An Intelligent Manufacturing System for Heat Treatment Scheduling

A thesis submitted for the degree of Doctor of Philosophy

By

Tawfeeq Al-Kanhal

School of Engineering and Design, Brunel University

April 2010

Dedication

This work is dedicated to my dear wife who I am indebted to, without her continuous sacrifices and patience the completion of this project could not be possible.

I also dedicate this work to the principles of each and every ambitious person who knows that patience and hard work make the dreams

Acknowledgements

Firstly, all praise is due to Allah (Glorified and Exalted is He), without his immeasurable blessings and favours none of this could have been possible.

I also wish to express thanks to Dr M. Abbod whom I have learned a lot from his extreme guidance, critical discussions and eternal patience.

Most importantly, acknowledgments also attached to my parents, brothers, sisters and daughter and her brothers for their love, commitment, highly appreciated caring of my family during some parts of my study and their encouragement. I am delighted also to express my regards to friends whom showed their minuteness, had a voice in this work and ferrying me around in time.

Abstract

This research is focused on the integration problem of process planning and scheduling in steel heat treatment operations environment using artificial intelligent techniques that are capable of dealing with such problems.

This work addresses the issues involved in developing a suitable methodology for scheduling heat treatment operations of steel. Several intelligent algorithms have been developed for these propose namely, Genetic Algorithm (GA), Sexual Genetic Algorithm (SGA), Genetic Algorithm with Chromosome differentiation (GACD), Age Genetic Algorithm (AGA), and Mimetic Genetic Algorithm (MGA). These algorithms have been employed to develop an efficient intelligent algorithm using Algorithm Portfolio methodology. After that all the algorithms have been tested on two types of scheduling benchmarks.

To apply these algorithms on heat treatment scheduling, a furnace model is developed for optimisation proposes. Furthermore, a system that is capable of selecting the optimal heat treatment regime is developed so the required metal properties can be achieved with the least energy consumption and the shortest time using Neuro-Fuzzy (NF) and Particle Swarm Optimisation (PSO) methodologies. Based on this system, PSO is used to optimise the heat treatment process by selecting different heat treatment conditions. The selected conditions are evaluated so the best selection can be identified. This work addresses the issues involved in developing a suitable methodology for developing an NF system and PSO for mechanical properties of the steel.

Using the optimisers, furnace model and heat treatment system model, the intelligent system model is developed and implemented successfully. The results of this system were exciting and the optimisers were working correctly.

List of Publications

Conferences'

1. T. Al-Kanhal and M.F. Abbod, Multi-agent System for Dynamic Manufacturing System Optimization, ICCS 2008, Part III, LNCS 5103, pp 634–643, Springer-Verlag, Berlin, Heidelberg 2008.
2. T. Al-Kanhal and M.F. Abbod, Intelligent Scheduling of Dynamic Manufacturing System, ResCon-2008, 1st SED Research Conference 2008, Brunel University, London, UK.
3. T. Al-Kanhal and M.F. Abbod, Scheduling of Dynamic Manufacturing System using an Intelligent Multi Agent Approach, SIIC'2008, Leeds, UK.
4. T. Al-Kanhal and M.F. Abbod, Modelling and Optimisation of Reheat Furnace, European Simulation Symposium, 8 –10 September, 2008, Liverpool, UK.
5. T. Al-Kanhal and M.F. Abbod, Intelligent Scheduling of Dynamic Manufacturing System, ResCon-2009, 2nd SED Research Conference 2009, Brunel University, London, UK.
6. T. Al-Kanhal and M.F. Abbod, Intelligent Modelling and Optimisation of Metals Heat Treatment Process, SIIC'2009, Surrey, UK.

Journals

7. T. Al-Kanhal and M.F. Abbod, Algorithms Portfolio for Solving Scheduling Problems. International Journal of Artificial Intelligence & Applications (IJAIA), 2010 submitted.
8. T. Al-Kanhal and M.F. Abbod, Intelligent Manufacturing Model for Scheduling Steel Heat treatment Operations. Journal of Scheduling, 2010 submitted.

List of Acronyms

AGA	Age Genetic Algorithm
AP	Algorithm Portfolios
ACO	Ant Colony Optimisation
BCO	Bee Colony Optimisation
EDD	Earliest Due Date
FSSP	Flowshop Scheduling Problems
GA	Genetic Algorithm
GACD	Genetic Algorithms with Chromosome Differentiations
JSSP	Shop Scheduling Problems
LB	Lower Bound
MGA	Mimetic Genetic Algorithms
NF	Neuro Fuzzy
NP-hard	Non-deterministic Polynomial-time
OSSP	Open Shop Scheduling Problem
PF	Pareto Front
PSO	Particle Swarm Optimisation
SGA	Sexual Genetic Algorithms
SMSP	Single Machine Scheduling Problem
TS	Tabu Search
UB	Upper Bound
WSM	Weighted Sum Method

Table of Contents

Chapter 1 Introduction and Outlines....1
1.1 Introduction....1
1.2 Motivations....3
1.3 Aims and Objectives....4
1.4 Challenges....5
1.5 Contributions....6
1.6 Thesis Outline....7
Chapter 2 Literature Review....8
2.1 Introduction....8
2.2 Scheduling Classification....9
2.2.1 Dispatching Rules and Scheduling....9
2.2.2 Genetic Algorithm....10
2.2.2.1 Application of GA to Job Shop Scheduling Problems....11
2.2.2.2 Application of GA to Flowshop Scheduling Problems....14
2.2.2.3 Applications of GA to Open Shop Scheduling Problem....15
2.2.2.4 Applications of GA to Single Machine Scheduling Problem....16
2.2.3 Application of Swarm Optimisation to Scheduling Problems....19
2.2.3.1 Application of Ant Colony Optimisation to Scheduling Problems....19
2.2.3.2 Application of Bee Colony Optimisation to Scheduling Problems....20
2.2.3.3 Application of Particle Swarm Optimisation to Scheduling Problems....20
2.2.4 Application of Scheduling to Heating Treatment Operations....23
2.3 Summary....24
Chapter 3 Intelligent Optimisation Techniques....25
3.1 Introduction....25
3.2 Genetic Algorithm....26
3.2.1 Implementation of GA....26
3.2.1.1 Initialisation and Representation....26
3.2.1.2 The Fitness Function....27
3.2.1.3 The Reproduction Operators....28
3.2.1.4 Selection Operators....28
a) Roulette Wheel Selection....29
b) Stochastic Universal Sampling....29
c) Truncation Selection....29
d) Tournament Selection....29
e) Local Neighbourhood Selection....30
f) Ranking selection....30

g) Steady State Selection 30
3.2.1.5 Crossover Operator 30
a) Single point crossover 31
b) Multi point crossover 31
3.2.1.6 Mutation Operator 32
3.2.1.7 Elitism 32
3.2.1.8 Termination 32
3.3 Sexual Genetic Algorithm 34
3.4 Age Genetic Algorithm 35
3.5 Genetic Algorithm with Chromosome Differentiation 37
3.6. Mimetic Genetic Algorithms 39
3.3 Particle Swarm Optimisation 41
3.3.1 The Concepts of Particle Swarm Optimisation 42
3.4 Comparison between GA and PSO 43
3.5 Algorithm Portfolios 44
3.6 Summary 46
Chapter 4 Scheduling 47
4.1 Introduction to Scheduling 47
4.2 Scheduling Classification 47
4.3 Definition of the Problem 48
4.4 Complexity of Algorithm 49
4.5 Complexity of Problem 50
4.6 Single Resource Scheduling with Tardiness and Earliest Penalties 50
4.6.1 Due-Date Time 51
4.6.2 Earliest Due Date Time 51
4.6.3 Tardiness Due-Date Time 51
4.6.4 Single Resource Scheduling with Tardiness and Earliest Penalties Model 51
4.6.5 Complexity of problem 52
4.7 Job Shop Scheduling Problem 53
4.7.1 Complexity of the Problem 54
4.8 Flow Shop Scheduling Problem 54
4.8.1 Complexity of the Problem 54
4.9 Open Shop Scheduling Problem 55
4.9.1 Complexity of the Problem 56
4.10 Optimisation Techniques for Solving Scheduling Problems 57
4.10.1 Dispatching Rules 57
4.10.2 Mathematical Programming 58
4.10.2.1 Integer Programming 58
4.10.2.2 Branch and Bound 58
4.10.2.3 Dynamic Programming 58

4.10.3 Heuristic Algorithm 59
4.10.4 Artificial Intelligence 59
4.10.4.1 Simulated Annealing 59
4.10.4.2 Ant Colony Optimisation 60
4.10.4.3 Tabu Search 60
4.10.4.4 Genetic Algorithm 61
4.10.4.5 Particle Swarm Optimisation 61
4.11 Scheduling Benchmarks 61
4.11.1 Opens Shop Benchmarks 61
4.11.2 Single Resource Benchmarks 62
4.12 Summary 62
Chapter 5 Intelligent Scheduling Systems Developments 63
5.1 Introduction 63
5.2 Developed Optimisers 63
5.3 Genetic Algorithm 64
5.3.1 The Fitness Function 64
5.3.1.1 Pareto Front 64
5.3.1.2 Weighted Sum Method 65
5.3.1.3 Single Objective Fitness 65
5.3.2 Encoding 65
5.3.3 Reproduction Operators 67
5.3.3.1 Crossover Operator 68
a) Classical Crossover 68
b) Symbolic Crossover 69
c) Multi Crossover 69
5.3.3.2 Mutation 69
5.3.3.3 GA Selection Operator 70
5.4 Classic GA Optimiser 73
5.5 SGA Optimiser 73
5.6 AGA Optimiser 74
5.7 GACD optimiser 76
5.8 Particle Swarm Optimisation 77
5.9 Mimetic Genetic Algorithms 78
5.10 Algorithms Portfolios Optimiser 79
5.11 Benchmarking 81
5.11.1 Single Resource Benchmark 81
5.11.2 Open Shop Scheduling Benchmark 84
a) Open Shop Scheduling Model 84
5.12 Integrated System Model for Scheduling Benchmark 85

5.12.1 Parameters Adjustment 85
5.12 Results of Single Machine Benchmarks 86
5.13 Results of Open Shop Benchmarks 91
5.14 Summary 94
Chapter 6 Heat Treatment of Metals 96
6.1 Introduction 96
6.2 Steel and its Heat Treatment Operation 96
6.2.1 Heat Treatment Operation of Steel 98
6.3 Furnace Model Design 100
6.4 Heat Treatment Model Design 107
6.4.1 Neuro-Fuzzy Model 107
6.4.2 Integrated PSO to NF System 116
6.5 Intelligent System for Heat Treatment Scheduling 118
6.5.1 The Mechanism of System 119
6.6 Summary 120
Chapter 7 Simulation Results 121
7.1 Introduction 121
7.2 Heating Treatment System Simulation Results 121
7.3 Intelligent System Results 125
7.3.1 System Results without Due Date Time 125
7.3.1.1 GA Results 125
7.3.1.2 AGA Results 126
7.3.1.3 GACD Results 126
7.3.1.4 SGA Results 127
7.3.1.5 MGA Results 127
7.3.1.6 AP Results 128
7.3.2 Results Discussion 128
7.3.3 System Results with Due Date Time 134
7.3.3.1 GA Results 135
7.3.3.2 AGA Results 135
7.3.3.3 GACD Results 136
7.3.3.4 SGA Results 136
7.3.3.5 MGA Results 137
7.3.3.6 AP Results 137
7.3.4 Results Discussion 138
7.4 Summary 142
Chapter 8 Conclusions and Future Work 143
8.1 Conclusions 143
8.2 Future Work 146
8.2.1 Optimisers 146

8.2.2 NF System Model 147
8.2.3 Furnace Model 148
8.2.4 Benchmarks 148
8.2.5 Real Time Implementation 148
References 150
Appendix A 162
Rule Bases 162
Appendix B 166
Experimental Data 166
Appendix C 171
Single Benchmark Generation 171

Chapter 1 Introduction and Outlines

1.1 Introduction

Intelligent Manufacturing is concerned with the application of Artificial Intelligence (AI) and Knowledge-based technologies in general to manufacturing problems. This includes a large number of technologies such as machine learning, expert systems, data mining and neuro computing. In addition it appears that these same technologies have so far proved more popular than Planning and Scheduling in such applications.

In this research, consideration of the state of the art in planning and scheduling with the emphasis on what technology is being used for applications is being investigated. An informal definition of what is meant by the terms planning and scheduling, has to be defined which should be accepted in the manufacturing community which is as follows:

Planning: the automatic or semi-automatic construction of a sequence of actions such that executing the actions is intended to move the state of the real world from
some initial state to a final state in which certain goals have been achieved [124].

This sequence is typically produced in partial order that is with only essential ordering relations between the actions, so that actions not so ordered appear in pseudo-parallel and can be executed in any order while still achieving the desired goals.

Scheduling: in the pure case, the organisation of a known sequence of actions or set of sequences along a time-line such that execution is carried out efficiently or possibly optimally. By exchanging, the allocation of a set of resources to such sequences of actions so that a set of efficiency or optimality conditions are met. Therefore scheduling can be seen as selecting among the various action sequences implicit in a partial-order plan in order to find the one that meets efficiency or optimality conditions and filling in all the resourcing detail to the point at which each action can be executed [16].

