(20) mt(9) st(9) vm(7)	23911	0.0767	67
af(9) bp(6) cp(9) fm(7) ht(3) if(6) me(29) mt(9) st(9) vm(9)	23639	0.079228	67
af(9) bp(7) cp(11) fm(7) ht(4) if(6) me(27) mt(9) st(9) vm(7)	23384	0.082069	70
af(2) bp(1) cp(3) fm(1) ht(1) if(1) me(36) mt(2) st(2) vm(1)	52844	0.017484	2961
af(2) bp(1) cp(2) fm(1) ht(1) if(1) me(33) mt(2) st(1) vm(1)	60064	0.0148	3755
af(2) bp(1) cp(2) fm(1) ht(1) if(1) me(23) mt(1) st(1) vm(1)	62476	0.013948	4656
af(2) bp(1) cp(2) fm(1) ht(1) if(1) me(33) mt(1) st(1) vm(2)	59676	0.015084	4676
af(1) bp(1) cp(2) fm(1) ht(1) if(1) me(33) mt(2) st(1) vm(1)	65629	0.013309	4791
af(1) bp(1) cp(1) fm(1) ht(1) if(1) me(23) mt(1) st(1) vm(1)	72041	0.011318	4928
af(1) bp(1) cp(2) fm(1) ht(1) if(1) me(23) mt(1) st(1) vm(1)	68041	0.012456	4970
af(2) bp(1) cp(1) fm(1) ht(1) if(1) me(33) mt(2) st(1) vm(1)	64064	0.013664	4971
af(1) bp(1) cp(1) fm(1) ht(1) if(1) me(14) mt(2) st(1) vm(1)	70239	0.012172	5049
af(2) bp(1) cp(1) fm(1) ht(1) if(1) me(23) mt(1) st(1) vm(1)	66476	0.012812	5052

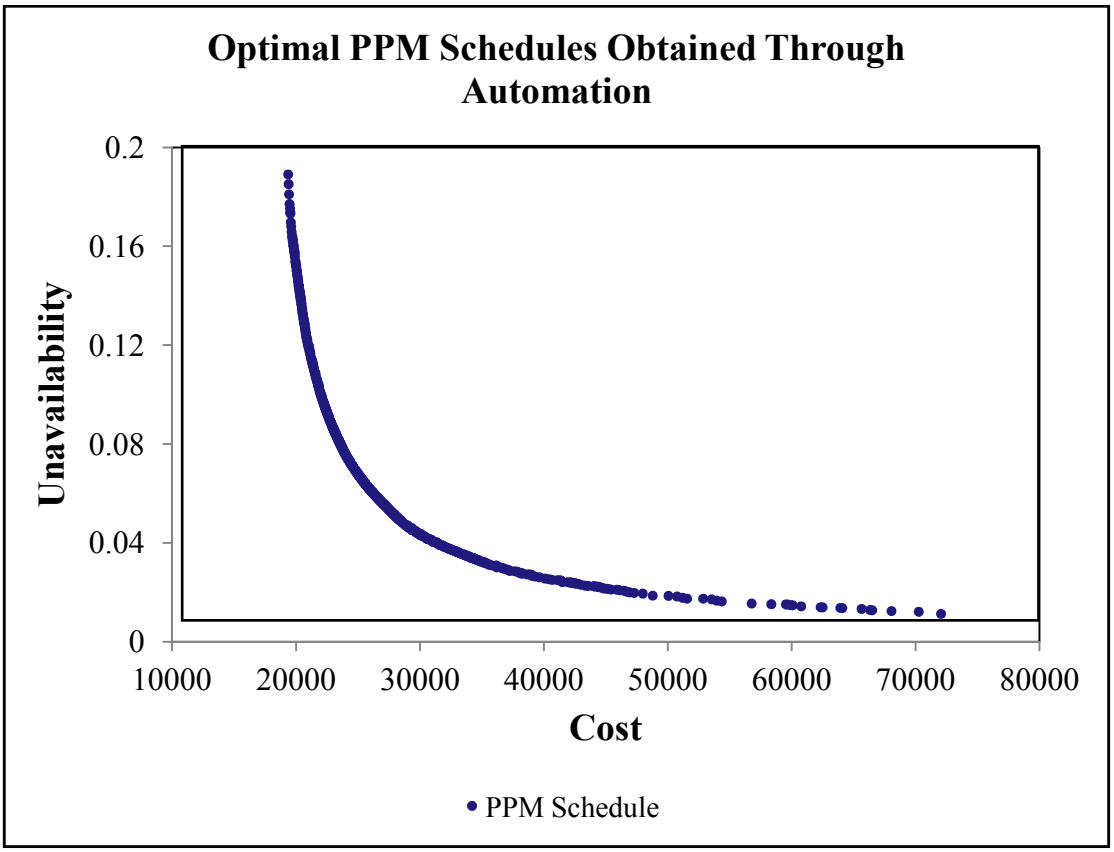

Figure 7.15 - Pareto frontier of PPM schedules obtained through automation

7.5.3 Automatically versus Manually Obtained PPM Schedules

To compare the results obtained through automation with those obtained through manual process Figure 7.14 and Figure 7.15 are combined into a single figure producing Figure 7.16. Since the goal of the optimisation is to minimise both objective functions (i.e. unavailability and cost), better approximations to the Pareto front will form a curve that spreads from the least possible value of one of the objective functions to the other. It can therefore be observed from Figure 7.16 that the manually obtained PPM schedules are inferior to those that were automatically obtained. In simple terms and according to the goal of the optimisation, the manually obtained PPM schedules appear in the front (or at the right) of the automatically obtained PPM schedules and are thus inferior.

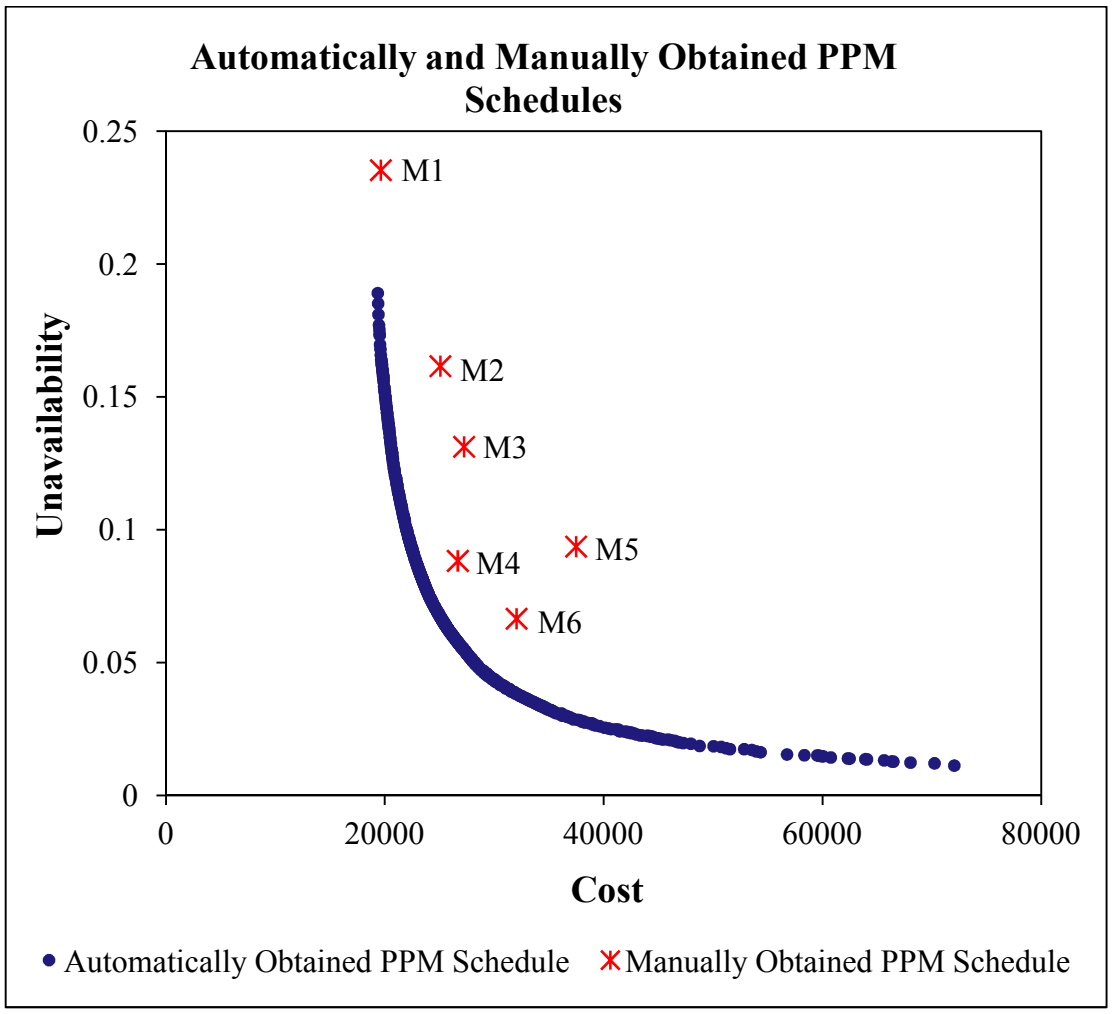

Figure 7.16 – Manually obtained PPM schedules and Pareto frontier of automatically obtained PPM schedules

There was no limit imposed on the maximum values of unavailability and cost for the automated PPM scheduling optimisation. However to specifically compare the two processes in the context of the strategy employed by the experts, automatically obtained PPM schedules with unavailability and cost value less than or equal to 0.3 and 40000 respectively were extracted from Figure 7.16 tranforming it to Figure 7.17. Figure 7.17 contains a total of 625 automatically obtained optimal or near optimal PPM schedules meeting the criteria as opposed to the 6 PPM schedules enumerated through manual process.

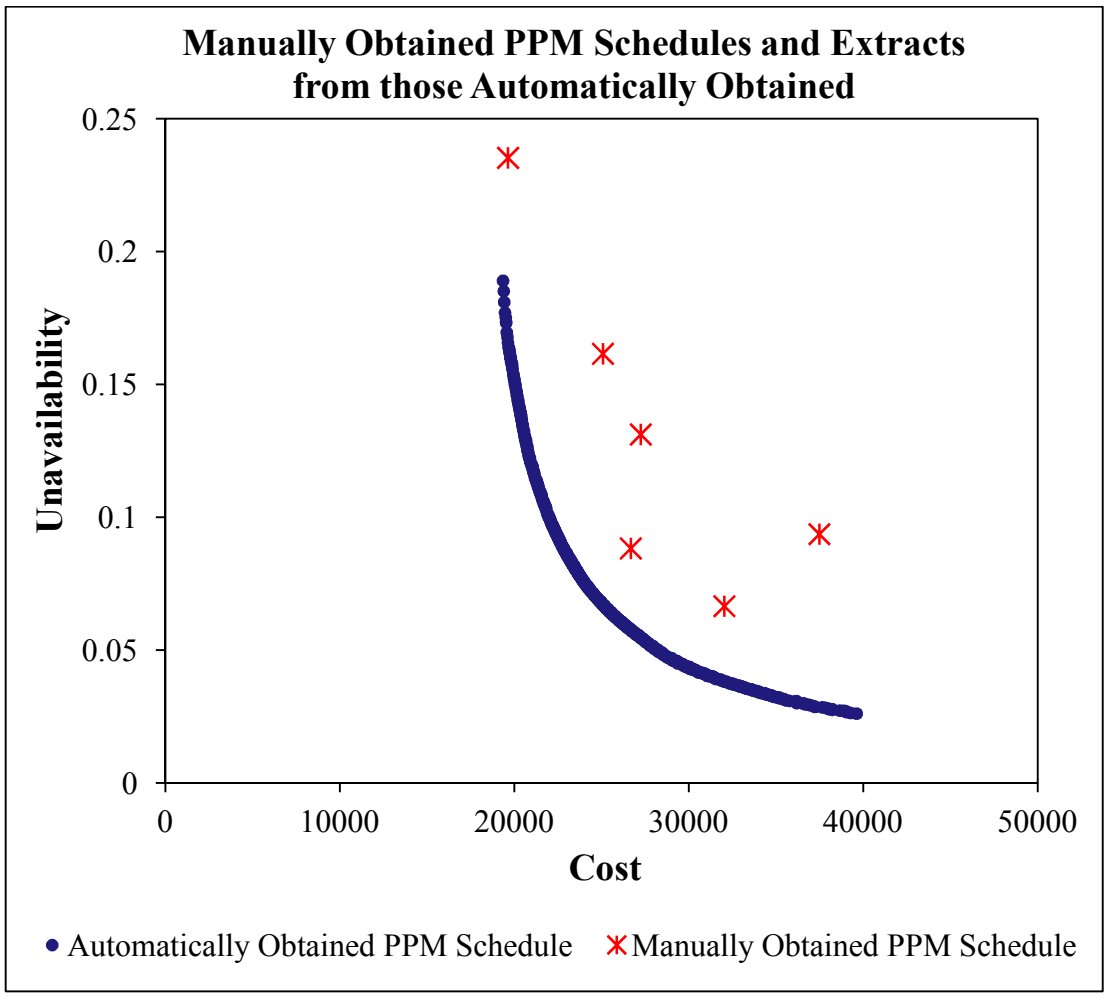

Figure 7.17 - Extracts of PPM schedules in accordance with experts' unavailability and cost value limits

To closely investigate the domination of the automatically obtained PPM schedules, it is useful to compare each of the manually obtained PPM schedules with the respective set of PPM schedules obtained automatically that dominated them. The basis for comparison is to investigate whether objective functions value (unavailability and cost) are improved for the set of automatically obtained PPM schedules (being considered) relative to those manually obtained.