Heat treatment can be defined as the process that is used to alter the physical and mechanical properties of materials without changing the product shape by controlling heating and cooling rates. In the steel industry, determining the optimal heat treatment regime that is required to obtain the desired mechanical properties of the steel is considered as one of the hard and complex processes in the industry. This is because the search space of heating treatment regime is large and it is more complicated in relating between the inputs and their outputs. Therefore, it is important to develop a system that is capable of selecting the optimal heat treatment regime so the required metal properties can be achieved with the least energy consumption and the shortest time. Moreover, scheduling of heat treatment operations jobs are known to have a computationally demanding objective function, which could turn to be infeasible when large problems are considered. This has lead many researchers who have applied scheduling to heat treatment operations jobs latterly. This is because heat treatment scheduling problems have attracted the attention of researchers as a result to heavy jobs that consuming much energy and long time. In fact, an efficient algorithm that is able to solve heat treatment jobs scheduling problems is required.

In the field of artificial intelligence, Neuro-Fuzzy (NF) refers to combinations of artificial neural networks and fuzzy logic. An NF system is defined as fuzzy system which employs a self learning algorithm derived and inspired by neural network concept to achieve its fuzzy sets and fuzzy rules using processing data samples [105]. However, in this research using this methodology the model that is able to predict the heat treatment regime for different types of steel are developed and this model is integrated with an optimisation methodology [68] for determining the optimum heat treatment regime. Furthermore, the furnace model that provides all requirements accurately such as the amount of consumed time and the amount of consumed fuel are developed using standard data for optimisation purposes.

In this research, different types of intelligent optimisation methodologies are explored such as Algorithm Portfolio (AP), Particle Swarm Optimisation (PSO), Genetic Algorithm (GA) with different classic and advanced versions: GA with chromosome differentiation (GACD), Age GA (AGA), and Sexual GA (SGA), and finally a Mimetic GA (MGA), which is based on combining the GA as a global optimiser and the PSO as a local optimiser for the purpose of planning and scheduling with the emphasis on the application of the technology to heat treatment operation jobs scheduling. These algorithms have been tested using two types of benchmark. The results of these tests show that GAs are found to be time consuming but robust optimisation technique which can meet the requirements of manufacturing systems. GA is capable to handle real world problems because the genetic representation of precedence relations among operations fits the needs of real world constraints in production scheduling. Moreover, GA is applicable to a wide array of varying objectives and therefore they are open to many operational purposes.

The intelligent system has been performed using these algorithms; the heat treatment model and the furnace model for scheduling heat treatment operation jobs of steel and predict the optimal heat treatment regime for each job considering the common due date time.

1.2 Motivations

Scheduling is mostly classified as Non-deterministic Polynomial-time (NP-hard) problem which means essentially, very difficult to solve and is generally difficult type optimisation problem. Moreover, scheduling problems are considered to be so difficult that it may be unknown whether or not a problem even has an optimal solution [98]. Furthermore, heat treatment operation job scheduling problems for steel is highly requested in industry as a result to heavy job that consume much energy and long time. Scheduling these jobs is considered as hard process due to the search space is large and several constraints have to be applied. Therefore, an efficient algorithm that is able to solve heat treatment jobs scheduling problems is highly required where the time and fuel consumption can be optimised. In fact to develop such that algorithm, several models have to develop such as different types of optimisers, furnace model and heat treatment model.

This thesis presents intelligence algorithms that are capable to deal with heat treatment scheduling problems using genetic algorithm methodologies which are considered as robust adaptive optimisation techniques. Moreover, an effective algorithm based on algorithms portfolio methodology can be developed using different types of genetic algorithms techniques where the communication between these algorithms at early stages can be considered where the fast and less accurate algorithm can pass its results to the slow and more accurate algorithm, which will benefit from the good results at an early stage.

Furthermore, in the steel industry, determining the optimal heat treatment regime that is required to obtain the desired mechanical properties of steel is considered important so the optimal regime region can be set. Therefore, it is important to develop a system that is capable of selecting the optimal heat treatment regime so the required metal properties can be achieved with the least energy consumption and the shortest time. This system can be developed and achieved using NF and PSO methodologies based on experimental data.

1.3 Aims and Objectives

This research focuses on the steel industries requirements such as scheduling heat treatment jobs considering the due date time and the regime of job that enables to achieve the desired mechanical properties. In order to achieve a model that it can do all of these requirements, several models are exigency to develop such as an intelligent system that is capable of dealing with heat treatment operation jobs scheduling of steel where it can provide the jobs schedule that consume the least fuel and shortest time with considering the due date time.

In order to achieve an efficient intelligent system, several intelligent algorithms with different techniques are needed to develop and based on algorithms portfolio methodology these algorithms can be used by allowing the alternating the best solution between these algorithm at early stages. This will lead to improve the performance of the system. Indeed, to measure the capacity of each algorithm different types of benchmarks for testing these algorithms are required.

Furthermore, this system has to predict the optimal heat treatment regime to achieve the desired mechanical properties for different types of steel. In order to achieve such model, several requirements are needed.

- A model using intelligent techniques for predicting the heat treatment regime.
- Experimental heat treatment operation of steel data.
- Fast intelligence optimiser technique for finding the optimal regime.

In order to implement the intelligent system, furnace model is required. in order for this model to be satisfied for this work, there are some conditions to be met:

- It should be designed for optimisation proposes.
- It should be able to count the time and fuel consumption accurately.
- It should be developed based on standard data.

Finally, implementing the whole algorithm by integrating all the models that have been developed to form the intelligent system. After that measure the capacity of the system and improving its performance by determining the best techniques such as selecting the cost function method and types of crossover.

1.4 Challenges

- In real world applications, scheduling problems is much more complex because of for example, the varies constraints, set of objectives and the size of the search space may be involved in relation to different types of scheduling and to solve the problem hence becomes much more difficult.

- Develop a system that is capable to schedule the heat treatment operation jobs with least energy consumed and short time considering the due date time and it should be able to predict the optimal heat treatment regime for determining the desired steel mechanical properties.

- Heat treatment system that is able to provide the optimal heat treatment regime for different types of steel.

- Efficient furnace model that can be used for optimisation propose.

- Efficient algorithm that can solve the scheduling problems.

- Simple method for decoding the scheduling problems using artificial intelligent techniques.

- Develop a GA that can guarantee feasible solutions.

- Develop a suitable technique that is able to measure the quality of multi objectives.

1.5 Contributions

In this work, generic intelligent scheduling systems have been developed. This system includes a number of different intelligent techniques, such as Genetic Algorithms (GA) and its derivates namely Age Genetic Algorithm (AGA), Mimetic Genetic Algorithms (MGA), Genetic Algorithms with Chromosome Differentiations (GACD), Sexual Genetic Algorithms (SGA), Particle Swarm Optimisation (PSO), and hybridisations of the systems. Moreover, an effective algorithm has been developed using all previews algorithms based on algorithms portfolio methodology. In this algorithm, communication between different algorithms is considered at early stages, where the fast convergence and less accurate algorithm can pass its results to the slow and more accurate algorithm, which will benefit from the good results at an early stage.

Two different types of scheduling benchmarks are chosen to evaluate each algorithm. The results show that several new optimal solutions can be found using several different effective crossovers and scale operation method that is avoiding the system from generating unfeasible solutions.

A furnace model for optimisation proposes has been developed. This model was able to provide accurate measurement of the time and fuel consumption. Using Neuro Fuzzy (NF) and PSO methodologies heat treatment system model successfully has been developed and this model was able to predict the optimal heat treatment regime for different types of steel to achieve the desired mechanical properties.

The intelligent system that is able to schedule heat treatment operation jobs with least time and fuel consumption considering the due date time and is able to predict the optimal heat treatment regime for different types of steel to achieve the desired mechanical properties has been successfully developed.

This system deals with multi objectives and measure the fitness of each solution that is generated by the system, the system have used Pareto Front (PF) and Weighted Sum Method (WSM) techniques that their capacity and effect on the system have been illustrated.

1.6 Thesis Outline

This thesis contains 8 chapters, the first of which is an introduction that followed by motivation, aims and objectives, challenges, contribution and outline of thesis.

Chapter 2 presents critical appraisal literature review that is related to this research.

Chapter 3 presents the theoretical methodology of the most common intelligent techniques that have been used in optimisation applications.

Chapter 4 presents the concept of scheduling problems, its kinds, some scheduling benchmarks and the most efficient methods that have been used to solve scheduling problems.

Chapter 5 illustrates intelligent algorithms development methodology and the results of each algorithm, then they have been tested using two benchmark types.

Chapter 6 outlines the developments of the furnace model and heat treatment system model including combining the whole system and explanation of its mechanism.

Chapter 7 presents simulation results for each algorithms and technique that have been used in this work.

Chapter 8 concludes the research and outlines recommendation to future work.

Chapter 2 Literature Review

2.1 Introduction

Scheduling is the organization of a known sequence of actions or set of sequences along a time-line such that execution is carried out efficiently and/or possibly optimally. By extension the allocation of a set of resources to such sequences of actions in order that a set of efficiency or optimality conditions are met. It is well known that optimisation of scheduling problems is one of the hardest combination of optimisation problems [16] and turn out to be a promising area in research communities over the last three decades, creating vast amounts of literature being published. However, there are still a need for an efficient algorithm to be developed in order to solve problems optimally in shorter time [16]. The investigation of scheduling optimisation problem has started as early as in 1954, when Johnson [64] presented his work "Optimal two and three stage production schedules with setup times included". After which many application areas of scheduling were developed such as mixed and pure integer programming formulations, dynamic programming, and branch and bound methods. Moreover, heuristic algorithms were being applied for problems which were known to be computationally difficult. For example, in1989 Feldman and Golumbic compared the effectiveness and efficiency of heuristic algorithms on scheduling problems arising in schools, whereby ''tasks'' were classes and ''resources'' were teachers, classrooms, and students [40] as a result scheduling optimisation problems had many domains.

In 1989, Fischetti and his colleagues [41] dealt with the ''Bus Driver'' Scheduling Problem, where the objective was to minimise the number of drivers needed to perform all necessary duties and for them not to work more than a set of specified number of working hours on a daily basis. This problem was proved as non-deterministic polynomial-time hard (NP-hard) in the strong sense by the authors.

In this chapter, scheduling classification is explained and the literature reviews of Artificial Intelligent (AI) applications such as genetic algorithm and particle swarm optimisation to scheduling problems are considered. Furthermore, outline scheduling of heating treatment operations in the literature is given.

2.2 Scheduling Classification

Scheduling problems may be classified according to various schemes. A standard notation scheme proposed for scheduling problems by Graham et al [52] and Blazewicz et al [10]. In general, scheduling problems depend on four categories.

- Number of jobs and operation to be processed
- Number and types of machines or resources that comprise the shop
- Flow pattern and further technological and management constraints. Possible values are:
    - single machine
    - job shop
    - flow shop
    - open shop
    - permutation flow shop
    - machines in parallel
    - job shop with parallel machines at each stage
- Criteria to be optimised.

2.2.1 Dispatching Rules and Scheduling

The concept of dispatching rules which was based on a rule that the next job should be processed from a set of jobs was developed in the early 1960's [42] [43] [9]. Later, using dispatch rules became very common and large varieties of different rules have been applied to a range of different scheduling problems. Some rules are very simple such as earliest-due-date and first-in-first-out while some other rules are complex such as "closest due date whose customer's inventory is less than a specified amount". This has led to propose different kinds of procedures for selecting the dispatch rules best suited to a given case. In general, depending on the absence of difficulty to implement scheduling performance, dispatch rules have been chosen.

Dispatching rules can be classified as [55] :

- Process-time based rules
- Due-date based rules
- Combination rules and
- Rules that are neither process-time based nor due-date based.

The following, First Come, First served, Smallest Number of Remaining Operations, Largest Number of Remaining Operations, Shortest Processing Time, Job of Identical Setup, and Critical Ratio Scheduling, are considered as examples or types of process-time based rules. Earliest Due Date (EDD) is classified as an example of Due-date based rules. In general, the due-date based rules give good results under light load conditions, but the performance of these rules deteriorates under high load levels [102].

Process- time and due-date information such as Least Slack rule and Critical Ratio was proposed by Blackstone et al [9] . In this work, the combination between two types of rules was used which is classified as an example of Combination rules and moreover, the due-date based rules and the combination rules can be categorized based on whether the priority value changes over time or not. A dispatching rule is called a dynamic rule if the priority value calculated at a particular instant differs from the value calculated at a later instant. If the priority value once calculated remains the same throughout the presence of the job in the shop floor, such a rule is called a time-independent or static rule. The EDD rule falls into the latter category of a time independent rule. On the other hand, the least slack rule falls into the category of dynamic rules. Haupt [55] elaborate in his work rules such as total work-content of jobs in the queue for the next operation which do not related to any domain. Therefore his work can fall under rules that are neither process-time based nor due-date based category.

2.2.2 Genetic Algorithm

Artificial Intelligence (AI) is classified as a branch of Science which provides feasible solutions for helping machines to solve complex problems using emulation human-like fashion. In other word, the science and engineering of making intelligent machines [33]. During the last two decades AI techniques have been applied to various types of scheduling problems which clearly illustrates the increasing interest of researchers in this domain. This is largely due to advantages of AI over classic techniques such as traditional Operations Research techniques or dispatching rules in tackling the complexity of scheduling problems. Also, the flexibility offered by AI techniques in providing explanations and modifications are often essential for practical implementation of such systems.

Several AI algorithms such as Genetic Algorithm (GA) and Particle Swarm Optimisation (PSO) have been applied to different scheduling problems. The contributions that are incorporated in this research are limited to those of GA and PSO.

The development of GA by John Holland of the University of Michigan in 1965 [58] has been widely applied in many engineering fields such as production scheduling problems.

GAs have attracted the attention of many researchers, several searchers proposed GAs to solve the Job Scheduling Problems (JSP). In fact, GA faced some difficulties in this application such as representing JSP successfully, crossover operator ability to generate feasible schedules without losing their efficiency and its ability to converge the optimal solution [61].

2.2.2.1 Application of GA to Job Shop Scheduling Problems

Job Shop Scheduling Problem (JSSP) is a schedule planning for low volume systems with many variations in requirements. It is a schedule planning for a number of jobs on limited number of machines where each job has number of operation which needs to be processed without interruption on a given machine. JSSP aims to minimise the length of time intervals on machines by finding the optimal schedule.

In 1991 Falkenauer and Bouffouix [39] applied GA to JSSP. They used the encoding scheme to make GA deal with JSSP. For example, the string ABCD would encode solution where operation A is performed first, D second, followed by C and B. This encoding method led to modifying the crossover procedure. They used a cost function to estimate the quality of solutions. The results of this work showed three different size scheduling problem (one of them was taken from industrial partners) that are optimised using Least Slack Time, Shortest Processing Time and GA. Although not much optimisation had been done, GA had better result than other methods. In 1995 Croce et al [22] applied encoding based on preference rule. There was some improvement in results [93]. Moreover, there is an improvement in the performance of algorithm by using an updating technique based on classes of equivalent chromosomes and present look ahead evaluation.

Although there were some improvement in the experimental results after applying encoding based on preference rule [38] [22], researchers still face difficulties in using GA for JSSP. However for enhancing the quality of GA applied to JSSP, local search technique was introduced such as neighbourhood search [103]. A good example for using local search technique and getting more improvement is Yamada and Nakano [127] [93]. In this work a new crossover proposed called Multi-Step Crossover Fusion (MSXF) to deal with local search and GA together. The results of the new GA which has modified in its crossover and included to the local search were optimal solutions for five jobs problems while near optimal solutions for ten jobs problems. Muth and Thompson benchmark [91] were used in this work. Stochastic Job shop Scheduling Problem (SJSSP) is considered as more realistic scheduling problem than JSSP in the real world [132]. Yoshitomi [132] introduced an approach for solving Stochastic Job Shop Scheduling Problems (SJSSP) using GA. The author modified GA to deal with SJSSP, where the fitness function was regarded as a fluctuation that may occur under stochastic circumstances specified by the distributions of stochastic variables. Moreover, the Roulette strategy was used for selecting the optimum solution in terms of the expected value where each individual has a number of frequencies which are used during selection operations. The experimental results demonstrate the success of this method as compared to the stochastic job shop problem.

Tsujimura et al [121] introduced a GA with symbiotic mechanism. The idea of symbiotic mechanism is to enhance GA by two types of processes:

- Total machine idle time schedule criteria are formed as the fitness function which processed by a co-evolution.

- The fitness function used the total job of waiting time schedule criterion to provide high diversity for chromosome population which is processed by a sub-evolution.

This process was carried out in order to provide high diversity to the chromosome population by employing the total job waiting time as the evaluation function. This proposal has been applied to Fisher and Thompson benchmark [42] 10 job-10 machine JSPP (10×10 FT) and 20 job-5 machines JSSP (20×5 FT) [42]. Experimental results show that the proposal algorithm is able to find good results around the optimal solution and sometimes the optimal solution.

Other researchers have hybridisation methods for job-shop scheduling problem which are classified into the following three categories [16]

- adaptive genetic operators
- heuristic-featured genetic operators
- hybrid genetic algorithms

The first approach is to revise or invent genetic operators so as to meet the nature of a given encoding representation. The second approach is to create new genetic operators inspired from conventional heuristics while the third approach involves hybridising conventional heuristics into the main loop of GA.

For developing hybrid framework Wang and Zheng [123][121] proposed combining GA and Simulated Annealing (SA) where GA to present parallel search architecture and SA to increase escaping probability from local optima at high temperatures and perform fine neighbour search at low temperatures. This hybrid strategy was applied to JSSP. Although, it shows that it is very effective and robust but it could not get the optimal solutions.

Liu et al [83] proposed Hybrid Taguchi-genetic algorithm to solve job-shop scheduling problem (JSSP). This approach combines GA with Taguchi method which is inserted between crossover and mutation operations of the GA. This work aims to get the advantages and ability of the Taguchi method which incorporated in the crossover operations to select the better genes for the crossover operator and consequently enhance the genetic algorithm. This proposal is tested on Fisher and Thompson's bench mark 10 job-10 machines JSP (10×10 FT) and 20 job-5 machines JSP (20×5 FT) [42]. Experimental results show improvement in quality compared to [121] who applied the same benchmarks. Moreover, the comparison shows that the hybrid Taguchi-genetic algorithm is more robust and able to find better result.

2.2.2.2 Application of GA to Flowshop Scheduling Problems

Flowshop scheduling problems (FSSP) deals with processing a set number of jobs through a set number of machines, where all jobs have to pass among machines in the same order. As FSSP is known as a combinatorial problem and conventional algorithms cannot be solved to guaranteed optimality. Therefore it is classified as NP-hard problem [67]. However, different approaches have been applied such as GA. In fact, much improvement found when GA is used to solve FSSP. For examples Yin et al [130] used GA after changing probability of crossover and mutation from fixed rate to dynamic rate according to the fitness value of the chromosomes where if the fitness value of the chromosome is higher, the probability of crossover and mutation will be lower. This proposal was tested to three scheduling problems which are taken from [117]. The result of this test shows that this GA performance gives effectiveness, efficiency and better result compared with traditional GA. Ponnambalam et al [99] used Travelling Salesman Problem (TSP) to improve GA where the initial solution was generated by TSP. This proposal applied to flow shop scheduling with multi objectives. Three objectives were used and combined into a single objective according to a fitness scalar. Each objective has a random weight and the sum of all the weights equal one. In this work TSP helps GA to get the optimal solution in the shortest time. This is due to the fact that the best solution is close to initial solution.

In other work, a multi-objective evolutionary search algorithm using genetic algorithm for scheduling of the mixed-model assembly line was proposed by Yu et al [133]. In this work, according to the mathematical model of Pareto [67] the ranking method was adopted for multi-objective GA. Moreover, the distance-dispersed approach was used to evaluate the fitness of the individuals. Finally, the results show that this proposed algorithm was quite effective.

2.2.2.3 Applications of GA to Open Shop Scheduling Problem

Open Shop Scheduling Problem was first presented by Conway et al [21] and later by Colemman et al [19]. OSSP has the same rules as job shop scheduling problem except no ordering constraints on the operations while job shop scheduling problem all the operations of a job are ordered.

Major application areas include but not limited to Basic Chromosome Representation, Directly Encoding the Operation, Fixed Heuristic Choice and Evolving Heuristic Choice have been used to improve GA by Fang et al [38] in 1994. The authors applied this proposal to well known benchmarks [111]. The result shows that for 4×4, 5×5 and 7×7 bench marks, at least one of the evolving heuristic choice methods able to find the optimal while other proposals could find good solutions close to the optimal. On the other hand, for 10×10, 15×15 and 20×20 benchmarks, all the proposals could not find the optimal solutions except Fixed Heuristic Choice could find the optimal result for benchmark of 15×15. In 1999 Khuri et al [70] developed three types of GA, consecutively Permutation Genetic Algorithm (PGA), Hybrid Genetic Algorithm (HGA) and Selfish Genetic Algorithm (FGA). In this work, Taillard benchmarks have been used to test each method. The results of this work in general close to optimal solutions and sometimes able to find the optimal results but it can be notice that PGA and FGA were more effectively in 4×4 and 5×5 benchmarks while HGA was more effectively in 7×7and 10×10 benchmarks. Moreover, in 7×7 and 10×10 benchmarks, no any proposal could find any optimal result. Hybridization strategies aim to enhance the performance of GA. However, Liaw [79] introduced a local improvement procedure which depends on Tabu Search for solving the OSSP that has been applied to improve the performance of GA. The result of this algorithm after testing shows that this proposal works effectively. For example some of the benchmark problems in [15] are solved to optimality such as 5×5 and 6×6 benchmarks while 7×7 benchmarks; proposer just could find solutions which are near optimal solution in most cases. Moreover, authors mention that many crossovers has been tested such as partially mapped crossover, order crossover, cycle crossover, order based crossover and position-based crossover, and found that linear order crossover works best for the problem under consideration. Lowa et al [85] applied three hybrids genetic heuristics, double genetic algorithm, simulated annealing genetic algorithm and tabu search. In this work, double genetic algorithm found the best performance between these three hybrids genetic heuristics.

2.2.2.4 Applications of GA to Single Machine Scheduling Problem

In Single Machine Scheduling Problems (SMSP), there are a number of jobs or operations that need to be processed on single machine. These jobs or operations are required to order by schedule. An optimiser is looking for the best schedules which can give an optimal result usually minimising total earliness/tardiness cost. Early and tardy penalty rates are allowed to be arbitrary for each job. This problem is known to be NP-hard, even for the case of a single family and a single penalty rate per job that is used for assessing both earliness and tardiness cost [54].

Miller et al [89] introduced a HGA to single machine scheduling problem with sequence dependent setup times and due dates. They used Standard GA, Wagner-Whitin algorithm and TSP to develop HGA where, GA is used as global searcher to find optimal solution area. While a Wagner-Whitin algorithm is used as a local searcher. TSP is considered to order jobs in each period, so that the best schedule in each period can finally be obtained. Real life data has been used to test HGA. The result of this test shows that HGA was much better than Just-In-Time heuristic and faster than the standard GA to cover the optimal result. Liu and Tang [82] presented genetic algorithm to single machine scheduling with ready times. Filtering and cultivation steps have been added to structure of GA where filtering step comes after selection of solutions for reproduction. Filtering operation will replaces two worst solutions with the best one recorded and the best in the current generation. Cultivation step is added after measuring fitness of each solution in the new generation. The aim of this step is to monitors the groups of successive generations for which the best solution found so far has not been changed. This modification improved the quality of GA as compared to classic GA. Nazif and Lee [94] developed a GA with optimised crossover (OCGA) to solve a single machine family scheduling problem. Using an undirected bipartite graph, optimised crossover has been developed to determine an optimal schedule which minimise the total weighted completion time of the jobs in the presence of the sequence independent family setup times. The aim of this crossover is to keep an optimal solution which may be lost when classic crossover is used as result of incorrect choices being made by stochastic operation. The simulation results show that OCGA performs better than the standard GA and Multi Crossover Genetic Algorithm (MXGA). Anew approach to single machine scheduling has been introduced by Duenas et al [31] where they considered a multi-objective problem with fuzzy due dates. The algorithm is tested and compared to GA with local search. The result of this comparison shows that this algorithm performs better than the genetic algorithm with local search.

In GA literature researchers used various techniques to represent individuals. Furthermore, different crossovers and mutations have been proposed for solving scheduling problems. Table 2.1 shows the summary of the representation methods, crossover and mutation operators which are mentioned in the literature review.

Table 2.1 Literature Review for the papers.

Source	Individuals Representation	Crossover Operator	Mutation Operator	Object Function
Falkenauer and Bouffouix [39]	Symbolic	Symbolic Crossover	Symbolic Mutation	JSSP
Croce et al [22]	Symbolic	Linear Order Crossover (LOX)	Swap Mutation	JSSP
Nakano and Yamada [93]	Binary	Conventional Crossover	Conventional Mutation	JSSP
Liu et al [83]	Symbolic	Crossover Using Swap-Change & Taguchi Methods	Swap-Change Mutation	JSSP
Yamada and Nakano [127]	Symbolic	Multi-Step Crossover Fu-sion (MSXF)	Multi-Step Mutation	JSSP
Tsujimura et al [121]	Symbolic	Partial Schedule Crossover PSX	Neighbour Search Mutation	JSSP
Yu et al [133]	Real Number Coding	Order Crossover	Inverse Mutation	Multi Objective Scheduling
Ponnambalam et al [99]	Real Number Coding	Partially Mapped Crossover	Reciprocal Exchange Mutation	FSSP
Yin et al [130]	Real Number Coding	Two-Point Crossover with Dynamic Rate	Swap Mutation with dynamic rate	FSSP
Liaw [79]	Permutation Representation	Linear Order Crossover LOX	Insertion & Swap Mutation	OSSP
Lowa and Yeha [85]	Permutation Representation	Crossover Using Concept of Mating	Heuristic Mutation	OSSP
Nazif and Lee [94]	Binary	Optimised Crossover	Binary Mutation	SMSP
Miller et al [89]	Real Number Coding,	Two-Point Ring-Like Crossover	Constrained Mutation	SMSP

2.2.3 Application of Swarm Optimisation to Scheduling Problems

Swarm optimisation such as Ant Colony Optimisation (ACO), Particle Swarm Optimisation and Bee Swarm Optimisation (BSO) has been applied widely to scheduling problems. In this literature review, several proposals are considered.