M1 is dominated by 10 automatically obtained PPM schedules and its comparison is presented in Table 7.46. The comparison for M2 is presented in Table 7.47. However 305 automatically obtained PPM schedules dominated M2 and therefore Table 7.47 contains the first and last 10 of these PPM schedules. Similarly, the improvement in

objective function values of automatically obtained PPM schedules over M3 is shown in Table 7.48. M3 was dominated by 371 of automatically obtained PPM schedules and Table 7.48 also shows the first and last 10 of the 371 PPM schedules. The comparison for M4 is presented in Table 7.49. M4 was dominated by 232 automatically obtained PPM schedules and Table 7.49 shows the first and last 10 of these PPM schedules. M5 was dominated by 460 automatically obtained PPM schedules and only the first and last 10 of these are shown in Table 7.50. M6 was also dominated by 218 automatically obtained PPM schedules and its first and last 10 PPM schedules are presented in Table 7.51.

In general if K denotes domination count then domination count for each manually enumerated PPM schedule infers that there are K PPM schedules obtained auomatically which are better alternatives.

Table 7.46 - Improvements in unavailability and cost of automatically obtained PPM schedules over M1

	PPM Schedule	Cost	Cost Improvement	Unavailability	Unavailability Improvement	Generation
Manual	af(35) bp(29) cp(26) fm(17) ht(6) if(34) me(25) mt(15) st(39) vm(26)	19642		0.235366		
Automated	af(22) bp(13) cp(17) fm(18) ht(15) if(14) me(33) mt(13) st(23) vm(17)	19524	118	0.173714	0.061652	1980
	af(22) bp(13) cp(17) fm(18) ht(10) if(11) me(27) mt(13) st(23) vm(17)	19635	7	0.16584	0.069526	2004
	af(22) bp(13) cp(17) fm(18) ht(8) if(14) me(33) mt(13) st(23) vm(17)	19606	36	0.167953	0.067413	2079
	af(22) bp(20) cp(17) fm(18) ht(10) if(11) me(28) mt(13) st(23) vm(17)	19537	105	0.173265	0.062101	2182
	af(22) bp(20) cp(17) fm(18) ht(8) if(14) me(21) mt(13) st(23) vm(17)	19508	134	0.175358	0.060008	2272
	af(22) bp(20) cp(17) fm(18) ht(10) if(14) me(21) mt(13) st(23) vm(17)	19467	175	0.177111	0.058255	2330
	af(22) bp(20) cp(17) fm(18) ht(15) if(21) me(24) mt(13) st(23) vm(17)	19356	286	0.1891	0.046266	2465
	af(22) bp(20) cp(17) fm(18) ht(10) if(21) me(21) mt(13) st(23) vm(17)	19397	245	0.185182	0.050184	2484
	af(22) bp(13) cp(17) fm(18) ht(10) if(14) me(21) mt(13) st(23) vm(17)	19565	77	0.169722	0.065644	2619
	af(22) bp(20) cp(17) fm(18) ht(15) if(14) me(21) mt(13) st(23) vm(17)	19426	216	0.181068	0.054298	2943

Table 7.47 - Improvements in unavailability and cost of automatically obtained PPM schedules over M2

	PPM Schedule	Cost	Cost Improvement	Unavailability	Unavailability Improvement	Generation
Manual	af(33) bp(34) cp(2) fm(13) ht(14) if(10) me(31) mt(5) st(22) vm(6)	25083		0.161624		
Automated	af(22) bp(10) cp(17) fm(12) ht(6) if(7) me(27) mt(13) st(15) vm(17)	20378	4705	0.138254	0.02337	51
	af(9) bp(6) cp(9) fm(7) ht(3) if(4) me(30) mt(7) st(9) vm(7)	24295	788	0.0733307	0.0882933	58
	af(22) bp(13) cp(17) fm(12) ht(6) if(9) me(27) mt(13) st(15) vm(17)	20210	4873	0.144162	0.017462	59
	af(9) bp(6) cp(9) fm(7) ht(4) if(6) me(30) mt(9) st(9) vm(7)	23732	1351	0.0783102	0.0833138	62
	af(9) bp(6) cp(11) fm(7) ht(4) if(6) me(27) mt(9) st(9) vm(7)	23482	1601	0.0809094	0.0807146	63
	af(9) bp(5) cp(11) fm(9) ht(5) if(6) me(20) mt(9) st(9) vm(9)	23160	1923	0.0844502	0.0771738	64
	af(9) bp(5) cp(9) fm(6) ht(3) if(6) me(33) mt(9) st(9) vm(7)	24075	1008	0.0749177	0.0867063	65
	af(9) bp(5) cp(9) fm(6) ht(5) if(6) me(20) mt(9) st(9) vm(7)	23911	1172	0.0766995	0.0849245	67
	af(9) bp(6) cp(9) fm(7) ht(3) if(6) me(29) mt(9) st(9) vm(9)	23639	1444	0.0792277	0.0823963	67
	af(9) bp(7) cp(11) fm(7) ht(4) if(6) me(27) mt(9) st(9) vm(7)	23384	1699	0.0820685	0.0795555	70
	af(22) bp(13) cp(17) fm(18) ht(8) if(14) me(33) mt(13) st(15) vm(17)	19866	5217	0.157932	0.003692	964
	af(15) bp(8) cp(17) fm(12) ht(6) if(9) me(27) mt(13) st(15) vm(17)	20671	4412	0.128754	0.03287	965
	af(9) bp(6) cp(11) fm(7) ht(4) if(5) me(36) mt(9) st(9) vm(9)	23447	1636	0.0815856	0.0800384	1645
	af(22) bp(13) cp(17) fm(12) ht(10) if(9) me(27) mt(13) st(23) vm(17)	19868	5215	0.157602	0.004022	1947

	af(22) bp(10) cp(17) fm(18) ht(8) if(11) me(33) mt(13) st(23) vm(17)	19774	5309	0.160377	0.001247	2052
	af(22) bp(13) cp(17) fm(18) ht(6) if(9) me(21) mt(13) st(23) vm(17)	19787	5296	0.160344	0.00128	2086
	af(22) bp(13) cp(17) fm(12) ht(10) if(11) me(27) mt(13) st(23) vm(17)	19798	5285	0.159883	0.001741	2091
	af(22) bp(13) cp(17) fm(12) ht(6) if(11) me(27) mt(13) st(23) vm(17)	19880	5203	0.156638	0.004986	2100
	af(22) bp(13) cp(17) fm(12) ht(8) if(11) me(33) mt(13) st(23) vm(17)	19839	5244	0.158094	0.00353	2158
	af(22) bp(10) cp(17) fm(18) ht(6) if(11) me(27) mt(13) st(23) vm(17)	19815	5268	0.158925	0.002699	2538

Table 7.48 - Improvements in unavailability and cost of automatically obtained PPM schedules over M3

	PPM Schedule	Cost	Cost Improvement	Unavailability	Unavailability Improvement	Generation
Manual	af(14) bp(29) cp(22) fm(8) ht(23) if(10) me(29) mt(9) st(8) vm(1)	27251		0.131278		
Automated	af(9) bp(6) cp(9) fm(7) ht(3) if(4) me(30) mt(7) st(9) vm(7)	24295	2956	0.0733307	0.0579473	58
	af(9) bp(6) cp(9) fm(7) ht(4) if(6) me(30) mt(9) st(9) vm(7)	23732	3519	0.0783102	0.0529678	62
	af(9) bp(6) cp(11) fm(7) ht(4) if(6) me(27) mt(9) st(9) vm(7)	23482	3769	0.0809094	0.0503686	63
	af(9) bp(5) cp(11) fm(9) ht(5) if(6) me(20) mt(9) st(9) vm(9)	23160	4091	0.0844502	0.0468278	64
	af(9) bp(5) cp(9) fm(6) ht(3) if(6) me(33) mt(9) st(9) vm(7)	24075	3176	0.0749177	0.0563603	65
	af(9) bp(5) cp(9) fm(6) ht(5) if(6) me(20) mt(9) st(9) vm(7)	23911	3340	0.0766995	0.0545785	67
	af(9) bp(6) cp(9) fm(7) ht(3) if(6) me(29) mt(9) st(9) vm(9)	23639	3612	0.0792277	0.0520503	67
	af(9) bp(7) cp(11) fm(7) ht(4) if(6) me(27) mt(9) st(9) vm(7)	23384	3867	0.0820685	0.0492095	70
	af(9) bp(5) cp(9) fm(6) ht(3) if(4) me(36) mt(7) st(9) vm(7)	24556	2695	0.0708978	0.0603802	71
	af(9) bp(7) cp(9) fm(7) ht(3) if(6) me(29) mt(9) st(9) vm(9)	23541	3710	0.080389	0.050889	72
	af(15) bp(10) cp(17) fm(12) ht(8) if(7) me(23) mt(13) st(15) vm(17)	20602	6649	0.130421	0.000857	633
	af(15) bp(10) cp(17) fm(12) ht(6) if(7) me(33) mt(13) st(15) vm(17)	20643	6608	0.128917	0.002361	634
	af(15) bp(10) cp(17) fm(12) ht(6) if(11) me(33) mt(13) st(15) vm(11)	20678	6573	0.127906	0.003372	648
	af(9) bp(5) cp(9) fm(7) ht(5) if(5) me(36) mt(9) st(9) vm(9)	23713	3538	0.0786364	0.0526416	681
	af(11) bp(6) cp(11) fm(7) ht(5) if(5) me(35) mt(9) st(9) vm(9)	23100	4151	0.0852757	0.0460023	719
	af(9) bp(6) cp(9) fm(7) ht(3) if(5) me(29) mt(9) st(9) vm(7)	23954	3297	0.0761287	0.0551493	767

	af(15) bp(8) cp(17) fm(12) ht(6) if(7) me(23) mt(13) st(15) vm(17)	20741	6510	0.12659	0.004688	889
	af(11) bp(6) cp(11) fm(7) ht(5) if(5) me(33) mt(9) st(9) vm(7)	23275	3976	0.0834004	0.0478776	927
	af(15) bp(8) cp(17) fm(12) ht(6) if(9) me(27) mt(13) st(15) vm(17)	20671	6580	0.128754	0.002524	965
	af(9) bp(6) cp(11) fm(7) ht(4) if(5) me(36) mt(9) st(9) vm(9)	23447	3804	0.0815856	0.0496924	1645

Table 7.49 - Improvements in unavailability and cost of automatically obtained PPM schedules over M4