2.2.3.1 Application of Ant Colony Optimisation to Scheduling Problems

ACO takes inspiration from the foraging behaviour of some ant species. It was introduced in the early 1990s by Dorigo [28]. ACO considered as a metaheuristic approach to deal with hard combinatorial optimisation problems such as scheduling problems.

After Dorigo [28] presented ACO, several other ACO algorithms which have the same characteristic have been developed for various applications but ACO's application to scheduling problems takes a few years to appear in the field of scheduling problems. However, in 2002 ACO has been applied to industrial scheduling problem using a multiple objective ant colony optimisation metaheuristic by Gravel et al [53]. In this work an efficient representation of continues horizontal aluminium casting process has been presented and simulated. Merkle et al [87] presented ACO to the Resource-Constrained Project Scheduling Problem. The benchmark problems in [87] [71] have been used and a combination of local and summation global pheromone evaluation methods was used by the ants for the construction of a new solution. Moreover, they alleged the changing strength of heuristic influence, the changing rate of pheromone evaporation over the ant generations and the restricted influence of the elitist solution by forgetting it at regular intervals. In this work two local searches were used to improve the performance of ACO. The result of this work has been compared with the results of various other randomised heuristics for the Resource-Constrained Project Scheduling Problem including genetic algorithms and simulated annealing on the set of largest instances in the benchmark [87] [71]. The result of this comparison shows that this algorithm work well and more flexible to implement restrictions to the number of evaluated schedules or without this restriction. Moreover, in this work 130 new best solutions have been found from 396 instances in the benchmark set.

Applications of PCO to non robustness heuristic scheduling problem proposed in 2003 by Ying and Lia [128]. The author used ACO to solve a single machine total weighted tardiness problems which was proposed by McNaughton [129]. In this work although there had been an improvement in the robustness and quality solutions, there is high probability to produce solutions with poor quality. Heinonen et al [56] studied the visibility of applying a hybrid ant colony optimisation algorithm to job-shop scheduling problem [91]. This study concluded to that the ACO performance is an easy algorithm to implement and very good fast solution can be found. In other words, it is fast in local search.

2.2.3.2 Application of Bee Colony Optimisation to Scheduling Problems

Chong et al [17] implemented honey bees foraging for solving job shop scheduling. Two major characteristics of the bee colony were mapped to be applied for job shop scheduling waggle dance and forage (or nectar exploration). However, the performance of this approach had been compared and evaluated with ant colony algorithm and tabu search algorithm. In fact, the experimental result showed that there was a gap between tabu research and the other two, where the tabu search could record the closest results to the best known solutions and had the most number of best solutions. Furthermore, it also managed to achieve the best results in the shortest execution time possible. On the other hand, bee colony could get a slight better performance and more number of best solutions than ant colony while the execution time was almost equal.

2.2.3.3 Application of Particle Swarm Optimisation to Scheduling Problems

Particle swarm optimisation (PSO) has been first presented by Kennedy and Eberhart [68] in 1995. It was simulated as a social behaviour of bird flocking or fish schooling to be used as an optimiser.

PSO has been applied to a great variety such as optimisation problems, artificial neural network training, pattern recognition, fuzzy control [32] [120] [7], continuous nonlinear functions [68], nonlinear constrained optimisation problems [31] and some other fields. Moreover, PSO has been growing rapidly with over 100 published papers every year.

All the related research totalling over 300 papers prior to 2004 [59] [78]. On the other hand, the number of the applications of PSO to scheduling problems is extremely low [78].

In 1989, Shaw and Whinston [110] proposed a PSO approach to scheduling of flexible manufacturing systems. Researchers, who have studied the nonlinear software problems adopting the PSO technique, usually believe that the parameters w, r1 and r2 are the key factors to affect the convergence of the PSO [18] [120] [92] and were r1 and r2 are chosen randomly, PSO cannot guarantee the optimisation's quality. Chuanwen and Bompard [18] provided a new method that introduces chaotic mapping with certainty, ergodicity and the stochastic property into particle swarm optimisation so as to improve the global convergence. This technique used to solve the short term generation scheduling of a hydro-system in a deregulated environment. The result introduced chaos mapping and an adaptive scaling term into the particle swarm optimisation algorithm, which increases its convergence rate.

Jerald et al [62] used four techniques GA, PSO, Mimetic Algorithm and Simulated Annealing for scheduling optimisation. They applied these techniques to Flexible Manufacturing Systems (FMS). The FMS contain five flexible machining cells each with two to six Computerised Numerical Control machines an independent and a self-sufficient tool magazine, one Automatic Tool Changer and one automatic pallet changer. Each cell is supported by one to three dedicated robots for intra-cell movement of materials between operations. The objective of the schedule is to minimise the machine ideal time and minimising the total penalty cost. Results of this works show that Particle swarm algorithm is found to be superior and gives the minimum combined objective function.

It is well known that the original PSO is designed as continuous technique. For solving discrete optimisation problems first, Kennedy and Eberhar [69] developed a discrete binary version of the PSO. There were two main differences the first is in the particle which was composed as binary variable while the second is in the velocity which is changed where it's probability having to be changed to give binary variable one value. Consequently, many researches came to solve discrete optimisation problems. Few research concentrate on scheduling using discrete PSO. Liaoa et al [78] applied this approach of discrete PSO in flow shop scheduling problems and found that the discrete PSO algorithm performs better than the continuous PSO algorithm of Tasgetiren et al. [114]. Moreover, compared to GA, PSO gave better performance when it was allowed to run on the same time. In this work they used local search scheme to incorporate PSO. They called it PSO-LS. The result shows how it is guided successfully by PSO. Although it take more time than PSO, it performs better for some problems such as Max–Min Ant System and Posed Ant Colony Algorithm which are the latest versions of ant-colony algorithms which have been applied to the scheduling problem for nearly 10 years [78]. Another research used Discrete Particle Swarm Optimisation for scheduling made by Pana et al [96] . The researchers in this paper applied standard and Variable Neighbourhood Descent to the no-wait flowshop scheduling problems. 110 benchmark instances of Taillard [117] are treated as no-wait flowshop problems and applied on both algorithms. The computational results show that the proposed algorithms generated either competitive or better results than those by the descent and Travailing Salesman algorithms of Grabowski and Pempera [51]. Moreover, the variable neighbourhood local search in the PSO algorithm enhanced the solution quality significantly.

Sha and Lin [109] proposed PSO for multi objective optimisation problems. In this work, diversification strategy has been developed to prevent PSO sticking to local optimal solutions. This strategy has been applied based on the following set of rules.

- If the solution of the particle dominates the gbest solution, assign the particle solution to the gbest.
- If the solution of the particle equals to any solution in the non-dominated solution set, replace the non-dominated solution with the particle solution.
- If the solution of the particle is dominated by the worst non-dominated solution and not equal to any non-dominated solution, set the worst non-dominated solution equal to the particle solution

Also, in this work PSO was modified by changing representation of particle position, particle movement, and particle velocity using priority value for global best and local best. Moreover, using mutation operator PSO performance has been improved. This proposal has been tested by Taillard’s benchmark [111]. The results show that Multi-

Objective PSO gave better results all the 23 problems than the Multi-Objective GA whose fitness function was weighted by sum of makespan, total tardiness, and total idle time of machines with random weights. In 22 of the 23 problems, the proposed PSO performed better for the solution considering the total tardiness. Finally, it can be said that the proposed algorithm was superior to the GA in solving the JSP with multiple objectives.

2.2.4 Application of Scheduling to Heating Treatment Operations

Recently many researchers have applied scheduling to heat treatment operations jobs. This is because deterministic manufacturing batch scheduling problems has attracted the attention of researchers as a result to heavy product which forced a gradual shift from continues manufacturing to batch manufacturing [104]. In fact, researcher has been looking for an efficient algorithm to solve scheduling problems of a batch processing machine.

The earliest work in the deterministic scheduling of batch processors has been presented in 1986 by Ikura and Gimple [60]. Authors introduced the phrases or terms of scheduling of batch. In this work, the problem of determining whether a schedule in which all the due dates can be met when release times and due dates are acceptable found has been addressed.

Scheduling a batch processing machine with incompatible job can be used in modeling a large number of heat treatment families to minimise number of tardy jobs. Jolai [65] used a dynamic programming algorithm to solve this problem and this problem has been proved as NP-hard when the number of family and the machine capacity are arbitrary. Monch et al [90] proposed two different decomposition approaches based on GA and simple heuristic algorithms to scheduling of parallel batch machines with unequal ready times of the jobs. This problem is found in the diffusion and oxidation areas of semiconductor wafer fabrication, where the machines can be modelled as parallel batch processor. Mathirajan et al [86] proposed problem of task scheduling with the use of parallel, non-identical initial processes in the presence of dynamic job arrivals, non-compatible task series and non-equal task quantities. Scheduling operations in heat treatment processes has an important meaning for the steel casting production times. In order to keep under control the amount of performed calculations in standard mathematical models, several heuristic algorithms were developed, which enabled obtaining suboptimal solutions in the opportunity of utilisation of furnaces in a heat treatment operations. Scheduling of production orders has been applied in a steel plant in order to find a particular technological operation start and finish times with efficiency limitations, with a purpose to minimize the sum of weighted times of all orders' finishing [113]. This problem was performed as a question of mixed integer programming. To solve the problem the methodology of solving the distribution composed of Lagrange relaxation, linear programming and heuristics was used.

2.3 Summary

The overall this chapter, scheduling problems such as JSSP, SMSP and OSSP have been proved as NP-hard problems. Therefore, different techniques have been proposed to solve scheduling problems such as dispatching rules, GA, PSO and ACO. Although there was a good improvement using GAs to solve different scheduling problems, GA's are not guaranteed to find the global optimum solution. In order to improve the performance of GAs, several techniques have been used such local searcher and different type of crossovers such as LOX, COX and PSX. In fact, these crossovers are designed for only scheduling problem to avoid unfeasible solutions and do not mimic the classical crossover of GA. Moreover, the encoding and decoding played main role in improving the performance of GAs.

Scheduling of heat treatment operation jobs is considered as one of the hardest scheduling problems this is due to several requirements required to adjust heat treatment operation and is handle with multi objectives. In fact this has lead to more complexity in modelling and scheduling of heat treatment operation. In summary, all the mentioned previews did not consider the heat treatment regime during scheduling and deal with job schedules or patching schedules.

Chapter 3 Intelligent Optimisation Techniques

3.1 Introduction

Optimisation can be defined as a series of actions aimed at finding the best possible solutions for any given problem(s). The simplest example of optimisation is the minimisation or maximisation of real functions by choosing the values of real or integer variables from within a possible set. Therefore, optimisation techniques are classified in the first instance on the number of objective functions to be achieved, from being either a Single Objective Function or Multi Objective Functions. Furthermore, they can also be classified into two basic classes being either Deterministic Algorithm, in which all variables are deterministic or Probabilistic or Stochastic Algorithm whereby some or all the parameters are probabilistic.

Classification based on the existence of constraints can be applied when constrained optimisation problems are subject to one or more constraints, while unconstrained optimisation problems are those without constraints. Taking manufacturing optimisation techniques as an example, these can be divided into two classes [107]. The traditional optimisation techniques entail using direct search and the gradient search methods while intelligent optimisation techniques employ techniques such as using variants of genetic algorithms, simulated annealing, particle swarm optimisation, ant colony system and tabu search. Recently, intelligent optimisation techniques have been applied successfully to solve a great variety of optimisation problems.

In this chapter, different types of intelligent optimization methodologies will be presented with direct focus on Genetic Algorithm (GA) [48] and its derivations such as Sexual Genetic (SGA) [79], Age Genetic Algorithm (AGA) [46], Genetic Algorithm using Chromosome Differentiations (GACD) [5] and Mimetic Algorithms (MA) [24]. An illustration of Particle Swarm Optimisation (PSO) [68] is given in details. Furthermore, an algorithm portfolio is presented which consists of a collection of different algorithms or different copies of the same algorithm running on different processors [50].

3.2 Genetic Algorithm

GA was developed by Holland in 1965 [58] based on the simulation principle of natural genetics. It is an exploratory search and optimisation method applied in computing in order to determine the optimal solutions for an optimisation or search problem. This algorithm has been widely used to solve manufacturing problems. The reason for GA's wide acceptance and application is due to its advantages over traditional methods, some of which are:

- It does not need many requirements to implement such as derivative information or auxiliary knowledge
- The algorithm works effectively as a global optimiser
- During the search population points can be executed in parallel rather than on a single point
- The algorithm use probabilistic transition rules as oppose to deterministic ones

3.2.1 Implementation of GA

In genetic algorithm, the problem is encoded in a series of bit strings which are in most cases manipulated by the algorithm; in an evolutionary algorithm, the decision variables and problem functions are used directly. In this section, an illustration of the basic steps in a GA is given.

3.2.1.1 Initialisation and Representation

Initially, many chromosomes are randomly generated to form an initial population with each chromosome representing different solution to the problem. Each bit or group of bits in this chromosome capture some characteristic of the solution. The length of the chromosome's string is usually determined according to the accuracy of the object function.

There exist many strategies to represent chromosomes such as integer and real-valued representation with binary coded approach being the most commonly employed representation approach. Figure 3.1 shows 3 variables have been presented in binary code.

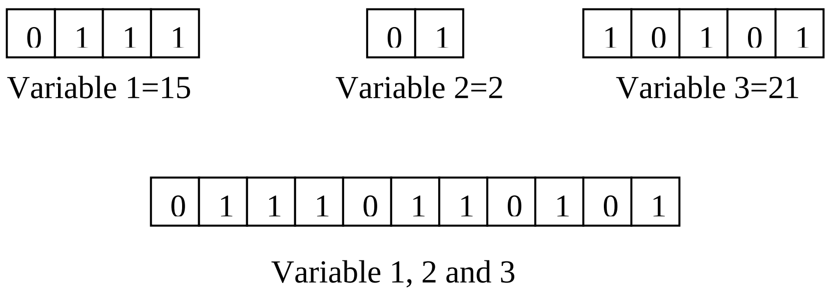


Figure 3.1 Representation of three variables in a single chromosome.

Using real value representation provide some benefits such as the requirement of less memory, greater freedom in applying different genetic operators and the elimination of the conversion step from chromosomes to phenotypes before the evaluation of each function [126].

3.2.1.2 The Fitness Function

The objective function is used to test each chromosome. The aim of this operation is to know the quality of each chromosome representation and determine the best one in this generation. The fitness score is given to each chromosome and the quality of each individual is measured using this fitness score. In order to evaluate each chromosome, there are many techniques [135] have been used.

- Root Mean Square (RMS) is used to measure the quality of the chromosomes as Equation 3.1.

$$RMS = \sqrt{\left(\frac{\max - \min}{2}\right)^2} \qquad \text{……………..………………………………………... (3.1)}$$

- Pareto Front (PF) Pareto Front method is applied using the concept of dominated and non-dominated solutions where the solution has to be non-dominated by another solution for all objective function until added to non-dominated set. This non-dominated set is called “pareto optimal” or “pareto front” when the plot of the objective functions whose non-dominated is applied.

Unfortunately, pareto front always does not gives a single solution, but rather a set of solutions.

- Weighted Sum Method (WSM) see Equation 3.2 have been known as common techniques for measuring multi objective fitness function.

$$WSM = \frac{scale\ of\ \ f_1 \times weight\ coefficient\ + \cdots +\ scale\ of\ \ f_n \times weight\ coefficient}{n} \ldots\ldots (3.2)$$

where f_n refer to the object function and n refers to number of objectives while WSM refers to weighted sum method.

3.2.1.3 The Reproduction Operators

The reproduction operations aim to achieve better generations. Several operators such as crossover and mutation are used to create a new generation in the reproduction stage. Brief discussions of operators are presented, namely: crossover, mutation, selection, elitism, and termination [107].

3.2.1.4 Selection Operators

After the completion of measuring the fitness function, the next step is the selection process with the aim of determining or choosing individuals in the recent generation that will create offspring in the next generation. The process of selection works to emphasise fitter individuals which will lead to achieve better individuals with higher fitness.

In order to achieve good selection the selection process must be adjusted. A good selection is able to determine highly fit individuals over the generation whereas poor selection leads to result of slow evolution. Furthermore, the design of the selection operator has to take into account crossover and mutation processes which may be in conflict with each and thus leading to lose the diversity of the generation [107] .

a) Roulette Wheel Selection

Roulette wheel selection is a technique which used to do selection operation [3]. In this technique, stochastic selection is applied using a random number. The individual whose segment spans the random number is selected. The process is repeated until the desired number of individuals is obtained. These leads to the fittest individuals have a greater chance of survival than weaker ones.

b) Stochastic Universal Sampling

Stochastic universal sampling has been applied widely as a result to its advantage over roulette wheel selection where zero bias and minimum spread has been provided by stochastic universal sampling [3]. This is because a fixed random number used to make each individual has an equal space according to fitness in roulette wheel selection. In this technique, a noise in roulette wheel selection technique has been prevented.

c) Truncation Selection

Truncation selection is considered as an artificial selection method. In this selection technique, individuals has been rearranged their order according to their fitness and the best individuals has been chosen for parents. Moreover, these selected parents produce uniform at random offspring. Truncation selection is used usually for mass selection [11].

d) Tournament Selection

Tournament selection is a selection technique which employs a random selection operator for choosing a tour’s number of individuals. The fittest individual in the generation is chosen to be parent. This operation repeated until uniform of offspring complete.

e) Local Neighbourhood Selection

Local neighbourhood selection method is taken from local neighbourhood technique [122] . Unlike other selection methods, Local neighbourhood selection based on exact region or place of individuals, the individuals have been interacted. The size of the neighbourhood can be determined by the structure of individuals and the distance between the places of individuals. Local neighbourhood selection operator is using the size of neighbourhood and random numbers to operate selection operation.

f) Ranking selection

In 1985 Baker [4] introduced rank selection technique. This method depends on measuring each individual according to their fitness value and thus allowing for the generation order to be rearranged. The target of this operation is to remove the less fit individuals from the new generation and keeping the highly fit individuals in the new generation.

g) Steady State Selection

In steady state selection method, part of the fittest individuals is kept at each generation for a new generation. While the less fit individuals are removed and replaced by new individuals. The new generation will be a companion between fittest individuals of previous generation which selected by steady state selection and the new individuals which are generated.

3.2.1.5 Crossover Operator

Crossover is used to generate new generation population of individuals from those selected in recent generation population by selection operator. Crossover is considered as the main operator of GA where the performance of GA highly depends on it [107]. To this end, there exists an abundant of crossover operators proposed in GA literature with the objective to improve its performance. Combining the features of two existing individuals is the main idea of all crossover purposes.

Crossover operator start working, when two chromosomes are taken randomly and the operator choose a random crossing site between chromosomes, and in the case of multi crossover more than one crossing site are chosen. Thereafter, some genes in each chromosome are exchanged using the crossover operator. For illustrative purpose, single and multi crossovers are taken as an example to explain the mechanics of crossover operation.

a) Single point crossover

In single point crossover, the crossover operator generates new chromosomes called offspring from the parents by taking the first part from one parent and the second from the other. This is an example of single point crossover when the crossing site is chosen at forth gene [107].

Before the crossover operation

Parent1	1111 1111
Parent2	0000 0000

After the crossover operation

Child1	1111 0000
Child2	0000 1111

b) Multi point crossover

In the multi point crossover two cutting sites or more are chosen and children are obtained from parents by exchanging the part located between these cutting sites [107].

Below is an example of multi point crossover when the crossing site has been chosen after each two genes.

Before the crossover operation

Parent1	11 11 11 11
Parent2	00 00 00 00

After the crossover operation

Child1	11 00 11 00
Child2	00 11 00 11

3.2.1.6 Mutation Operator

Mutation operator is working randomly using mimic or simulate the phenotype of mutation [58]. The operator alters one or more gene in chromosome by replacing another gene. The aim of this operation is to prevent the generation from sticking at any local optimal area in the search space. Therefore, mutation operator is used only when the probability of the operator satisfying and this probability always are very low.

3.2.1.7 Elitism

Elitism operation had been proposed Jong in 1975 [66]. Later, various techniques have been used to keep the local optimal with the population of GA at each generation. This is because the local optimal may be lost during generating the new generation and as a result of the operations of selection, crossover and mutation.

3.2.1.8 Termination

In GA, the process of each operator such as selection, crossover, mutation, ranking and elitism is repeated until a termination condition has been satisfied. In order to explain the terminating conditions some selected examples of different termination criteria are given below such as:

An optimal solution has been found.

- Number of iteration has reached the maximum iteration number.
- The computation time reached the maximum time for the whole processes.

GA has several operators which have various different techniques. In other words, the performance of GA is very different from that of most traditional optimisation techniques. Figures 3.2 and 3.3 show that the working principles of GA.

```
Begin
  Generating population randomly
      While termination condition is not satisfied
            Measuring population fitness
              Loop
                    Reproduction
                    Crossover
                    Mutation
              End loop

      End while If termination condition is satisfied
End
```

Figure 3.2 Pseudo code of GA.

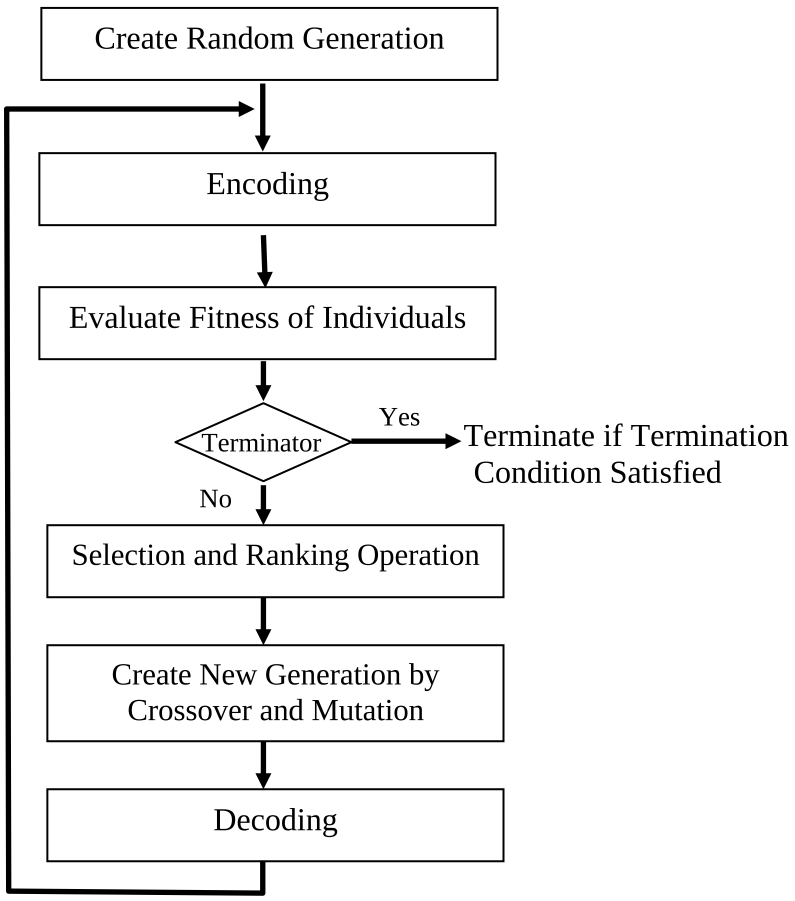

Create Random Generation
Encoding
Evaluate Fitness of Individuals
Terminator
Yes
Terminate if Termination Condition Satisfied
No
Selection and Ranking Operation
Create New Generation by Crossover and Mutation
Decoding

Figure 3.3 Flow chart of classic GA.

3.3 Sexual Genetic Algorithm

SGA is an optimisation technique which is derived from the traditional GA. In other word SGA is an improved version of GA. In classic GA, the selection operation chooses parent chromosomes for reproduction using only one selection strategy. When taking into consideration the model of sexual selection in the area of population genetics it becomes obvious that the process of selection mating partners in natural populations is different for male and female individuals. Inspired by the idea of male effort and female choice, Lis and Eiben [80] have developed sexual GA that utilises two different selection strategies for the selection of two parents required for the crossover. The first kind of selection scheme utilises random selection and another selection strategy uses roulette wheel for the selection of two parents. The remaining steps are similar to those of GA [79].

In order to explain the performance and working principles of SGA Figures 3.4 and 3.5 are given.

```
Begin
  Generating population randomly
      While termination condition is not satisfied
            Evolution population fitness
            Divided the individuals to males and females
            Loop
                Reproduction using idea of
                male effort and female choice

                Crossover
                Mutation

            End loop
      End while when termination condition is satisfied
End
```

Figure 3.4 Pseudo code of SGA.