	PPM Schedule	Cost	Cost Improvement	Unavailability	Unavailability Improvement	Generation
Manual	af(3) bp(7) cp(11) fm(7) ht(24) if(6) me(25) mt(8) st(5) vm(11)	26679		0.088344		
Automated	af(9) bp(6) cp(9) fm(7) ht(3) if(4) me(30) mt(7) st(9) vm(7)	24295	2384	0.0733307	0.0150133	58
	af(9) bp(6) cp(9) fm(7) ht(4) if(6) me(30) mt(9) st(9) vm(7)	23732	2947	0.0783102	0.0100338	62
	af(9) bp(6) cp(11) fm(7) ht(4) if(6) me(27) mt(9) st(9) vm(7)	23482	3197	0.0809094	0.0074346	63
	af(9) bp(5) cp(11) fm(9) ht(5) if(6) me(20) mt(9) st(9) vm(9)	23160	3519	0.0844502	0.0038938	64
	af(9) bp(5) cp(9) fm(6) ht(3) if(6) me(33) mt(9) st(9) vm(7)	24075	2604	0.0749177	0.0134263	65
	af(9) bp(5) cp(9) fm(6) ht(5) if(6) me(20) mt(9) st(9) vm(7)	23911	2768	0.0766995	0.0116445	67
	af(9) bp(6) cp(9) fm(7) ht(3) if(6) me(29) mt(9) st(9) vm(9)	23639	3040	0.0792277	0.0091163	67
	af(9) bp(7) cp(11) fm(7) ht(4) if(6) me(27) mt(9) st(9) vm(7)	23384	3295	0.0820685	0.0062755	70
	af(9) bp(5) cp(9) fm(6) ht(3) if(4) me(36) mt(7) st(9) vm(7)	24556	2123	0.0708978	0.0174462	71
	af(9) bp(7) cp(9) fm(7) ht(3) if(6) me(29) mt(9) st(9) vm(9)	23541	3138	0.080389	0.007955	72
	af(9) bp(5) cp(9) fm(7) ht(4) if(5) me(36) mt(9) st(9) vm(7)	23970	2709	0.0759384	0.0124056	554
	af(11) bp(5) cp(11) fm(7) ht(5) if(5) me(33) mt(9) st(9) vm(7)	23373	3306	0.0822466	0.0060974	559
	af(11) bp(7) cp(11) fm(7) ht(5) if(5) me(33) mt(9) st(9) vm(9)	23002	3677	0.0864293	0.0019147	573
	af(9) bp(6) cp(11) fm(7) ht(4) if(5) me(30) mt(9) st(9) vm(7)	23622	3057	0.0797027	0.0086413	578
	af(9) bp(6) cp(9) fm(7) ht(4) if(5) me(36) mt(9) st(9) vm(7)	23872	2807	0.0771001	0.0112439	617
	af(9) bp(5) cp(9) fm(7) ht(5) if(5) me(36) mt(9) st(9) vm(9)	23713	2966	0.0786364	0.0097076	681
	af(11) bp(6) cp(11) fm(7) ht(5) if(5) me(35) mt(9) st(9) vm(9)	23100	3579	0.0852757	0.0030683	719
	af(9) bp(6) cp(9) fm(7) ht(3) if(5) me(29) mt(9) st(9) vm(7)	23954	2725	0.0761287	0.0122153	767
	af(11) bp(6) cp(11) fm(7) ht(5) if(5) me(33) mt(9) st(9) vm(7)	23275	3404	0.0834004	0.0049436	927
	af(9) bp(6) cp(11) fm(7) ht(4) if(5) me(36) mt(9) st(9) vm(9)	23447	3232	0.0815856	0.0067584	1645

Table 7.50 - Improvements in unavailability and cost of automatically obtained PPM schedules over M5

	PPM Schedule	Cost	Cost Improvement	Unavailability	Unavailability Improvement	Generation
Manual	af(2) bp(8) cp(2) fm(1) ht(2) if(24) me(7) mt(14) st(15) vm(13)	37487		0.093716		
Automated	af(9) bp(6) cp(9) fm(7) ht(3) if(4) me(30) mt(7) st(9) vm(7)	24295	13192	0.0733307	0.0203853	58
	af(9) bp(6) cp(9) fm(7) ht(4) if(6) me(30) mt(9) st(9) vm(7)	23732	13755	0.0783102	0.0154058	62
	af(9) bp(6) cp(11) fm(7) ht(4) if(6) me(27) mt(9) st(9) vm(7)	23482	14005	0.0809094	0.0128066	63
	af(9) bp(5) cp(11) fm(9) ht(5) if(6) me(20) mt(9) st(9) vm(9)	23160	14327	0.0844502	0.0092658	64
	af(9) bp(5) cp(9) fm(6) ht(3) if(6) me(33) mt(9) st(9) vm(7)	24075	13412	0.0749177	0.0187983	65
	af(9) bp(5) cp(9) fm(6) ht(5) if(6) me(20) mt(9) st(9) vm(7)	23911	13576	0.0766995	0.0170165	67
	af(9) bp(6) cp(9) fm(7) ht(3) if(6) me(29) mt(9) st(9) vm(9)	23639	13848	0.0792277	0.0144883	67
	af(9) bp(7) cp(11) fm(7) ht(4) if(6) me(27) mt(9) st(9) vm(7)	23384	14103	0.0820685	0.0116475	70
	af(9) bp(5) cp(9) fm(6) ht(3) if(4) me(36) mt(7) st(9) vm(7)	24556	12931	0.0708978	0.0228182	71
	af(9) bp(7) cp(9) fm(7) ht(3) if(6) me(29) mt(9) st(9) vm(9)	23541	13946	0.080389	0.013327	72
	af(4) bp(3) cp(4) fm(3) ht(2) if(2) me(33) mt(3) st(4) vm(3)	33644	3843	0.0350592	0.0586568	1261
	af(4) bp(3) cp(4) fm(3) ht(2) if(2) me(33) mt(3) st(5) vm(3)	32864	4623	0.0366173	0.0570987	1261
	af(4) bp(3) cp(3) fm(3) ht(1) if(2) me(36) mt(5) st(4) vm(3)	33914	3573	0.0345882	0.0591278	1282
	af(4) bp(3) cp(5) fm(3) ht(1) if(2) me(33) mt(3) st(5) vm(3)	32938	4549	0.0366096	0.0571064	1295
	af(3) bp(3) cp(3) fm(3) ht(1) if(2) me(36) mt(4) st(3) vm(3)	36156	1331	0.0308778	0.0628382	1333
	af(4) bp(2) cp(4) fm(3) ht(2) if(2) me(33) mt(3) st(5) vm(3)	33550	3937	0.0352946	0.0584214	1364
	af(4) bp(2) cp(3) fm(2) ht(1) if(2) me(36) mt(3) st(3) vm(3)	37162	325	0.0289667	0.0647493	1432
	af(4) bp(2) cp(3) fm(2) ht(2) if(2) me(33) mt(3) st(3) vm(3)	36588	899	0.029945	0.063771	1477
	af(3) bp(2) cp(4) fm(3) ht(2) if(2) me(35) mt(3) st(3) vm(3)	36170	1317	0.0308119	0.0629041	1496
	af(9) bp(6) cp(11) fm(7) ht(4) if(5) me(36) mt(9) st(9) vm(9)	23447	14040	0.0815856	0.0121304	1645

Table 7.51 - Improvements in unavailability and cost of automatically obtained PPM schedules over M6

	PPM Schedule	Cost	Cost Improvement	Unavailability	Unavailability Improvement	Generation
Manual	af(5) bp(11) cp(7) fm(3) ht(7) if(1) me(10) mt(3) st(4) vm(16)	32042		0.066594		
Automated	af(9) bp(4) cp(7) fm(6) ht(3) if(4) me(29) mt(7) st(8) vm(7)	25262	6780	0.065839	0.0007545	123
	af(9) bp(4) cp(7) fm(6) ht(3) if(4) me(36) mt(5) st(8) vm(7)	25463	6579	0.0645153	0.0020782	129
	af(9) bp(4) cp(7) fm(6) ht(3) if(4) me(33) mt(5) st(9) vm(7)	25203	6839	0.0664292	0.0001643	133
	af(9) bp(4) cp(7) fm(5) ht(3) if(4) me(36) mt(5) st(8) vm(5)	25976	6066	0.061487	0.0051065	134
	af(9) bp(4) cp(7) fm(5) ht(3) if(4) me(36) mt(5) st(8) vm(7)	25626	6416	0.0635045	0.003089	135
	af(9) bp(5) cp(7) fm(5) ht(3) if(4) me(33) mt(5) st(8) vm(7)	25430	6612	0.0647529	0.0018406	136
	af(9) bp(5) cp(7) fm(6) ht(3) if(4) me(36) mt(5) st(8) vm(7)	25267	6775	0.0657624	0.0008311	138
	af(9) bp(4) cp(7) fm(5) ht(3) if(4) me(36) mt(5) st(6) vm(5)	26496	5546	0.0586973	0.0078962	145
	af(9) bp(5) cp(7) fm(5) ht(3) if(4) me(36) mt(5) st(8) vm(5)	25780	6262	0.0627381	0.0038554	147
	af(9) bp(4) cp(7) fm(5) ht(3) if(4) me(20) mt(5) st(6) vm(7)	26146	5896	0.0607208	0.0058727	151
	af(4) bp(3) cp(5) fm(4) ht(2) if(3) me(36) mt(3) st(5) vm(3)	31385	657	0.0400735	0.02652	1033
	af(4) bp(3) cp(5) fm(4) ht(2) if(3) me(33) mt(4) st(5) vm(3)	30983	1059	0.0409025	0.025691	1038
	af(5) bp(3) cp(5) fm(4) ht(2) if(2) me(36) mt(4) st(5) vm(3)	30943	1099	0.0409332	0.0256603	1042
	af(4) bp(3) cp(5) fm(3) ht(2) if(3) me(33) mt(4) st(5) vm(3)	31472	570	0.0397643	0.0268292	1047
	af(4) bp(3) cp(5) fm(3) ht(2) if(2) me(33) mt(4) st(5) vm(3)	31962	80	0.0384114	0.0281821	1065
	af(5) bp(3) cp(5) fm(4) ht(2) if(2) me(36) mt(5) st(4) vm(3)	31321	721	0.0401656	0.0264279	1074
	af(5) bp(3) cp(5) fm(3) ht(2) if(2) me(33) mt(4) st(5) vm(3)	31432	610	0.0397951	0.0267984	1075
	af(5) bp(3) cp(5) fm(4) ht(2) if(3) me(33) mt(4) st(5) vm(3)	30453	1589	0.0422826	0.0243109	1079
	af(5) bp(3) cp(5) fm(3) ht(2) if(3) me(33) mt(4) st(5) vm(3)	30942	1100	0.0411461	0.0254474	1090
	af(5) bp(3) cp(5) fm(3) ht(2) if(2) me(33) mt(3) st(5) vm(3)	31834	208	0.0389651	0.0276284	1179

In general, it can be observed in Table 7.46, Table 7.47, Table 7.48, Table 7.49, Table 7.50 and Table 7.51 that unavailability and cost were improved in all the automatically obtained PPM schedules relative to the manually obtained PPM schedules which they dominated. Obtaining better approximations to the Pareto front will require an exhaustive exploration of the PPM feasible region. In contrast manual process is inefficient in dealing with such, especially when the size of the feasible region is huge; for instance the ones used in the thesis. With automated optimisation, a large number of optimal or near optimal PPM schedules can be obtained within short period of time. It also provides the system engineer with wide range of optimal or near optimal design options. For instance the 684 optimal PPM schedules obtained by automated optimisation as opposed to the 6 enumerated through manual process. The comparison therefore suggests that automated optimisation produces better approximations to the Pareto front than manual process.

7.6 Revisiting the Research Hypothesis

The research hypothesis put forward and tested in this thesis is that "the optimisation of complex safety critical systems with respect to availability and cost taking into account the dynamic effects of scheduled preventive maintenance is both feasible and beneficial and can be achieved through a novel integration of state-of-the-art model based safety analysis technique with recent work on meta-heuristics."

In this section, the hypothesis is revisited and checked against the work done so far in the thesis. In order to evaluate the work against the hypothesis, it is necessary to evaluate the research objectives which were set out in the introduction chapter. These evaluations are discussed next.

7.6.1 Objective I - Understand earlier work and understand limitations, e.g. restrictions on system modelling, and modelling of dependability and cost attributes

Chapters 2 and 3 were dedicated to investigating existing work on maintenance and optimisation respectively of a system. Existing safety analysis techniques were

investigated while several emerging ones were highlighted. Maintenance was also identified as a factor in improving system dependability. As a result, advances in maintenance engineering were investigated where the limitations on the use of RBDs in evaluating system dependability was discussed. Existing gaps in optimising preventive maintenance were also identified.

7.6.2 Objective II - Investigate modelling the effects of preventative maintenance

In chapter 4, the dynamic effect of preventive maintenance on the component of a system was modeled based on the *age reduction model*, where perfect and imperfect maintenance were both considered. The results of this investigation were the establishment of generic evaluation models that calculate the effect of maintenance on component reliability, availability, and consequently unavailability. It was shown that the effect of maintenance on the reliability and availability of a system can be calculated using those component models in the Esary-Proschan calculation that can be applied on cut sets of system fault trees. The whole process was automated within HiP-HOPS enabling fully automated evaluation of the effects of maintenance on a system under PPM and IPM policies.

7.6.3 Objective III - Investigate cost modelling

Chapter 4 also models the effect of perfect and imperfect preventive maintenance on the cost of a component and consequently a system.

7.5.4 Objective IV - Investigate application of heuristics on systems optimisation, especially application of genetic algorithm

The application of genetic algorithms on system optimisation was investigated in chapter 3 where various algorithms were discussed, and the advantages of using the non-dominated sorting genetic algorithm (NSGA) II were highlighted.

7.6.5 Objective V - Define and model the optimisation problem

In chapter 6, the encoding of the preventive maintenance scheduling optimisation problem was established. Constraints on the optimisation were also established and the objective functions were defined in terms of the unavailability and cost models derived in this thesis for calculating the effects of maintenance on a system.

7.6.6 Objective VI - Design and implement an appropriate optimisation algorithm

Chapter 6 also contains an established variant of the NSGA II that has been designed precisely for solving the optimisation problems as it was defined in *objective v* above.

7.6.7 Objective VII - Evaluate the approach via application on a case study

The approach to optimisation of maintenance developed in this thesis was thoroughly evaluated on the case studies and the results of various runs of the optimisation with different formulation of constraints imposed were discussed in detail. Overall it was shown that the approach is feasible, it works and can yield useful results that can feed the difficult and presently manual engineering process of optimising maintenance schedules. The experiments give evidence that the algorithms are fast and can produce good solutions within a short period of time, even in cases when the solution space is large. The combination of reliability modelling under assumptions of maintenance, HiP-HOPS, and GAs has been shown to be effective.

7.7 Chapter Summary

This chapter evaluated the approach to optimisation developed in this thesis. In the light of the results of this evaluation, it was shown that the research hypothesis was thorough, tested and that the objectives of this work were met. Additionally, a comparison between manual and automated PM optimisation was presented. The comparison showed that automated optimisation is more efficient.

8 CONCLUSIONS

In the introduction chapter of this thesis, the following hypothesis was postulated:

> "The optimisation of complex safety critical systems with respect to availability and cost taking into account the dynamic effects of scheduled preventive maintenance is both feasible and beneficial and can be achieved through a novel integration of state-of-the-art model based safety analysis technique with recent work on meta-heuristics,"

where the work was mainly aimed at:

(i) investigating the effects of periodic maintenance policies on the design of safety critical systems,
(ii) and establishing and demonstrating the scheduling optimisation of such policies through automation.

To achieve the aims of the research, the following objectives were set out to logically progress with the work and to enable testing of the stated hypothesis:

(i) understand earlier work and understand limitations, e.g. restrictions on system modelling, and modelling of dependability and cost attributes,
(ii) investigate modelling the effects of preventative maintenance,
(iii) investigate cost modelling,
(iv) investigate application of heuristics on systems optimisation, especially application of genetic algorithm,
(v) define and model the optimisation problem,
(vi) design and implement an appropriate optimisation algorithm,
(vii) and finally to evaluate the approach via application on case study.

The aims of the research were investigated successively and thus, in achieving the first one, two types of periodic maintenance policies were considered; *perfect* and *imperfect*

preventive maintenance. For both policies, the *proportional age reduction* model was used in the mathematical modelling for the following.

- component reliability
- component availability and subsequently unavailability
- component and system cost

It was assumed that components failure characteristics follow the *Weibull* distribution and hence the use of such distribution in the mathematical modelling. The calculation for system reliability and unavailability was based on their respective component evaluations and the Esary-Proschan calculation was used.

To achieve the second aim of the research, a genetic algorithm was used in defining the search problem, where also a variant of the selection technique "non-dominated sorting genetic algorithm (NSGA) II" was defined. To advance the search towards population of improved system designs and also against set system requirements, two categories of constraints were established; *primary* and *secondary*. The primary constraints are those that are fundamental to the optimisation, whereas the secondary are those that are optional depending on the system design requirements. A combination of both primary and secondary constraints has been termed as *composite* constraint. Having established these required parameters, the optimisation problem for perfect and imperfect preventive maintenance scheduling with respect to system unavailability and cost was defined.

The established mathematical models for reliability, unavailability and cost were implemented in a well established semi-automated dependability analysis tool HiP-HOPS. An algorithm for the formulated preventive maintenance scheduling optimisation problem was also implemented in HiP-HOPS to extend its existing support for optimisation. The preventive maintenance scheduling optimisation problem was evaluated on a model of the *fuel oil service system* (as a case study) that supplies the main engine of a ship.

The evaluation of the defined preventive maintenance scheduling optimisation problem via the case study met research expectations, and optimal (or near optimal) preventive maintenance schedules which represent trade-offs between unavailability and cost were obtained under both perfect and imperfect preventive maintenance policies.

To further demonstrate the scalability of the defined preventive maintenance scheduling optimisation problem, the approach was also evaluated on an *aircraft wheel brake system* as a second case study. This evaluation was based on the composite constraint under perfect preventive maintenance, where results obtained were optimal preventive maintenance schedules under such policy.

Finally, the evaluation was performed against the set out objectives of the research. It showed that each objective has been met and thus research aims achieved which ultimately validates the postulated hypothesis.

8.1 Research Contributions

The key contribution of this thesis is the establishment of a novel, automated method for optimisation of dependability and cost of a system that performs optimisation of both maintenance schedules and selection of alternative implementations of components. Though there is some earlier work both on optimisation of maintenance and architecture optimisation, both the formulation of the problem as addressed in the thesis and its solution via a combination of automated dependability analysis techniques with genetic algorithms is novel. Within this context the thesis makes a number of smaller contributions:

- Establishes mathematical models using Weibull distribution for:
  - (i) component reliability under imperfect preventive maintenance
  - (ii) component availability and subsequently unavailability for both perfect and imperfect preventive maintenance
  - (iii) component and system basic cost for perfect and imperfect preventive maintenance

- Establishes the following constraints for scheduled preventive maintenance optimisation:
  (i) Primary constraints – to ensure that preventive maintenance scheduling is not performed too early or too late
  (ii) Secondary constraints – constituting of two parts; expert judgement on component preventive maintenance time and system architectural modification through component substitution
  (iii) Composite constraint – comprising of both primary and secondary constraints. This constraint allows for flexibility; in that the system engineer can apply all the constraints in a single instance of the optimisation
- Establishes an algorithm for the preventive maintenance scheduling optimisation problem which is a variant of the non-dominated sorting genetic algorithm (NSGA) II.
- Establishes the calculation for the number of potential PM individuals in a PM solution space **X** and the number of feasible PM individuals in a feasible PM region $f\mathbf{X}$ both under primary and expert judgement constraint.
- Extends the use of a state-of-the-art dependability analysis technique (HiP-HOPS) in the sphere of maintainability, with improvements that enable analysis and optimisation of maintenance.

As a result of the above novel contributions, dependability analysis in HiP-HOPS is extended with new capabilities for objective functions evaluation and optimisation under assumptions of preventive maintenance.

8.2 Thesis Limitations

Although, this work has made significant contributions in automated dependability analysis of systems, there are areas of inadequacy. These areas are as highlighted below.

- *Shortest PM Interval* - The procedure through which the shortest PM interval T for all components is evaluated is informal and imprecise. It is only based on a

value that is set less than the MTTF (in the case of PPM) or MTBF (in the case of IPM) of the component that fails most often in the system.

- *Constant Improvement Factor* - The assumption of constant improvement factor *f* under IPM implies that improvement of component condition is independent of its level of wear or damage.
- *Homogeneity of PM Policy* - This work assumes a homogenous type of PM policy. This means that all components identified for PM undergo the same PM policy (i.e. PPM or IPM).
- *Computational Cost* - For a large system model, great amount of time may be required in synthesising the cut sets and also in the evaluation of objective functions.

8.3 Suggestion for Further Work

The following are identified areas for which the optimisation of preventive maintenance scheduling as investigated in this work can be extended.

- To incorporate replacement policy in the scheduled preventive maintenance; it is essential to replace a component when it approaches or is just beyond its useful life. This will improve system availability and also prevent unnecessary cost that may be incurred due to unplanned maintenance.

- To establish an absolute evaluation model or algorithm for the calculation of the number of potential PM individuals in PM solution space $\mathbf{X}$ and feasible PM individuals in feasible PM region $f\mathbf{X}$ under component substitution.

- To establish a strategy for which components can be grouped for scheduled preventive maintenance. This will serve as a cost saving measure and it will be interesting to also investigate the effect of the grouping on system reliability and unavailability.

- To develop an automated system for predictive maintenance based on online monitoring of the conditions of components. An optimal PM individual obtained from the scheduled PM optimisation from HiP-HOPS could serve as an input to such system. This will also help reduce the occurrence of unplanned maintenance in-between PM times, thereby improving availability and saving cost.

REFERENCES

Andrews, J D and Moss, T R, 1993, *Reliability and Risk Assessment*, John Wiley & Sons, New York

Artana, K B and Ishida, K, 2002, *Spreadsheet Modelling of Optimal Schedule for Components in Wear-out-Phase*, Reliability Engineering and System Safety, (77)1, pp81-91

Ashlock, D, 2005, *Evolutionary Computation for Modeling and Optimization*, New York, Springer

Birolini, A, 2007, *Reliability Engineering: Theory and Practice*, 5th Ed, Berlin, Springer

Borges, C C H and Barbosa, H J C, 2000, *A Non-Generational Genetic Algorithm for Multiobjective Optimization*, Proceedings of 2000 Congress on Evolutionary Computation, San Diego, California Vol.1, pp172-179.

Bris, R, Châtelet, E and Yalaoui, F, 2003, *New Method to Minimize the Preventive Maintenance Cost of Series-Parallel Systems*, Reliability Engineering and System Safety, (82)3, pp247-255

Bukowski, J V and Goble, W M, 1995, *Using Markov Models for Safety Analysis of Programmable Electronic Systems*, ISA Transactions, 34(2), pp193-198

Castro, I T, 2009, *A Model of Imperfect Preventive Maintenance with Dependant Failure Modes*, European Journal of Operational Research, (196)1, pp217-224

Chambers, L, 2006, *A Hazard Analysis of Human Factors in Safety-Critical Systems Engineering*, Proceedings of the 10th Australian Workshop on Safety Critical Systems and Software, Vol. 55, pp27-41, Sydney, Australia

Coit, D W and Smith, A E, 1996, *Reliability Optimization of Series-Parallel System Using a Genetic Algorithm*, IEEE Transactions on Reliability, (45) 2, pp254-266

Corne, D W, Knowles, J D, and Oates, M J, 2000, *The Pareto Envelope-Based Selection Algorithm for Multiobjective Optimization*, Proceedings of the sixth International Conference on Parallel Problem Solving from Nature VI (PPSN VI), pp839-848, Berlin, 2000

Corne, D W, Jerram, R N, Knowles, J D and Oates, M J, 2001, *PESA II: Region Based Selection in Evolutionary Multiobjective Optimization*, Proceedings of the Genetic and Evolutionary Computation Conference (GECCO'2001), pp283-290, San Francisco, California, 2001

Correia, V M F, Soares, C M M and Soares, C A M, 2001, *Refined Models for the Optimal Design of Adaptive Structures using Simulated Annealing*, Composite Structures, 54(2-3), pp161-167

Deb, K, Agarwal, S, Pratab, A and Meyarivan, T, 2000, *A Fast Elitist Non-dominated Sorting Genetic Algorithm for Multi-objective Optimization: NSGA-II*, Proceedings of the sixth International Conference on Parallel Problem Solving from Nature VI (PPSN VI), Vol. 1917/2000, pp849-858, Berlin, 2000