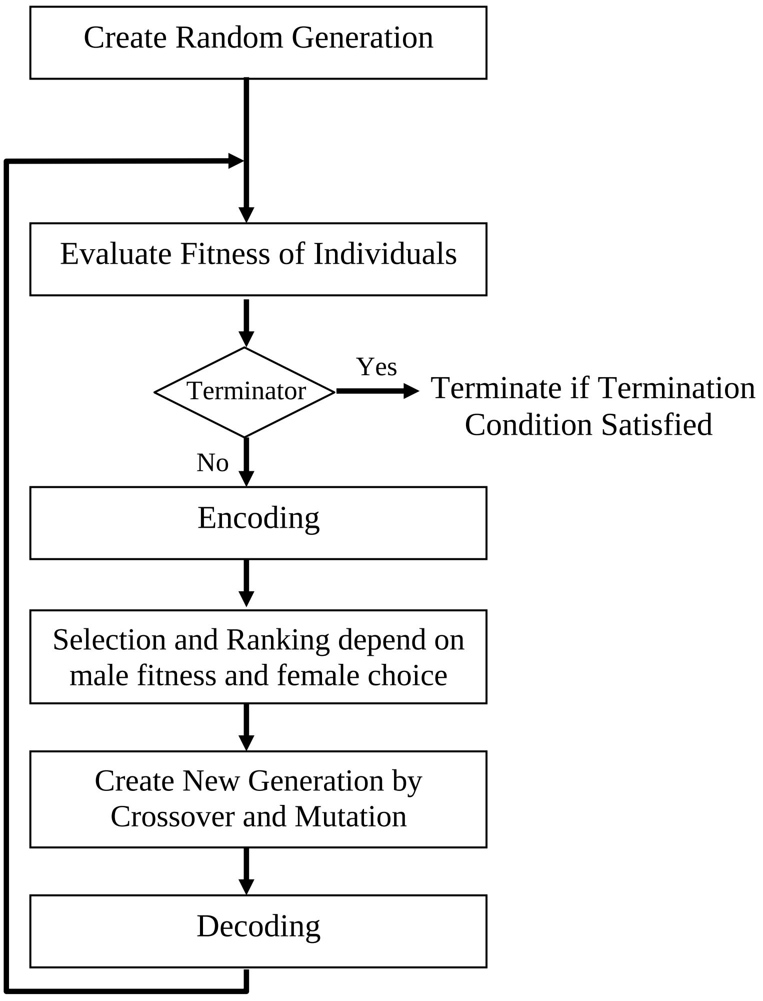

Create Random Generation
Evaluate Fitness of Individuals
Terminator
Yes
Terminate if Termination Condition Satisfied
No
Encoding
Selection and Ranking depend on male fitness and female choice
Create New Generation by Crossover and Mutation
Decoding

Figure 3.5 Flow chart of SGA.

3.4 Age Genetic Algorithm

AGA can be defined as a modified version of traditional GA. This modification aims to improve the performance of GA. GA operates on a population of potential solutions based on the principle of survival of the fittest to generate an improved generation or population which leads to approximations to the optimal solution.

Unlike GA, AGA emulates the natural genetic system more closely to the fact that the age of an individual affects its performance and hence it should be introduced in GAs. As soon as a new individual is generated in a population its age is assumed to be zero. Every iteration age of each individual is increased by one. As in natural genetic system, young and old individuals are assumed to be less fit compared to adult individuals [46] . The effective fitness of an individual at any iteration is measured by considering not only the objective function value, but also including the effect of its age. In GA once a particular individual becomes fit, it goes on getting chances to produce offspring until the end of the algorithm; if a proportional selection is used; thereby increasing the chance of generating similar type of offspring. More fit individuals do not normally die, and only the less fit ones die. Whereas in AGA, fitness of individuals with respect to age is assumed to increase gradually up to a pre-defined upper age limit (number of iterations), and then gradually decreases. This, more or less, ensures a natural death for each individual keeping its offspring only alive. Thus, in this case, a particular individual cannot dominate for a longer period of time. For the rest of the process, the evolution of AGA is same as that in GA.

The working principles of AGA and its procedure has been illustrated in Figures 3.6 and 3.7

```
Begin
    Generating population randomly
    While termination condition is not satisfied
        Evaluate the generation age
        Evaluate the population fitness
        According to the age of each generation
        the population is chosen for operation
        Loop
            Reproduction
            Crossover
            Mutation
        End loop
    End while when termination condition is satisfied
End
```

Figure 3.6 Pseudo code of AGA.

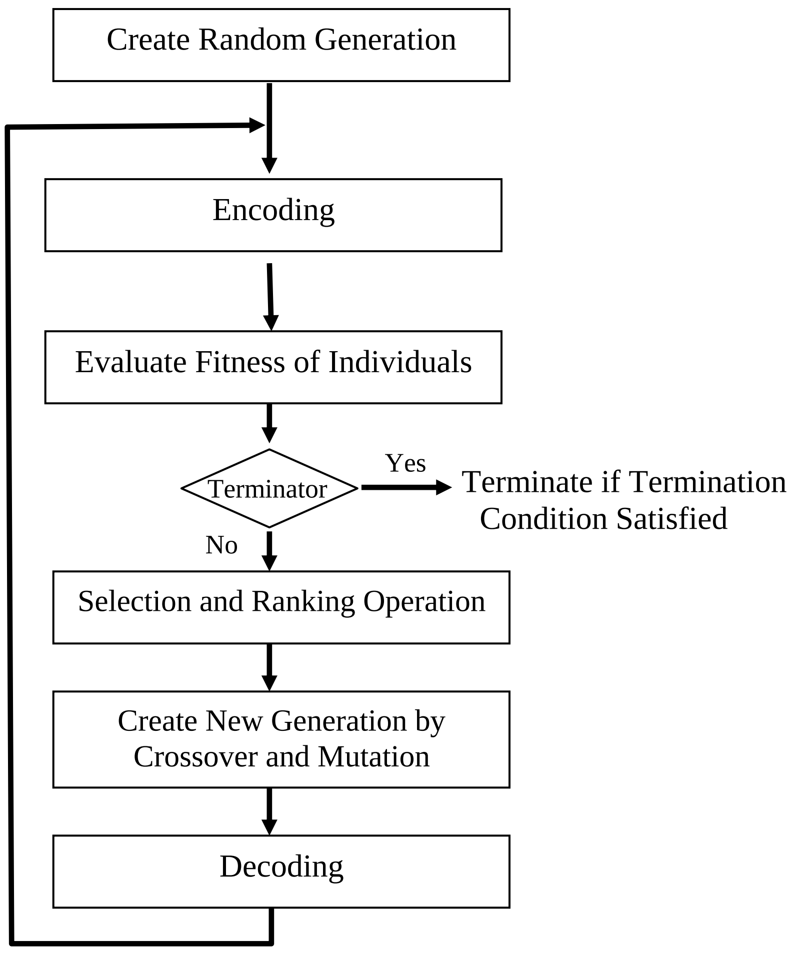

Create Random Generation
Encoding
Evaluate Fitness of Individuals
Terminator
Yes
Terminate if Termination Condition Satisfied
No
Selection and Ranking Operation
Create New Generation by Crossover and Mutation
Decoding

Figure 3.7 Flow chart of AGA.

3.5 Genetic Algorithm with Chromosome Differentiation

In the literature several approaches have been developed to improve the performance of traditional GA. GACD is considered as a modified version of GA which aims to increase the capability of GA in search space [5].

In traditional GA all individuals have the same sex and any two individuals can be crossed over. While in GACD the population is divided into male and female population on the basis of sexual differentiation. In addition, these populations are made artificially dissimilar, and both the populations are generated in a way that maximises the hamming distance between the two classes. The crossover is only allowed between individuals belonging to two distinct populations, and thus introduces greater degree of diversity and simultaneously leads to greater exploration in the search space. Selection is applied over the entire population, which serves to exploit the information gained so far. Thus, GACD accomplishes greater equilibrium between exploration and exploitation, which is one of the main features for any adaptive system. The chromosomes in the case of GACD are different as it contains additional gene that helps in determining the sex of the individuals in the current population.

In order to illustrate the performance of GACD and its working principles, pseudo code and flow chart of GACD are given in Figures 3.8 and 3.9.

```
Begin
    Generating population randomly
    While termination condition is not satisfied
        Evaluate the population fitness
        Loop
            Adding additional genes
            Determine the sex of the individuals
            Reproduction according to sex
            of the individuals
            Crossover
            Mutation

        End loop

    End while when termination condition is satisfied
End
```

Figure 3.8 Pseudo code of GACD.

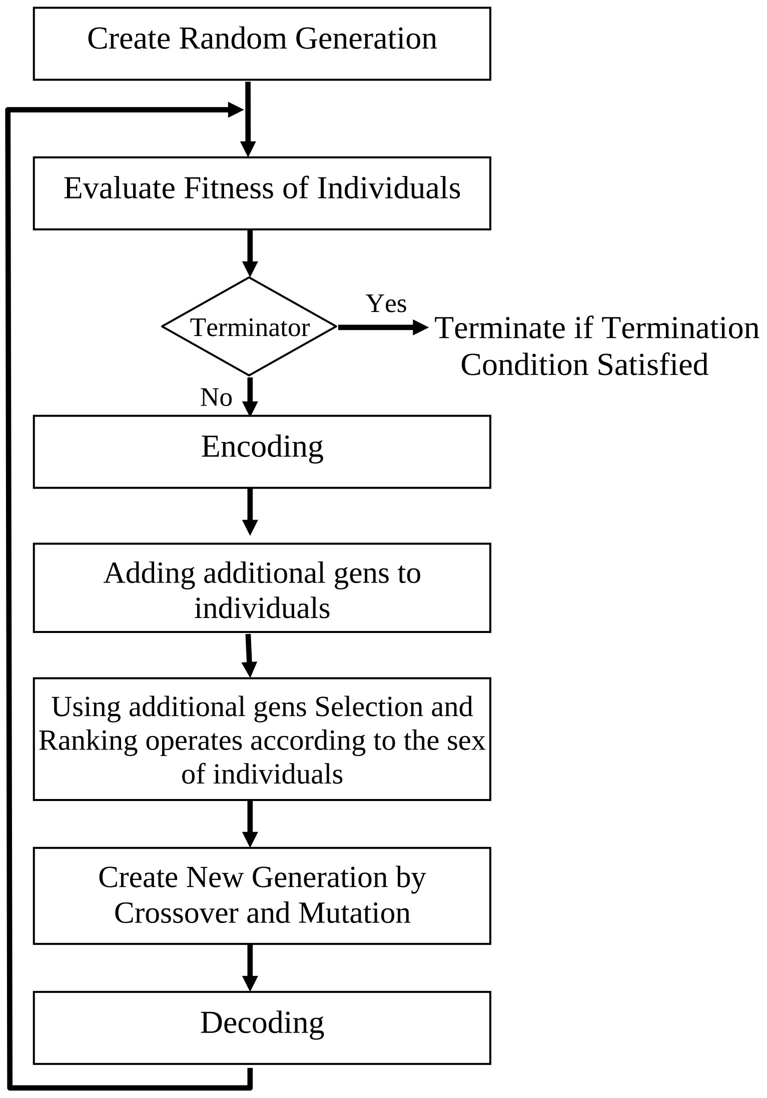

Create Random Generation
Evaluate Fitness of Individuals
Terminator
Yes
Terminate if Termination Condition Satisfied
No
Encoding
Adding additional gens to individuals
Using additional gens Selection and Ranking operates according to the sex of individuals
Create New Generation by Crossover and Mutation
Decoding

Figure 3.9 Flow chart of GACD.

3.6. Mimetic Genetic Algorithms

Many optimisation techniques used a local search method to improve the capability of optimisers in search space. In the literature several local searches have been used with GA to solve optimisation problems. MGA is considered as a heuristic method which is inspired by the notions of a mime [24]. In MGA, the chromosomes are formed by the mimes not genes (as in conventional GA). The unique aspect of the MGA algorithm is that all chromosomes and offspring are allowed to gain some experience before being involved in the process of evolution. The experience of the chromosomes is simulated by incorporating local search operation. Merz and Freisleben [88] proposed a method to perform local search through pair wise interchange heuristic. The local neighbourhood search is defined as a set of all solutions that can be reached from the current solution by swapping two elements in the chromosome.

MGA performance and working principles has been illustrated by pseudo code and flow chart in Figures 3.10 and 3.11.

```
Begin
    Generating population randomly
    While termination condition is not satisfied
        Evaluate population fitness
        Loop
            Reproduction
            Local search
            Crossover
            Mutation
        End loop
    End while when termination condition is satisfied
End
```

Figure 3.10 Pseudo code of MGA.

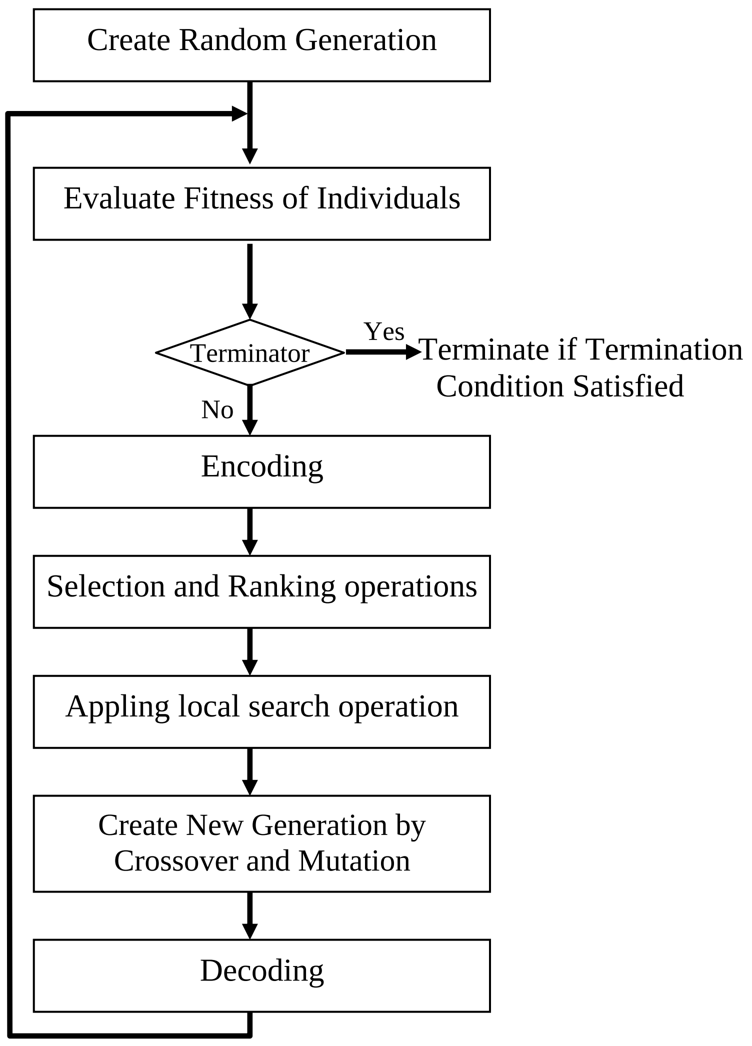

Create Random Generation
Evaluate Fitness of Individuals
Terminator
Yes
Terminate if Termination Condition Satisfied
No
Encoding
Selection and Ranking operations
Appling local search operation
Create New Generation by Crossover and Mutation
Decoding

Figure 3.11 Flow chart of MGA.