Deb, K, Pratab, A, Agarwal, S and Meyarivan, T, 2002, *A Fast and Elitist Multiobjective Genetic Algorithm: NSGA-II, IEEE Transactions on Evolutionary Computation*, (6)2, pp182-197

Diller, A, 1999, *Z an Introduction to Formal Methods*, 2nd Ed, New York, John Wiley & Sons

Ebeling, C E, 1997, *An Introduction to Reliability and Maintainability Engineering*, USA, McGraw-Hill

Erickson, M, Mayer, A and Horn, J, 2002, *Multi-objective Optimal Design of Groundwater Remediation Systems: Application of the Niched Pareto Genetic Algorithm (NPGA)*, Advances in Water Resources, 25(1), pp51-65

Fard, N S, 1997, Determination of Minimal Cut Sets of a Complex Fault Tree, Computers & Industrial Engineering, (33)1–2, pp59-62

Favuzza, S, Ippolito, M G and Sanseverino, E R, 2006, *Crowded Comparison Operators for Constraints Handling in NSGA-II for Optimal Design of the Compensation System in Electrical Distribution Networks*, Advanced Engineering Informatics, (20)2, pp201-211

Fleischer, M, 1995, *Simulated Annealing: Past, Present, and Future*, Proceedings of the 27th Conference on Winter Simulation, pp155-161, Arlington, USA, December 03-06, 1997

Gen, M and Cheng, R, 1997, Genetic Algorithms and Engineering Design, New York, John Wiley & Sons

Goto, S, Onizuka, Y and Nakamura, M, 2006, *Optimal Maintenance Scheduling by Average System Availability Through Appropriate Segmentation of Maintenance Intervals*, International Journal of Innovative Computing, Information and Control, (2)5, pp1083-1096

Guo, H and Yang, X, 2007, *A Simple Reliability Block Diagram Method for Safety Integrity Verification*, Reliability Engineering and System Safety, (92)9, p1267-1273

Guo, H and Yang, X, 2008, *Automatic Creation of Markov Models for Reliability Assessment of Safety Instrumented Systems*, Reliability Engineering and System Safety, 93(6), pp829-837

Galloway, A., McDermid, J., Murdoch, J. and Pumfrey, D., 2002, Automation of System Safety Analysis: Possibilities and Pitfalls, In Proceedings of the 20th International System Safety Conference (ISSC), System Safety Society

Hart, E, 2006, *Non-Deterministic Search Techniques* [online], Available: http://www.dcs.napier.ac.uk/~benp/ecresources/Search2006.pdf [Accessed 06 June 2009]

Herring, M S, Owens, B D, Leveson, N, Ingham, M and Weiss, K A, 2007, *A Safety-Driven, Model-Based System Engineering Methodology*, Part I, MIT Technical Report, December 2007

Huang, H, Tian, Z and Zuo, M J, 2005, *Intelligent Interactive Multiobjective Optimization Method and its Application to Reliability Optimization*, IIE Transactions, (37)11, pp983-993

Jin, T and Coit, D W, 2003, *Approximating Network Reliability Estimates Using Linear and Quadratic Unreliability of Minimal Cuts*, Reliability Engineering and System Safety, (82)1, pp41-48

Konak, A, Coit, D W and Smith, A E, 2006, *Multi-Objective Optimization Using Genetic Algorithms: A Tutorial*, Reliability Engineering and System Safety, (91)9, p992-1007

Kunkle, D, 2005, *A Summary and Comparison of MOEA Algorithms* [online], Available: http://www.ccs.neu.edu/home/kunkle/papers/techreports/moeaComparison.pdf [Accessed 07 August 2010]

Leveson N, 1997, *A Demonstration Safety Analysis of Air Traffic Control Software* [online], MIT, Available: http://sunnyday.mit.edu/papers/dfw2.ps [Accessed 21 February 2008]

Leveson, N, 2003, *A New Approach to Hazard Analysis for Complex Systems*, International Conference of the System Safety Society, Ottawa, August 2003

Leveson, N G, 2002, *System Safety Engineering: Back to the Future*, Aeronautics and Astronautics, Massachusetts Institute of Technology

Leveson, N G and Dulac, N, 2005, *Safety and Risk-Driven Design in Complex Systems-of-Systems*, 1st NASA/AIAA Space Exploration Conference, February 2005, Orlando

Lust, T, Roux, O and Riane, F, 2009, *Exact and Heuristic Methods for the Selective Maintenance Problem*, European Journal of Operational Research, (197)3, pp1166-1177

MIL-HDBK-338B, 1998, *Military Handbook: Electronic Reliability Design Handbook* [online], Accessed 21 August 2009], Available: http://www.sre.org/pubs/Mil-Hdbk-338B.pdf

MacFarlane, A and Tuson, A, 2009, *Local Search: A Guide for the Information Retrieval Practitioner*, Information Processing and Management, (2009)45, pp159-174

Márquez, A C, 2007, *The Maintenance Management Framework: Models and Methods for Complex Systems Maintenance*, Springer Series in Reliability Engineering, London: Springer

Marseguerra, M, Zio, E and Martorell, S, 2006, *Basics of Genetic Algorithms Optimization for RAMS Applications*, Reliability Engineering and System Safety, (91)9, pp977-991

Martorell, S, Sanchez, A and Serradell V, 1999, *Age-Dependent Reliability Model Considering Effects of Maintenance and Working Conditions*, Reliability Engineering and System Safety, (64)1, pp19-31

Muramatsu, D, 2008, *Online Signature Verification Algorithm using Hill-Climbing Method*, IEEE/IFIP International Conference on Embedded and Ubiquitous Computing, euc, Vol. 2, pp133-138

Netjasov, F and Janic, M, 2008, *A Review of Research on Risk and Safety Modelling in Civil Aviation*, Journal of Air Transport Management, (14)4, pp213-220

Nggada, S H, Papadopoulos, Y I and Parker, D J, 2010a, *Extending HiP-HOPS with Capabilities of Planning Preventative Maintenance*, 6th Annual International Conference on Computer Science and Information Systems, 25-28 June 2010, Athens, Greece

Nggada, S H, Parker, D J and Papadopoulos, Y I, 2010b, *Dynamic Effect of Perfect Preventive Maintenance on System Reliability and Cost Using HiP-HOPS*, IFAC-MCPL 2010, 5th Conference on Management and Control of Production and Logistics, 8 - 10 September 2010, Coimbra – Portugal, IFAC Proceedings Volumes (IFAC-PapersOnline), pp204 - 209, ISSN: 14746670

Nggada, S. H., Papadopoulos, Y. I. and Parker, D. J., 2010c, *Extending HiP-HOPS with Capabilities of Planning Preventative Maintenance*, Strategic Advantage of Computing Information Systems in Enterprise Management, (ed) Majid Sarrafzadeh, chapter 17 in volume containing revised selected papers from the International Conference in Computer Systems and Information Systems in 2009 & 2010, pp231-242, ISBN: 978-960-6672-93-4

Painton, L A and Campbell, J E, 1995, *Genetic Algorithms in Optimization of System Reliability*, IEEE Transactions on Reliability, (44) 2, pp 172-178

Papadopoulos, Y I and McDermid, J A, 1999, *Hierarchically performed hazard origin and propagation studies,* In: 18th International Conference in Computer Safety, Reliability and Security, Toulouse, France, pp139-152

Papadopoulos, Y, Parker, D and Grante, C, 2004, *A Method and Tool Support for Model-Based Semi-Automated Failure Modes and Effects Analysis of Engineering Designs*, Proceedings of the 9th Australian Workshop on Safety Critical Systems and Software, Vol. 47, pp89-95, Brisbane, Australia

Papadopoulos, Y, Walker, M, Hamann, R, Uhlig, A, Rude, E, Gratz, U and Lien, R, 2008, *Semi Automatic Failure Analysis Based on Simulation Models*, Proceedings of the ASME 27th International Conference on Offshore Mechanics and Artic Engineering, 15-20 June 2008, Estoril, Portugal

Parker, D J, 2010, *Multi-Objective Optimisation of Safety-Critical Hierarchical Systems*, PhD Thesis, University of Hull

Parker, D J and Papadopoulos, Y I, 2007, *Effective Multicreteria Redundancy Allocation via Model-Based Safety Analysis*, IFAC

Pham, H and Wang, H, 1996, *Imperfect Maintenance*, European Journal Operational Research, (94)3, pp425-438

Rouvroye, J L and van den Bliek , E G, 2002, *Comparing Safety Analysis Techniques, Reliability Engineering and System Safety*, 75(3), pp289-294

Sanchez, A, Carlos, S, Martorell, S and Villanueva, J F, 2009, *Addressing Imperfect Maintenance Modelling Uncertainty in Unavailability and Cost Based Optimization*, Reliability Engineering and System Safety, (94)1, pp23-32

Sharvia, S, 2010, *Integrated Application of Compositional and Behavioural Safety Analysis*, PhD Thesis, University of Hull

Sheu, S, Lin, Y and Liao, G, 2006, *Optimum Policies for a System with General Imperfect Maintenance*, Reliability Engineering and System Safety, 91(3), pp362-369

Storey, N, 1996, *Safety-Critical Computer Systems*, London, Addison Wesley Longman

Suman, B, 2004, *Study of Simulated Annealing Based Algorithms for Multiobjective Optimization of a Constrained Problem*, Computers and Chemical Engineering, 28(9), pp1849-1871

Suman, B, Hoda, N and Jha, S, 2009, *Orthogonal Simulated Annealing for Multiobjective Optimization*, Computers and Chemical Engineering, doi:10.1016/j.compchemeng.2009.11.015

Thimbleby, H, Cairns, P and Jones, M, 2001, *Usability Analysis with Markov Models*, ACM Transactions on Computer-Human Interaction, (8)2, pp99-132

Tsai, Y, Wang, K and Teng, H, 2001, *Optimizing Preventive Maintenance for Mechanical Components using Genetic Algorithms*, Reliability Engineering and System Safety, (74)1, pp89-97

Tsai, Y, Wang, K and Tsai, L, 2004, *A Study of Availability-Centered Preventive Maintenance for Multi-Component Systems*, Reliability Engineering and System Safety, (84)3, pp261-270

van Dijkhuizen, G and van der Heijden, M, 1999, *Preventive Maintenance and the Interval Availability Distribution of an Unreliable Production System*, Reliability Engineering and System Safety, (66)1, pp13-27

Wang, K S, Chang, W H, Tsai, Y T and Hsu, F S, 1996, *Using Genetic Algorithm Planning Preventive Replacement of Components in a System*, Proceedings of the 13th National Conference of the Chinese Society of Mechanical Engineers, Taiwan: National Taiwan University, pp271-278

Wei-zhong, A and Xi-Gang, Y, 2009, A *Simulated Annealing-Based Approach to the Optimal Synthesis of Heat-Integrated Distillation Sequences*, Computers & Chemical Engineering, 33(1), pp199-212

Weise, T, 2008, *Global Optimization Algorithms - Theory and Application* [online], Available: http://*www.it-weise.de/projects/book.pdf* [Accessed February 21 2008], Thomas Weise

Wolforth, I, Walker, M, Papadopoulos, Y and Grunske, L, 2010, *Capture and Reuse of Composable Failure Patterns*, International Journal of Critical Computer-Based Systems, (1)1-3, pp128-147

Xi, B, Liu, Z, Raghavachari, M, Xia, C H and Zhang, L, 2004, *A Smart hill-Climbing Algorithm for Application Server Configuration*, Proceedings of the 13th International Conference on World Wide Web, pp287-296, New York

Yijie, S and Gongzhang, S, 2008, *Improved NSGA-II Multi-Objective Genetic Algorithm Based on Hybridization-Encouraged Mechanism*, Chinese Journal of Aeronautics, (21)6, pp540-549

Yildiz, A R, 2009, *An Effective Hybrid Immune-Hill Climbing Optimization Approach for Solving Design and Manufacturing Optimization Problems in Industry*, Journal of Materials Processing Technology, (209)6, pp2773-2780

Zhang, C, Chen, Y, Shi, M and Peterson, G P, 2009, *Optimization of Heat Pipe with Axial 'Ω'-Shaped Micro Grooves Based on a Niched Pareto Genetic Algorithm (NPGA)*, Applied Thermal Engineering, (29)16, pp3340-3345

APPENDIX A – The Puzzle of PM using Exponential Distribution

Throughout this work, it was assumed that failure characteristics of components follow the Weibull distribution. An earlier attempt however, was made to use the exponential distribution instead and this proved non-applicable in the problem of maintenance. This puzzle is discussed next.

The exponential distribution was the original choice for establishing PM models for reliability, unavailability and cost for components. This was because of its simplicity and ease of use. More so, the exponential distribution is widely used in the application area of component lifetime (MIL-HDBK-338B, 1998). However, using the exponential distribution generated some puzzling results which are here discussed.

The reliability of a component using exponential distribution without the effect of PM is given by equation A.1 (Birolini, 2007).

$$R_m(t) = e^{-\lambda t} \qquad (A.1)$$

To model the effect of PPM on system reliability equation 4.8 (on page 62) which is the universal model for component reliability is used. The probability of surviving until the *n-th* PM stage is given by equation A.2.

$$R(T_p)^n = e^{(-\lambda T_p)^n} \qquad (A.2)$$

The probability of surviving the remaining time t_{rem} is given by equation A.3. Where $t_{rem} = t - nT_p$.

$$R(t_{rem}) = e^{-\lambda(t - nT_p)} \qquad (A.3)$$

Hence, combining equations A.2 and A.3 gives component reliability under PPM using exponential distribution as shown in equation A.4.

$$R_m(t) = e^{(-\lambda T_p)^n} e^{-\lambda(t - nT_p)} \qquad (A.4)$$

Equation A.4 simplifies to:

$$R_m(t) = e^{-n\lambda T_p}\, e^{-(\lambda t - n\lambda T_p)}$$
$$R_m(t) = e^{-n\lambda T_p}\, e^{(-\lambda t + n\lambda T_p)}$$
$$R_m(t) = e^{-n\lambda T_p}\, e^{-\lambda t}\, e^{n\lambda T_p} \quad \text{; applying the laws of indices}$$
$$R_m(t) = e^{-n\lambda T_p}\, e^{n\lambda T_p}\, e^{-\lambda t}$$

Further application of the laws of indices gives:

$$R_m(t) = e^{-\lambda n T_p + \lambda n T_p}\, e^{-\lambda t}$$
$$R_m(t) = e^{0}\, e^{-\lambda t}$$
$$R_m(t) = 1 \,.\, e^{-\lambda t}$$
$$R_m(t) = e^{-\lambda t} \qquad (A.5)$$

Equation A.5 is not different from A.1 which is the reliability R(t) of a component using exponential distribution without PM. This case reveals that the exponential distribution is not sufficient for modelling a real life problem under PM. Such similar case is also found in Ebeling (1997, pp204). A simple conclusion from equations A.1 and A.5 is that under exponential distribution, PPM has no effect on component reliability, and consequently system reliability.

To further probe the characteristics of the exponential distribution, it is worth to also investigate the effect of IPM on component reliability. To do this, the universal model for component reliability under IPM (equation 4.30 on page 71) and equation A.1 are considered; giving equation A.6 as the reliability of a component under IPM using exponential distribution.

$$R_m(t) = \left(\prod_{j=1}^{n} \left(1 - e^{-\lambda((j-1)t_r)} + e^{-\lambda\left((j-1)t_r + T_p\right)} \right) \right) \left(1 - e^{-\lambda(nt_r)} + e^{-\lambda\left(nt_r + (t - nT_p)\right)} \right) \quad (A.6)$$

Under PPM, the improvement factor $f = 1$, whereas under IPM this value is less than 1. This suggests that the effect of PPM on component reliability and eventually system reliability should be better than that of IPM. Therefore it seems logical to conclude that using the exponential distribution, IPM will have negative effect on component reliability, and consequently system reliability since PPM has no effect. Unlike equation A.5, it is unlikely to tell if equation A.6 is better, same or worse than A.1. However, if improvement factor $f = 1$, equation A.6 models a PPM problem and evaluates giving same results as equation A.1 or A.5; this characteristic is as expected.

An alternative approach to investigating IPM using exponential distribution is to illustrate a tabular and graphical representation of the evaluations of equations A.1 and A.6. In order to understand the characteristic of the exponential distribution under IPM, it becomes imperative for the evaluation of equation A.6 to be considered for two different improvements factor. For the purpose of this illustration, the following arbitrary failure data is used.

Failure rate λ = 6.67e-04
Time scale t = 1500
PM time T_p = 180
Improvement factor f_1 = 0.875
Improvement factor f_2 = 0.375

Due to space limitation, it is impossible to show the component reliability for a time step of 1 unit. Therefore a time step of 60 units is considered, implying that the time sequence is 0, 60, 120, … , 1380, 1440, 1500. Table A.1 shows evaluation of the component reliability over time in HiP-HOPS using exponential distribution. It illustrates two major result categories; component reliability (i) under no PM policy and

(ii) under IPM. The latter contains two sub-categories; (a) component reliability with improvement factor $f = f_1 = 0.875$ and (b) $f = f_2 = 0.375$.

To visually depict the effect of IPM, Figure A.1 is the graphic representation of Table A.1. The graphic illustration shown in Figure A.1 takes into account a time step of 1 unit. The red plot or R_1 is the reliability of the component with improvement factor $f = f_2 = 0.375$, the deep red (dark red) plot or R_2 is its equivalent reliability with improvement factor $f = f_1 = 0.875$, while the black plot or R_3 is the reliability under No PM policy.

Table A.1 and Figure A.1 reveal that (i) IPM is better than PPM (ii) component reliability is better improved when the improvement factor value is lower; e.g. component reliability is improved with $f = 0.375$ than with $f = 0.875$. This violates the assertion of the age reduction model and therefore it can be concluded that the exponential distribution is not sufficient in modelling a PM problem under the age reduction model.

Table A.1 - Component reliability using exponential distribution under IPM and No PM

Component Reliability using Exponential Distribution			
		With IPM	
Time (t)	**No PM = PPM** ($f = 1$)	$f = f_1 = 0.375$	$f = f_2 = 0.875$
0	1	1	1
60	0.960789	0.960789	0.960789
120	0.923116	0.923116	0.923116
180	0.886920	0.886920	0.886920
240	0.852144	0.854657	0.852662
300	0.818731	0.823658	0.819746
360	0.786628	0.793875	0.788121
420	0.755784	0.767082	0.758132
480	0.726149	0.741340	0.729318
540	0.697676	0.716608	0.701635
600	0.670320	0.694171	0.675334
660	0.644036	0.672613	0.650064
720	0.618783	0.651901	0.625785
780	0.594521	0.632965	0.602677
840	0.571209	0.614771	0.580474
900	0.548812	0.597291	0.559143
960	0.527292	0.581194	0.538803
1020	0.506617	0.565729	0.519260
1080	0.486752	0.550870	0.500484
1140	0.467666	0.537097	0.482548
1200	0.449329	0.523865	0.465316
1260	0.431711	0.511151	0.448760
1320	0.414783	0.499295	0.432918
1380	0.398519	0.487903	0.417697
1440	0.382893	0.476959	0.403073
1500	0.367879	0.466695	0.389055

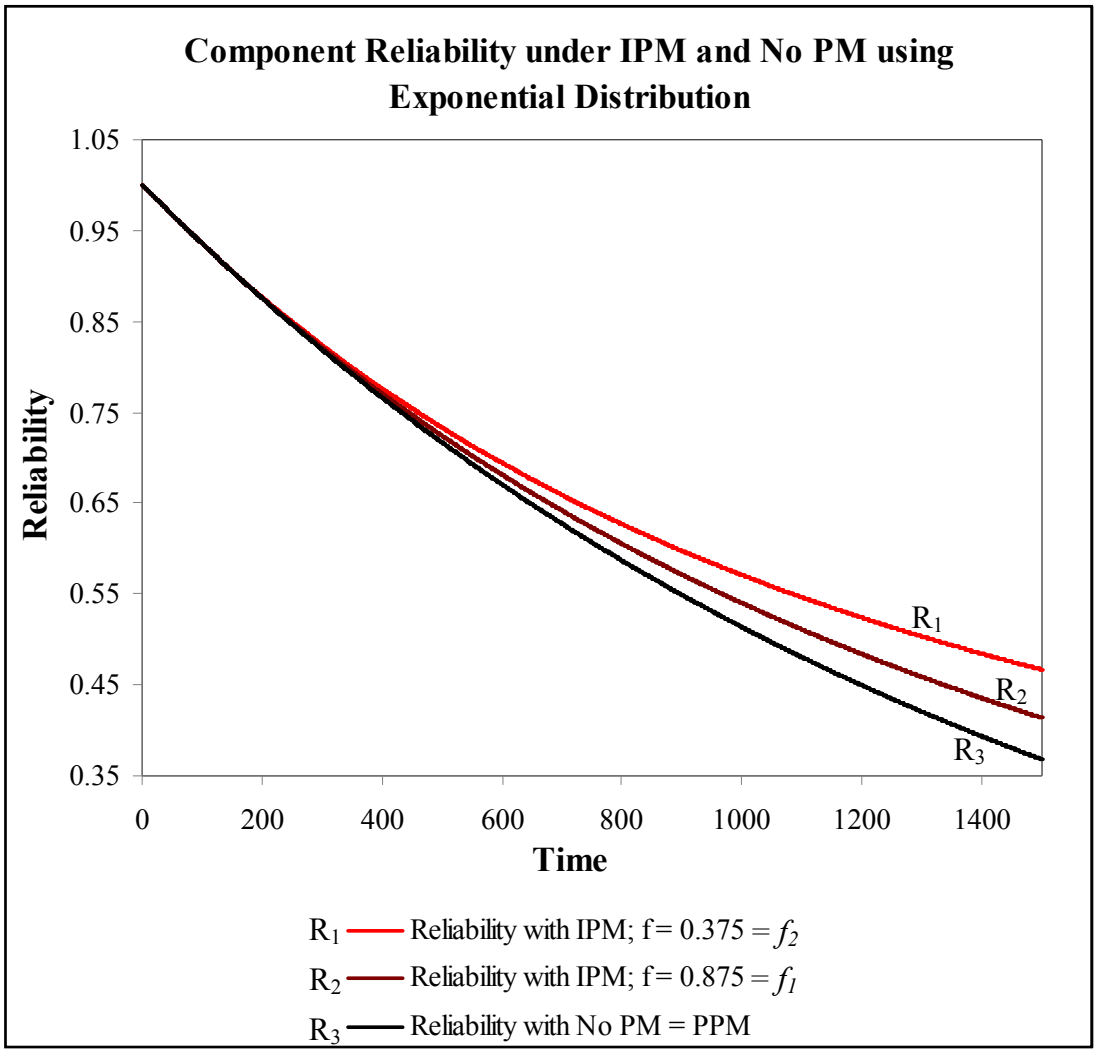

Figure A.1 - Component reliability under IPM and No PM using exponential distribution

APPENDIX B – FOSS Optimal PPM Schedules under Component Substitution in Early Generations

The following tables show the set of optimal PPM schedules obtained in generations 1 to 3 under component substitution of secondary constraints. The tables reveal in these early generations the existence of diverse implementations of the components subjected to component substitution. The progressive search of optimal PPM schedules through to 5120 generation as presented in chapter 7 (section 7.3.1.2) consists of the components implementations that were found best suitable for the given design objectives, while weaker implementations were dominated.