3.3 Particle Swarm Optimisation

Particle Swarm Optimisation (PSO) is first developed by Kennedy and Eberhart in 1995 [68]. It was simulated as a social behaviour of bird flocking or fish schooling to be used as an optimiser for solving various problems domains. Later, several PSO proposals have been presented to various optimisation problems. Moreover, recently the applications of PSO have been increased dramatically. This is because the advantages of PSO. For example:

- It does not have genetic operators such as crossover and mutation
- Easy to implement

- Fast comparing to other optimisers such as GA
- Using real numbers not binary
- Random search method

3.3.1 The Concepts of Particle Swarm Optimisation

PSO is a robust stochastic optimisation technique based on the movement and intelligence of swarms. The concept of PSO is taken from behaviour of bird flocking or fish schooling where each brid or fish called agent or particle moves through the solution space of the optimisation problem searching for the optimum solution and thus its position represents a potential solution for the problem. The scenario of this behaviour begins when the birds are looking for only one piece of food randomly. The birds do not know where the food is but they know how far it is at any iteration. The nearest bird to the food leads other birds to the food. PSO learns from this scenario and is used to solve optimisation problems where PSO lays in accelerating each bird toward its local best and the global best locations, with a random weighted acceleration at each moment or iteration.

These ideas were the main concepts or principles of PSO. The concepts of PSO mathematically have been represented in the following in Equations 3.1 and 3.2.

$$v_{id}(k+1) = \chi[\omega_{\kappa}(k)v_{id}(k)+c_1 r_1[p_{id}(k)-x_{id}(k)]+c_2 r_2[p_{gd}(k)-x_{id}(k)]] \quad \text{.............}(3.1)$$

$$x_{id}(k+1) = x_{id}(k)+v_{id}(k+1) \quad \text{...}(3.2)$$

where ω is the inertia weight, which represents the particle's preference to continue moving in the same direction it was going on the previous iteration, as introduced by Eberhart and Shi [32], χ is the constriction coefficient, which serves as a balancing factor for the local and global search, c1 and c2 are cognitive and social factors respectively, often set equal to 2, k represents the iteration number, r1 and r2 are random numbers between [0, 1], i (i = 1, 2, . . . , n) is the index representing the particles in the swarm and d (d = 1, 2, . . . , n) is the index for dimensions of the searching space.

Pseudo code of PSO is given in Figure 3.12 to illustrate the working principles of PSO.

```
Begin
    Generating particles position randomly
    Generating particles velocity randomly
    While termination condition is not satisfied
        Evaluate particles fitness
        Loop
            Update particles position randomly
            Update particles velocity randomly
        End loop
    End while when termination condition is satisfied
End
```

Figure 3.12 Pseudo code of PSO.

3.4 Comparison between GA and PSO

GA and PSO have many common properties as such both algorithms begin with a group of a randomly generated population and searches for optimal solution by updating generations. Moreover, both do not guarantee to find the optimal solution. Furthermore, both algorithms have fitness values for each individual to evaluate the population. On the other hand, there are some differences; unlike GA, PSO does not have operators such as crossover and mutation. In PSO, the individuals or solutions called particles fly through the search space and they are led by current optimal particle. These particles update themselves with the internal velocity and they also have memory which is an important factor in implementing the algorithm. PSO algorithm allows one-way sharing of information to others. While GA selection uses roulette, the PSO algorithm does not, and thus has the advantage of saving a lot of time. In GA chromosomes share information with each other, and thus the whole population moves like one group towards an optimal area.

This comparison shows that PSO has more advantages than GA such as PSO is easy to implement, take less time and there are few parameters to adjust. However, GA is better in global search while PSO is better in local search.

3.5 Algorithm Portfolios

Several optimisation algorithms have been proposed for solving combinatorial optimisation problems. However, algorithm portfolio consists of a collection of different algorithms or different copies of the same algorithm running on different processors [50]. The main objective behind the portfolios approach is the large variations in the performance of meta-heuristics for same problem of varying complexity or using different random seeds on the same problem instance. The purpose of embedding the evolutionary algorithms to form portfolios is to reduce the computational time required in obtaining the optimal solutions. This strategy is useful when good solutions can be found (not necessarily optimal ones) within stipulated time frames. Various types of optimisation algorithm are used in this work in order to obtain efficient optimisation.

An algorithm portfolio is a promising approach to solve various complex and dynamic problems with the help of algorithm portfolios. Such systems can be implemented on multi-processor machines where the communication between different algorithms, which in this case, at early stages, the fast and less accurate algorithm can pass its results to the slow and more accurate algorithm, which will benefit from the good results at an early stage. However, Figures 3.13 and 3.14 show the performance algorithm portfolio using pseudo code and flow chart for illustration.

```
Begin
      Generating population randomly
      While termination condition is not satisfied
            Evaluate generation fitness
            Loop
                Reproduction
                Process optimiser 1, Process optimiser 2 … Process optimiser n
            End loop
      End while when termination condition is satisfied
End
```

Figure 3.12 Pseudo code of AP.

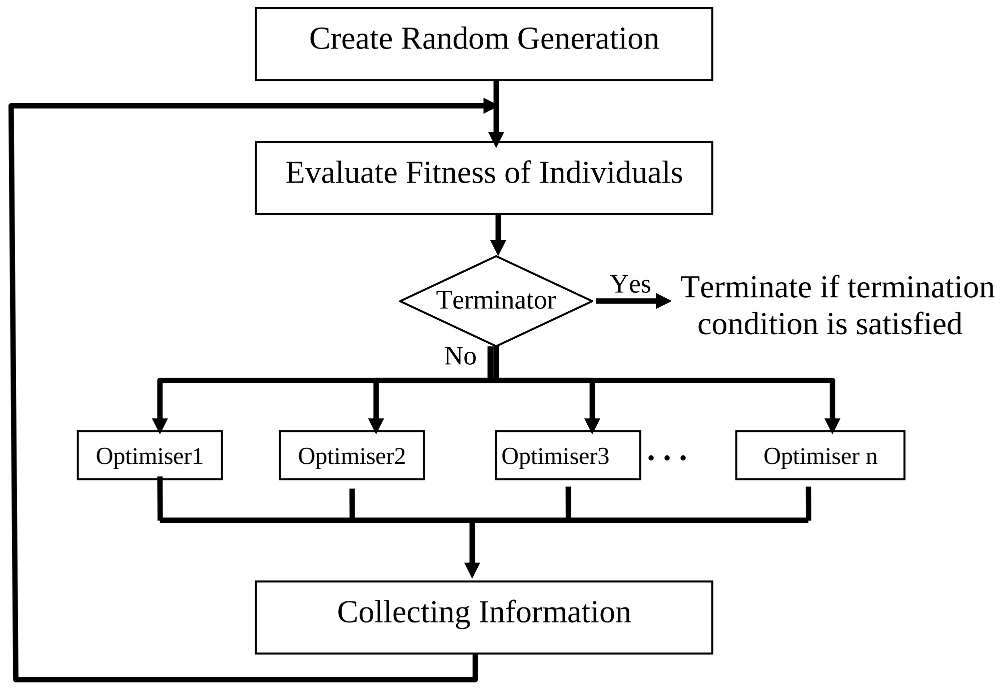


Figure 3.14 Flow chat of AP.

3.6 Summary

In this chapter, different types of intelligent optimisation methodologies have been given, namely: AP, PSO, GA, SGA, AGA, GACD and MGA. Moreover, the working principles and the performance of each optimiser has illustrated with a brief explanation of several selection techniques such as roulette wheel selection, tournament selection, rank selection, steady state selection which playing a main role in improving capacity of GA optimisers. Furthermore, the comparison between GA and PSO has been given which is concluded by that PSO is faster than GA and GA is more effective as global optimiser.

Finally, in this research, the advantages of each algorithm will be employed to solve scheduling of heating treatment operations problems. PSO is good as local search so will be used with GA to develop MGA. Moreover, portfolio algorithm can be applied using different intelligent techniques, such as GA and its derivates.

Chapter 4 Scheduling

4.1 Introduction to Scheduling

The concept of scheduling is not a new concept as it has existed for example in the building of bridges, dams and railways over hundreds of years ago whereby, none of the works could have been possible without the concept of scheduling the jobs involved. However, as a theory scheduling first appeared in the mid 1950’s being used to address problems within industrial applications which as a result increased the complexity of scheduling. However, in industry and manufacturing, scheduling is very important especially when several jobs have to be processed on limited resources. In this case of scheduling, the complexity is directly proportional to expending of search space area [34].

In this chapter, the concept of scheduling and its classification is illustrated. Different types of scheduling problems are described such as job shop problem, open shop problems, single machine or resource scheduling problems and flow shop scheduling problem. The complexity of scheduling problem and techniques which have been used to solve various scheduling problems are explained. The most common scheduling benchmarks which have been used to measure the capacity of each algorithm are referred.

4.2 Scheduling Classification

In real life, the classification is used to identify things so that others can distinguish them. Therefore, classifications aim to group similar things which are sharing in some characteristics. Scheduling problems may be classified according to various schemes [9] [52]. In general, scheduling problems depend on four categories:

1. Number of jobs and operation to be processed
2. Number and types of machines or resources that comprise the shop. Such machines can have single processor or multi-processor.

3. Optimisation technique and criteria.

   According to the criteria to be optimised, it is possible to classify scheduling problem according to the number of objectives such as:

   - single objective scheduling problem
   - multi-objective scheduling problem

   Another classification can be applied based on optimiser methodologies. The most common types are:

   - deterministic scheduling technique
   - stochastic scheduling technique

   Another classification can be applied based on the problem type:

   - static scheduling problem
   - dynamic scheduling problem

4. Flow pattern and further technological and management constraints. Possible values are:

   - single machine or resource scheduling problem
   - job shop scheduling problem
   - flow shop scheduling problem
   - open shop scheduling problem
   - permutation flow shop scheduling problem
   - parallel machines scheduling problem
   - job shop with parallel machines at each stage scheduling problem

4.3 Definition of the Problem

Many common terms or phrases have been found in each since or subject. These phrases mean specific meaning. Therefore it is very important to explain these words for the reader. However, in scheduling problem, there are common terms and phrases. In this section, the important phrases are illustrated and defend.

Scheduling optimisation problems is defined as, the organisation of a known sequence of actions or set of sequences along a time-line as such that execution is carried out efficiently and/or possibly optimally. By extension the allocation of a set of resources to such sequences of actions in order that a set of efficiency or optimality conditions are met [29].

Generally in scheduling problems, each job has one operation or more that to be processed on a single or multi resources with some conditions or restrictions. Moreover, scheduling problems aims to minimise the sum of interval time by performance of several jobs to allocation of resources. The sum of the interval time can be determined by measuring the time period between the start times of the first released job and the completion time of the last completed job. This amount of time is known as a makespan in the application of scheduling problems. Due-date time at which some external agency would like to have the job leaves the shop. It is the time by which the processing of the last operation should be completed. Furthermore, when job past its due date time, the mount time of this job is considered as tardiness time. While when job has been finish before its due date time, the mount time of this job is considered as earliness time. Flow Time is defined as the amount of time which used to process an operation. The total flowtime of a schedule is the total job flowtime in that schedule. In manufacturing, the time required to produce a part is called flowtime of scheduling jobs [98] [118].

4.4 Complexity of Algorithm

The complexity of algorithm can be divided to two branches [118]:

- Time Complexity
- Spatial Complexity

Time complexity can be defined as the processing time that consumed to implement the algorithm. Spatial complexity can be defined as the required space memory to implement scheduling algorithm. In the design algorithm these two complexities must to be considered. In the field of scheduling, the best algorithm which is able to solve scheduling problem with less complexity is required.