Table B.1 – FOSS optimal PPM schedules under component substitution found in generation 1

Optimal PPM Schedule	Cost	Unavailability	Generation
af(6) bp(14) cp(11) fm.fm_3(3) ht.ht_4(6) if(5) me(12) mt.mt_3(5) st(10) vm(8)	20726	0.151054	1
af(16) bp(12) cp(16) fm.fm_5(11) ht.ht_2(1) if(3) me(18) mt.mt_4(12) st(12) vm(7)	19662	0.188598	1
af(5) bp(4) cp(9) fm.fm_5(9) ht.ht_1(8) if(1) me(13) mt.mt_3(2) st(8) vm(13)	22064	0.12543	1
af(1) bp(8) cp(7) fm.fm_1(8) ht.ht_2(3) if(4) me(13) mt.mt_2(10) st(2) vm(14)	27627	0.117562	1
af(1) bp(18) cp(4) fm.fm_4(2) ht.ht_2(1) if(6) me(8) mt.mt_3(7) st(6) vm(4)	28210	0.112561	1
af(4) bp(3) cp(5) fm.fm_3(5) ht.ht_1(11) if(2) me(12) mt.mt_1(10) st(14) vm(4)	22381	0.120765	1
af(8) bp(7) cp(16) fm.fm_3(9) ht.ht_1(1) if(15) me(18) mt.mt_3(9) st(8) vm(13)	20213	0.176841	1
af(20) bp(13) cp(11) fm.fm_5(8) ht.ht_4(6) if(16) me(13) mt.mt_1(7) st(16) vm(8)	19495	0.225869	1

Table B.2 – FOSS optimal PPM schedules under component substitution found in generation 2

Optimal PPM Schedule	Cost	Unavailability	Generation
af(6) bp(14) cp(11) fm.fm_3(3) ht.ht_4(6) if(5) me(12) mt.mt_3(5) st(10) vm(8)	20726	0.151054	1
af(16) bp(12) cp(16) fm.fm_5(11) ht.ht_2(1) if(3) me(18) mt.mt_4(12) st(12) vm(7)	19662	0.188598	1
af(1) bp(3) cp(4) fm.fm_4(2) ht.ht_1(11) if(2) me(8) mt.mt_3(7) st(6) vm(4)	29048	0.083612	2
af(8) bp(7) cp(16) fm.fm_3(6) ht.ht_1(11) if(15) me(18) mt.mt_3(9) st(8) vm(13)	19774	0.186406	2
af(5) bp(3) cp(9) fm.fm_3(5) ht.ht_1(8) if(2) me(12) mt.mt_3(2) st(14) vm(13)	21320	0.130654	2
af(4) bp(4) cp(9) fm.fm_3(5) ht.ht_1(11) if(2) me(13) mt.mt_3(2) st(8) vm(13)	21649	0.121931	2
af(2) bp(3) cp(7) fm.fm_3(5) ht.ht_1(11) if(4) me(13) mt.mt_1(10) st(2) vm(4)	25706	0.095798	2
af(5) bp(4) cp(9) fm.fm_5(1) ht.ht_1(6) if(1) me(13) mt.mt_3(2) st(8) vm(13)	23395	0.107477	2
af(4) bp(12) cp(5) fm.fm_3(5) ht.ht_2(1) if(2) me(18) mt.mt_4(12) st(6) vm(4)	22392	0.110819	2
af(20) bp(13) cp(9) fm.fm_5(9) ht.ht_1(8) if(6) me(13) mt.mt_1(7) st(16) vm(13)	19296	0.210463	2
af(16) bp(12) cp(11) fm.fm_5(8) ht.ht_2(1) if(3) me(18) mt.mt_1(7) st(16) vm(8)	19909	0.173911	2
af(8) bp(14) cp(11) fm.fm_3(9) ht.ht_1(1) if(5) me(12) mt.mt_3(5) st(10) vm(13)	20365	0.157951	2
af(8) bp(7) cp(5) fm.fm_3(5) ht.ht_1(11) if(2) me(18) mt.mt_1(11) st(14) vm(13)	20669	0.155199	2
af(5) bp(4) cp(9) fm.fm_3(6) ht.ht_1(1) if(15) me(18) mt.mt_3(2) st(8) vm(13)	21193	0.137173	2
af(8) bp(7) cp(5) fm.fm_3(5) ht.ht_1(11) if(2) me(18) mt.mt_3(9) st(8) vm(13)	20998	0.138374	2
af(4) bp(3) cp(5) fm.fm_3(5) ht.ht_2(3) if(4) me(5) mt.mt_1(10) st(2) vm(4)	25873	0.084505	2
af(8) bp(7) cp(11) fm.fm_5(8) ht.ht_4(6) if(16) me(18) mt.mt_1(7) st(8) vm(8)	20118	0.173308	2
af(16) bp(12) cp(9) fm.fm_5(11) ht.ht_1(8) if(3) me(13) mt.mt_4(12) st(12) vm(13)	19345	0.193829	2
af(4) bp(3) cp(5) fm.fm_3(5) ht.ht_1(11) if(2) me(12) mt.mt_1(10) st(8) vm(4)	22641	0.108369	2
af(8) bp(7) cp(4) fm.fm_3(9) ht.ht_1(1) if(6) me(18) mt.mt_3(7) st(6) vm(4)	21888	0.113971	2

Table B.3 – FOSS optimal PPM schedules under component substitution found in generation 3

Optimal PPM Schedule	Cost	Unavailability	Generation
af(4) bp(4) cp(9) fm.fm_3(5) ht.ht_1(11) if(2) me(13) mt.mt_3(2) st(8) vm(13)	21649	0.121931	2
af(16) bp(12) cp(9) fm.fm_5(11) ht.ht_1(8) if(3) me(13) mt.mt_4(12) st(12) vm(13)	19345	0.193829	2
af(8) bp(7) cp(4) fm.fm_3(9) ht.ht_1(1) if(6) me(18) mt.mt_3(7) st(6) vm(4)	21888	0.113971	2
af(8) bp(7) cp(5) fm.fm_3(5) ht.ht_1(11) if(2) me(15) mt.mt_1(7) st(8) vm(8)	21104	0.131113	3
af(4) bp(3) cp(5) fm.fm_3(5) ht.ht_2(3) if(2) me(18) mt.mt_1(10) st(2) vm(4)	25003	0.079217	3
af(8) bp(7) cp(9) fm.fm_3(9) ht.ht_1(8) if(6) me(18) mt.mt_3(5) st(14) vm(4)	20125	0.149747	3
af(6) bp(7) cp(4) fm.fm_3(9) ht.ht_1(1) if(6) me(18) mt.mt_3(7) st(6) vm(4)	22153	0.109216	3
af(8) bp(3) cp(11) fm.fm_3(5) ht.ht_2(3) if(16) me(18) mt.mt_1(7) st(8) vm(8)	20388	0.147981	3
af(8) bp(7) cp(11) fm.fm_3(5) ht.ht_1(11) if(2) me(18) mt.mt_1(7) st(8) vm(8)	20604	0.140048	3
af(8) bp(13) cp(11) fm.fm_3(9) ht.ht_1(8) if(6) me(12) mt.mt_1(7) st(16) vm(13)	19393	0.182096	3
af(8) bp(7) cp(5) fm.fm_3(5) ht.ht_1(11) if(2) me(12) mt.mt_3(9) st(8) vm(4)	21523	0.123944	3
af(4) bp(7) cp(5) fm.fm_3(5) ht.ht_2(5) if(2) me(18) mt.mt_1(10) st(2) vm(4)	24596	0.091059	3
af(8) bp(13) cp(11) fm.fm_3(6) ht.ht_1(11) if(5) me(18) mt.mt_3(5) st(8) vm(13)	19926	0.161357	3
af(8) bp(7) cp(12) fm.fm_3(5) ht.ht_1(1) if(2) me(18) mt.mt_3(2) st(8) vm(13)	21191	0.126583	3
af(8) bp(3) cp(4) fm.fm_3(9) ht.ht_1(1) if(6) me(11) mt.mt_3(7) st(6) vm(13)	21755	0.119722	3
af(5) bp(7) cp(9) fm.fm_3(5) ht.ht_1(6) if(2) me(13) mt.mt_3(2) st(8) vm(13)	21229	0.124035	3
af(4) bp(3) cp(5) fm.fm_4(2) ht.ht_2(3) if(2) me(8) mt.mt_1(1) st(2) vm(4)	28802	0.062674	3
af(16) bp(4) cp(9) fm.fm_5(11) ht.ht_1(8) if(15) me(18) mt.mt_4(12) st(12) vm(13)	19289	0.202093	3
af(4) bp(4) cp(9) fm.fm_3(9) ht.ht_1(1) if(2) me(13) mt.mt_3(5) st(8) vm(13)	21874	0.116201	3
af(8) bp(3) cp(5) fm.fm_3(5) ht.ht_1(11) if(2) me(18) mt.mt_1(5) st(8) vm(4)	22047	0.111487	3

af(8) bp(4) cp(9) fm.fm_3(5) ht.ht_1(11) if(2) me(13) mt.mt_3(7) st(8) vm(13)	20694	0.136227	3
af(16) bp(7) cp(11) fm.fm_5(8) ht.ht_1(10) if(3) me(18) mt.mt_1(7) st(12) vm(8)	19865	0.172717	3
af(16) bp(12) cp(9) fm.fm_5(11) ht.ht_1(8) if(9) me(13) mt.mt_4(12) st(12) vm(13)	19065	0.20627	3
af(4) bp(3) cp(9) fm.fm_3(5) ht.ht_1(11) if(2) me(18) mt.mt_3(2) st(8) vm(13)	21845	0.119518	3
af(16) bp(7) cp(9) fm.fm_5(11) ht.ht_1(8) if(3) me(18) mt.mt_4(12) st(8) vm(8)	19878	0.169696	3
af(16) bp(12) cp(9) fm.fm_3(9) ht.ht_1(8) if(3) me(13) mt.mt_4(12) st(6) vm(13)	19697	0.175777	3
af(11) bp(7) cp(5) fm.fm_3(5) ht.ht_2(1) if(2) me(18) mt.mt_3(9) st(8) vm(13)	20875	0.132238	3
af(4) bp(3) cp(5) fm.fm_3(5) ht.ht_2(3) if(4) me(12) mt.mt_1(9) st(8) vm(4)	22313	0.098893	3

APPENDIX C – FOSS Optimal PPM Schedules under Composite Constraint in Early Generations

The following tables show the set of optimal PPM schedules obtained in generations 1 to 3 under composite constraint. For those components that are subjected to substitution, the tables reveal the existence of diverse implementations at these early generations. The progressive search through to 5120 generation as presented in chapter 7 (section 7.3.1.4) consists of the components implementations that were found best suitable for the given design objectives, while weaker implementations were dominated.

Table C.1 – FOSS optimal composite PPM schedules found in generation 1

Optimal PPM Schedule	Cost	Unavailability	Generation
af(8) bp(3) cp(15) fm.fm_3(7) ht.ht_2(4) if(16) me(8) mt.mt_3(7) st(7) vm(4)	21568	0.149628	1
af(12) bp(3) cp(10) fm.fm_3(6) ht.ht_1(2) if(5) me(8) mt.mt_4(5) st(7) vm(4)	21662	0.121737	1
af(12) bp(15) cp(10) fm.fm_5(6) ht.ht_2(3) if(9) me(8) mt.mt_4(9) st(7) vm(4)	20828	0.159168	1
af(20) bp(5) cp(15) fm.fm_5(10) ht.ht_2(3) if(8) me(8) mt.mt_2(13) st(7) vm(4)	20657	0.184962	1
af(15) bp(15) cp(5) fm.fm_3(11) ht.ht_1(3) if(7) me(8) mt.mt_3(6) st(7) vm(4)	21405	0.154631	1
af(11) bp(1) cp(6) fm.fm_5(4) ht.ht_2(4) if(1) me(8) mt.mt_2(10) st(7) vm(4)	23978	0.108053	1
af(15) bp(8) cp(12) fm.fm_3(2) ht.ht_1(4) if(7) me(8) mt.mt_1(6) st(7) vm(4)	21658	0.142682	1
af(8) bp(1) cp(1) fm.fm_5(11) ht.ht_1(1) if(9) me(8) mt.mt_3(2) st(7) vm(4)	27422	0.10267	1
af(5) bp(7) cp(13) fm.fm_3(7) ht.ht_1(14) if(8) me(8) mt.mt_2(10) st(7) vm(4)	21258	0.156341	1
af(16) bp(11) cp(9) fm.fm_5(9) ht.ht_2(9) if(2) me(8) mt.mt_3(6) st(7) vm(4)	21502	0.150245	1
af(1) bp(8) cp(5) fm.fm_1(7) ht.ht_2(6) if(3) me(8) mt.mt_3(2) st(7) vm(4)	27643	0.095736	1
af(7) bp(6) cp(15) fm.fm_5(11) ht.ht_2(2) if(13) me(8) mt.mt_4(7) st(7) vm(4)	21428	0.154142	1

Table C.2 – FOSS optimal composite PPM schedules found in generation 2

Optimal PPM Schedule	Cost	Unavailability	Generation
af(15) bp(6) cp(12) fm.fm_3(2) ht.ht_2(2) if(7) me(8) mt.mt_4(7) st(7) vm(4)	21364	0.135692	2
af(5) bp(7) cp(5) fm.fm_3(7) ht.ht_2(6) if(3) me(8) mt.mt_2(10) st(7) vm(4)	22030	0.116058	2
af(1) bp(7) cp(5) fm.fm_3(7) ht.ht_2(6) if(3) me(8) mt.mt_3(4) st(7) vm(4)	27086	0.095045	2
af(15) bp(8) cp(6) fm.fm_5(4) ht.ht_1(3) if(7) me(8) mt.mt_2(10) st(7) vm(4)	21168	0.142526	2
af(8) bp(1) cp(1) fm.fm_5(11) ht.ht_2(4) if(1) me(8) mt.mt_3(2) st(7) vm(4)	28156	0.087099	2
af(1) bp(1) cp(6) fm.fm_1(7) ht.ht_2(4) if(1) me(8) mt.mt_3(2) st(7) vm(4)	30054	0.073886	2
af(1) bp(6) cp(5) fm.fm_1(7) ht.ht_2(6) if(3) me(8) mt.mt_3(2) st(7) vm(4)	27741	0.091945	2
af(12) bp(3) cp(10) fm.fm_3(6) ht.ht_1(2) if(2) me(8) mt.mt_4(5) st(7) vm(4)	22082	0.114116	2
af(5) bp(1) cp(12) fm.fm_5(4) ht.ht_2(4) if(1) me(8) mt.mt_2(10) st(7) vm(4)	24523	0.101188	2
af(16) bp(11) cp(9) fm.fm_5(10) ht.ht_2(9) if(8) me(8) mt.mt_2(13) st(7) vm(4)	20431	0.177785	2
af(11) bp(11) cp(6) fm.fm_5(4) ht.ht_1(4) if(7) me(8) mt.mt_2(10) st(7) vm(4)	21029	0.145604	2
af(12) bp(3) cp(10) fm.fm_3(2) ht.ht_2(6) if(3) me(8) mt.mt_3(1) st(7) vm(4)	22468	0.108529	2

Table C.3 – FOSS optimal composite PPM schedules found in generation 3

Optimal PPM Schedule	Cost	Unavailability	Generation
af(5) bp(7) cp(5) fm.fm_3(7) ht.ht_2(4) if(3) me(8) mt.mt_2(10) st(7) vm(4)	22045	0.113451	3
af(12) bp(3) cp(10) fm.fm_3(2) ht.ht_2(6) if(3) me(8) mt.mt_4(5) st(7) vm(4)	22018	0.114269	3
af(8) bp(3) cp(1) fm.fm_3(2) ht.ht_2(6) if(1) me(8) mt.mt_3(1) st(7) vm(4)	27463	0.077697	3
af(12) bp(3) cp(6) fm.fm_5(4) ht.ht_1(3) if(7) me(8) mt.mt_2(10) st(7) vm(4)	21560	0.127919	3
af(1) bp(1) cp(6) fm.fm_1(7) ht.ht_1(2) if(1) me(8) mt.mt_3(2) st(7) vm(4)	30293	0.072065	3
af(11) bp(11) cp(5) fm.fm_3(7) ht.ht_1(4) if(3) me(8) mt.mt_2(10) st(7) vm(4)	21227	0.138616	3
af(11) bp(11) cp(6) fm.fm_3(7) ht.ht_1(4) if(3) me(8) mt.mt_2(10) st(7) vm(4)	20977	0.140268	3
af(12) bp(3) cp(10) fm.fm_5(4) ht.ht_1(3) if(3) me(8) mt.mt_3(1) st(7) vm(4)	22461	0.110286	3
af(12) bp(3) cp(6) fm.fm_3(6) ht.ht_1(2) if(2) me(8) mt.mt_2(10) st(7) vm(4)	21911	0.116146	3
af(1) bp(1) cp(1) fm.fm_5(11) ht.ht_2(6) if(1) me(8) mt.mt_3(2) st(7) vm(4)	33176	0.071926	3
af(5) bp(1) cp(10) fm.fm_3(2) ht.ht_2(4) if(3) me(8) mt.mt_3(1) st(7) vm(4)	24552	0.084376	3
af(11) bp(11) cp(9) fm.fm_5(10) ht.ht_2(9) if(8) me(8) mt.mt_2(10) st(7) vm(4)	20431	0.164848	3
af(11) bp(11) cp(6) fm.fm_3(7) ht.ht_1(4) if(7) me(8) mt.mt_2(10) st(7) vm(4)	20697	0.149898	3
af(16) bp(10) cp(9) fm.fm_5(4) ht.ht_2(9) if(7) me(8) mt.mt_2(10) st(7) vm(4)	20689	0.1614	3
af(1) bp(3) cp(5) fm.fm_3(2) ht.ht_2(6) if(3) me(8) mt.mt_3(2) st(7) vm(4)	28068	0.074323	3

APPENDIX D – FOSS Optimal IPM Schedules under Component Substitution in Early Generations

The following tables show the set of optimal IPM schedules obtained in generations 1 to 3 under component substitution of the established secondary constraints. The tables show that at early generations there exist diverse implementations of the components subjected to component substitution. The progressive search which stretched to 5120 generation as presented in chapter 7 (section 7.3.2.2) consists of the components implementations that were found suitable for the given design objectives, while weaker implementations were dominated.

Table D.1 – FOSS optimal IPM schedules under component substitution found in generation 1

Optimal PPM Schedule	Cost	Unavailability	Generation
af(18) bp(10) cp(10) fm.fm_3(11) ht.ht_1(6) if(12) me(5) mt.mt_2(6) st(11) vm(14)	19814.3	0.335068	1
af(17) bp(16) cp(7) fm.fm_5(8) ht.ht_4(7) if(21) me(18) mt.mt_2(48) st(17) vm(11)	18644.1	0.583008	1

Table D.2 – FOSS optimal IPM schedules under component substitution found in generation 2

Optimal PPM Schedule	Cost	Unavailability	Generation
af(17) bp(16) cp(10) fm.fm_5(8) ht.ht_4(7) if(12) me(18) mt.mt_2(32) st(17) vm(11)	18392.5	0.50632	2
af(18) bp(10) cp(7) fm.fm_3(11) ht.ht_4(7) if(12) me(18) mt.mt_2(6) st(17) vm(14)	18467.5	0.373066	2

Table D.3 – FOSS optimal IPM schedules under component substitution found in generation 3

Optimal PPM Schedule	Cost	Unavailability	Generation
af(18) bp(10) cp(7) fm.fm_3(1) ht.ht_4(7) if(12) me(18) mt.mt_2(6) st(17) vm(14)	18467.5	0.296113	3
af(17) bp(16) cp(10) fm.fm_5(8) ht.ht_4(7) if(12) me(18) mt.mt_2(6) st(17) vm(14)	18393.4	0.339928	3
af(18) bp(10) cp(10) fm.fm_3(11) ht.ht_4(7) if(12) me(18) mt.mt_2(6) st(17) vm(14)	18216.7	0.365653	3
af(18) bp(10) cp(10) fm.fm_3(11) ht.ht_4(7) if(12) me(18) mt.mt_2(32) st(17) vm(11)	18215.8	0.525561	3
af(17) bp(10) cp(10) fm.fm_5(8) ht.ht_4(7) if(12) me(18) mt.mt_2(6) st(17) vm(14)	18392	0.344728	3

APPENDIX E – FOSS Optimal IPM Schedules under Composite Constraint in Early Generations

The following tables show the set of optimal IPM schedules obtained in generations 1 to 3 under composite constraint. For components that are subjected to substitution, the tables show how diverse the implementations could be at these early generations. The progressive search for optimal IPM schedules which stretched through to 5120 generation as presented in chapter 7 (section 7.3.2.4) consists of the components implementations that were found best suitable for the given design objectives, while weaker implementations were dominated.

Table E.1 – FOSS optimal composite IPM schedules found in generation 1

Optimal IPM Schedule	Cost	Unavailability	Generation
af(11) bp(13) cp(8) fm.fm_1(18) ht.ht_4(13) if(11) me(8) mt.mt_1(7) st(7) vm(4)	21200.9	0.443305	1
af(18) bp(9) cp(14) fm.fm_1(18) ht.ht_2(7) if(13) me(8) mt.mt_1(5) st(7) vm(4)	21313	0.400925	1
af(20) bp(13) cp(9) fm.fm_3(14) ht.ht_2(18) if(20) me(8) mt.mt_4(23) st(7) vm(4)	20244.6	0.614967	1
af(21) bp(11) cp(10) fm.fm_5(52) ht.ht_2(52) if(16) me(8) mt.mt_2(33) st(7) vm(4)	20103.8	0.825095	1
af(13) bp(10) cp(12) fm.fm_5(27) ht.ht_1(9) if(21) me(8) mt.mt_2(43) st(7) vm(4)	20122.8	0.687246	1
af(15) bp(12) cp(16) fm.fm_1(14) ht.ht_4(1) if(3) me(8) mt.mt_2(27) st(7) vm(4)	20753.1	0.519288	1
af(9) bp(13) cp(15) fm.fm_1(11) ht.ht_3(7) if(20) me(8) mt.mt_4(19) st(7) vm(4)	21004.7	0.497803	1
af(13) bp(13) cp(12) fm.fm_3(62) ht.ht_2(35) if(19) me(8) mt.mt_2(80) st(7) vm(4)	19923.9	0.894676	1
af(12) bp(19) cp(9) fm.fm_5(36) ht.ht_4(43) if(13) me(8) mt.mt_2(68) st(7) vm(4)	20041.6	0.85509	1

Table E.2 – FOSS optimal composite IPM schedules found in generation 2

Optimal IPM Schedule	Cost	Unavailability	Generation
af(20) bp(13) cp(16) fm.fm_1(14) ht.ht_4(1) if(20) me(8) mt.mt_4(23) st(7) vm(4)	20654.5	0.461939	2
af(12) bp(19) cp(9) fm.fm_3(14) ht.ht_2(18) if(13) me(8) mt.mt_2(68) st(7) vm(4)	19923.6	0.756157	2
af(11) bp(10) cp(8) fm.fm_1(18) ht.ht_1(9) if(21) me(8) mt.mt_1(7) st(7) vm(4)	21282.4	0.359507	2
af(15) bp(12) cp(16) fm.fm_1(14) ht.ht_4(1) if(3) me(8) mt.mt_2(15) st(7) vm(4)	20753.1	0.444993	2
af(13) bp(13) cp(9) fm.fm_3(62) ht.ht_4(43) if(19) me(8) mt.mt_2(68) st(7) vm(4)	19864.9	0.884303	2
af(11) bp(13) cp(8) fm.fm_1(18) ht.ht_4(13) if(19) me(8) mt.mt_1(7) st(7) vm(4)	21201.6	0.439853	2
af(11) bp(13) cp(12) fm.fm_3(62) ht.ht_4(13) if(19) me(8) mt.mt_2(80) st(7) vm(4)	19865	0.868525	2
af(13) bp(11) cp(10) fm.fm_3(62) ht.ht_2(52) if(19) me(8) mt.mt_2(33) st(7) vm(4)	19922.8	0.841809	2
af(20) bp(11) cp(9) fm.fm_3(14) ht.ht_2(18) if(20) me(8) mt.mt_2(2) st(7) vm(4)	19927.1	0.514662	2

Table E.3 – FOSS optimal composite IPM schedules found in generation 3

Optimal IPM Schedule	Cost	Unavailability	Generation
af(11) bp(11) cp(9) fm.fm_3(37) ht.ht_4(13) if(19) me(8) mt.mt_2(2) st(7) vm(4)	19863.6	0.60447	3
af(20) bp(13) cp(16) fm.fm_3(14) ht.ht_4(1) if(20) me(8) mt.mt_2(2) st(7) vm(4)	19872.4	0.401902	3
af(20) bp(13) cp(9) fm.fm_3(14) ht.ht_4(13) if(19) me(8) mt.mt_2(2) st(7) vm(4)	19869.4	0.486034	3
af(20) bp(11) cp(9) fm.fm_1(14) ht.ht_4(1) if(20) me(8) mt.mt_2(2) st(7) vm(4)	20334.3	0.330551	3