4.5 Complexity of Problem

Scheduling problems can be considered to be one of the hardest and most complicated optimisation problems, being due to the sheer vast expanse of the search space created, which inevitably means that the relative number of solutions is vast. To clarify what is meant here, the examples of two scheduling jobs are taken, one containing five tasks and the other containing ten tasks both with three resources available to it would mean that, the first containing the five tasks would have the number of solutions equal to $(5!)^3$. In this case the number of solutions is 1728×10^3 and it would not be easy to solve. However the second scheduling containing ten tasks with the number of solutions equal to $(10!)^3$ which would result with the 4.7785×10^{19} solutions, this being too large and hence extremely difficult to solve.

In real world applications, scheduling is much more complex because for example of the varies constraints, set of objectives and the size of the search space may be involved in relation to different types of scheduling and to solve the problem hence becomes much more difficult. Scheduling is therefore mostly classified as Non-deterministic Polynomial-time (NP-hard) problem which means essentially, very difficult to solve and are generally difficult type optimisation problems. Moreover, scheduling problems are considered to be so difficult that it may be unknown whether or not a problem even has an optimal solution [98].

4.6 Single Resource Scheduling with Tardiness and Earliest Penalties

In real life, scheduling jobs are very important and are considered as helpful tool whether in industry or any other places of need. Minimising the time or resources will first benefit from scheduling jobs. The classic methods of scheduling have been applied to minimise tardiness due date time. Recently, there has been a problem in industry from finishing before the due date time such as products store. This is leading to investigate the scheduling with consideration the tardiness and earliness due date time by many researchers. Recently it has received growing attention during the last few decades. Many algorithms have been proposed to solve the different variants of this problem [73] [2].

Single resource scheduling with tardiness and earliest penalties is considered as a special case of the scheduling problem. In this case each job has only a single operation. Since the set of jobs can be divided based on the machine required to perform the operation, each machine in the shop is independent of the others and can be scheduled separately. Therefore, in this case, it could focus on a single machine, and to the set of jobs that is to be processed on that machine [37].

4.6.1 Due-Date Time

The date time has the main role in scheduling optimisation problems especially single resource scheduling optimisation problems. It can be defined as the point of time at which customers or agents would like to have the job leaves the shop. In other word, the date time of last operation processing should be completed. It often results from a choice made by the decision maker and form to the analyst [118].

4.6.2 Earliest Due Date Time

Earliest due date can be defined as the mount of job's time when a job when this job finish before the due date time. In manufacturing completing jobs earlier than their due dates should be discouraged as much as tardiness time. This is because some reasons such as products store problems [2].

4.6.3 Tardiness Due-Date Time

Tardiness due time can be considered when completing job's time later than their due dates. In this case the mount of job's time will recognise as tardiness due date time which is undesirable in manufacturing [2].

4.6.4 Single Resource Scheduling with Tardiness and Earliest Penalties Model

The function of optimisers is to minimise tardiness and earliest penalties as much as possible. Thus scheduling of single resource with tardiness and earliest penalties has received growing attention during the last decade. Many algorithms have been proposed to solve the deferent variants of this problem.

In single resource scheduling problems with tardiness and earliest penalties, there are several jobs able to be optioned at time zero. These jobs have to be processed on a single resource. Each of these jobs needs exactly one operation. Flowtime of each job is known and it can be measured. Earliest due date time and tardiness due date time are deterministic and known. The aim of this scheduling problem is to minimise the makespan. Equation 4.1 shows how to count the makespan for a feasible schedule in case of the jobs finishing before the due date time. Equation 4.2 shows how to count the makespan for a feasible schedule in case of the jobs finishing after the due date time [73] [118].

$$S_j = \sum_{i}^{n} E_i \times f_i + \sum_{1}^{i} C_i \quad \ldots\ldots \quad (4.1)$$

$$S_j = \sum_{i}^{n} T_i \times f_i + \sum_{1}^{i} C_i \quad \ldots\ldots \quad (4.2)$$

where

S_j refers to the makespan (cost function) for feasible schedule.

f_i refers to the flowtime of a job

E_i refers to earliest due date time

T_i refers to tardiness due date time

C_i refers to accumulation of flowtime

The jobs of set S_j = (1, 2, . . . , n) have to be processed without pre-emption on the single resource. If a job has finished after due date time, tardiness due date time penalty has to be applied while if a job finishes before the due date time, earliest due date time has to be applied.

4.6.5 Complexity of problem

Single resource scheduling problems with tardiness and earliest penalties has been studied by researcher intensively. Since this problem is generated has been considered as NP-hard problem [77]. There are two types of common due-date problems which have been proven to be as NP-hard problems, namely if a restrictive common due date is given or if different jobs are given different weight to both penalties

In literature, there are two techniques to give weight to the tardiness and earliness due date time in scheduling optimisation problems. The first technique is to give each of them the same mount. The second technique is given both the same weight. In case of tardiness and earliness due date time have the same weight, this case of scheduling problems has been considered as solvable in pseudo-polynomial time [72]. While in cases of given both penalties same weight, the scheduling problems have been demonstrated as NP-hard in strong sense [77] [76]. Table 4.1 illustrate the level of complexity for single resource scheduling problems.

Table 4.1 Complexity of single resource scheduling problem.

Tardiness and earliest penalties	Restrictive due date time	Unrestrictive due date time
Weight of all penalties are equal	NP-hard, solvable in pseudo-polynomial time	Solvable in polynomial time
Weight of all penalties are not equal	NP-hard	NP-hard

4.7 Job Shop Scheduling Problem

The classic structure of job shop scheduling problem is first proposed in 1964 by Roy and Sussmann [106] and after that the form of JSSP has became as well commonly known scheduling problem in the application of production scheduling and industrial engineering.

JSSP can be described as the processing of scheduling for number of jobs which have number of operations on the number of machines or recourses considering these constraints:

- each machine can process at most one operation at given time.
- all the operations of a job are ordered and must be processed according to this ordered
- each operation needs to be processed during an uninterrupted period of a given length on a given machine

The aim of JSSP is to find a schedule, that is, an allocation of the jobs and operations on machines that has minimal flow-time.

4.7.1 Complexity of the Problem

JSSP is as one of the hardest computational problems. This complexity is caused by a set of constraints which JSSP has to implement. Therefore, JSSP is classified as NP-hard problem [45] [44].

4.8 Flow Shop Scheduling Problem

Flow shop scheduling problems is a typical computational optimisation problem that has been studied intensively [117] [132]. FSSP has some common characters with JSSP but there are some differences such as jobs in FSSP have to be processed in specific order.

Flow-shop scheduling can be introduced as the processing of scheduling jobs, where each job is to be processed with an identical order of machines. Each job several operations which have predecessor and direct successor except first and last jobs that have one of wither predecessor or direct successor [99].

4.8.1 Complexity of the Problem

The FSSP has been studied and proved that it considered as NP-hard [1]. This is because FSSP algorithm has several constraints which makes hard to solve optimally in reasonable time [45] [44].

4.9 Open Shop Scheduling Problem

The open shop scheduling problem [19] is considered as a special type of job shop scheduling problem where OSSP has the same rules of JSSP except no ordering constraints on the operations while job shop scheduling problem all the operations of a job are ordered. This makes the process of scheduling the operations and jobs on the machines more complicated than JSSP

Recently there has been an increased attention in research related to open shop scheduling [74] [79] [85] [101] [81] [84] [115]. Several applications has been done in different various areas in real world scheduling environments. For example, in manufacturing, production scheduling [74] [85] plane whether products or maintenance operations that it has to be done in aircraft, trains, buses, and ships. Further applications of open shop scheduling problems in automobile repair [38], quality control departments, teacher-class assignments [125], and satellite communications [101].

In OSSP, the 'operation' would generally refer to the processes steps involved. In manufacturing cases, these processes are to convert raw materials into completed products while each product in the stage of production is called a 'job' where each job contain more than one operations which have to be done by recourses. These resources are known as machines in the manufacturing. The production operations have some rules in order to operate the products successfully. Therefore, each of the resources has given certain constraints. The principles of classic OSSP are given as the following:

- two or more operations from the same job cannot be processed. In other words only one job can be handled by one machine at any given time.
- all jobs arrive at the same time and that time (t) is assumed to be t = 0.
- no ordering constraints on the operations.
- no job is processed twice on the same machine
- each job must be processed to completion
- there is no consideration for due dates of any job.

- each machine can do only one operation at given time and not allowed to interrupt any machine by other operations.
- there are no constraints on the processing times of each job on each machine.
- jobs may be started at any time no release times exist.
- jobs must wait for the next machine to be available.
- there is only one of each type of machine.
- machines are available at any time.
- the technological constraints are known in advance and are immutable.
- the aim of OSSP is to find a feasible schedule which satisfying the constraints by organising the set of jobs on machines for minimising the total time of flow-times.

4.9.1 Complexity of the Problem

The constraints lead to some difficulties such as same machines may be busy with operation of some products and some products have to wait until these products processing are finished. This is because all products enter the system in the same time and these products will go to most of the machines. This makes the process of scheduling all more difficult.

OSSP belongs to class of NP-hard problem. In 1976 Teofiloa and Sahni [115] proved that OSSP of more than two resources are considered as NP-complete. Later, Breit and Schmidt [12] studied OSSP with 2 resources for minimising makespan. In this study has been proven that the non pre-emptive two machine open shop problem with two jobs on one machine is considered as class of NP-hard. The corresponding problem with allowed pre-emption has been proved that the two machine problem is NP-hard in the ordinary sense [30].

It can be concluded by that OSSP is NP-hard problem which classic method will not be able to solve it optimally in reasonable time. Therefore, several algorithms have introduced to solve OSSP such dispatching rules, genetic algorithm and particle swarm optimisation. Some algorithms which used in literature are discussed in next following section.

4.10 Optimisation Techniques for Solving Scheduling Problems

In literature, many techniques have been used to solve scheduling problems. But there is still a need for developing an efficient algorithm in order to solve scheduling problems optimally in shorter time [33]. Several algorithms have showed that a lot of improvement in their results have been made. In this section, some of these techniques which have used to solve different scheduling problems are illustrated.

4.10.1 Dispatching Rules

The dispatching rules can be defined as choosing the operation from a set of jobs to be processing using specific rules where operations and jobs can be scheduled on resources taking into consideration certain rules, depending on the final objective or purpose for which a schedule is being generated. This technique was developed in the early 1960’s [42] [43] [9]. Later, using dispatch rules became very common and large varieties of different rules have been applied to a range of different scheduling problems. The common dispatching rules are listed in the following:

- earliest due date.
- first come, first served.
- smallest number of remaining operations.
- largest number of remaining operations.
- shortest processing time.
- largest processing time.
- job of identical setup.
- critical ratio scheduling.

Recently, researchers have been employed composite of different rules to improve dispatching rules techniques. This concept is known as composite dispatching rules.

4.10.2 Mathematical Programming

In this section, several mathematical programming techniques are illustrated. These techniques have been used to solve various scheduling problems [107].

4.10.2.1 Integer Programming

Integer programming [63] is an optimisation technique which used mathematical programming for solving optimisation problems and it has three common types:

- Linear Integer programming
- Nonlinear Integer programming
- Mixed Integer programming

The linear integer programming refers to mathematical programming with various different variables in the objective function and constraints while nonlinear integer programming deals with mathematical programming with continuous and discrete variables and nonlinearities in the objective function and constraints. Mixed integer programming is using both linear and nonlinear integer programming for solving optimisation problems.

4.10.2.2 Branch and Bound

Branch and bound has been developed by Land and Doig [75] in 1960 for the purpose of optimisation of problems which could be formulated as linear programming problems with additional constraints. This technique works based on determining a feasible region in which the solution exists in order to reduce the search to find the exact value within that region. The feasible region is divided to several braches in order to improve the search performance. In this technique, there are upper bound and lower bound for each braches where the upper bound refers to the fittest result in the region and lower bound refers to less fitness result in the region.

4.10.2.3 Dynamic Programming

It can be defined as a mathematical optimisation technique which is considered more complex than linear programming. In this technique, the problem is spilt to single stage order and each stage has only one variable (state-structured models) [6].

4.10.3 Heuristic Algorithm

Heuristic algorithm generally in optimisation problems aims to produce an acceptable solution to a problem using some optimisation techniques such as a campsite between different two optimisation algorithms. However, heuristic algorithm should be capable to solve optimisation problems which are known to be computationally difficult.

Scheduling is one of hardest optimisation problems. This difficulty has been discussed in previous sections. This is lead to introduce many heuristic proposals [79] [85] [38] [23]. Several heuristic algorithms can be applied to scheduling problems such as:

- Composite dispatching rule.
- Beam search.
- Genetic algorithm with local search.

4.10.4 Artificial Intelligence

During the last two decades AI techniques have been applied to various types of scheduling problems which clearly illustrates the increasing interest of researchers in this domain. This is largely due to advantages of AI over classic techniques such as traditional operations research techniques or dispatching rules in tackling the complexity of scheduling problems. Examples of AI techniques as illustrated in the following subsection:

4.10.4.1 Simulated Annealing

Simulated annealing method is taken from simulation of the physical process of annealing. Annealing can be described as the process of the cooling molten metal after heating this metal to reach the specific crystallite. This process of cooling is used to produce a more optimal solution [107].

4.10.4.2 Ant Colony Optimisation

Ant colony optimisation is a metaheuristic approach which developed to deal with hard optimisation problems. It has been introduced in the early 1990s by Dorigo et al [27] [28]. After that a number of ACO algorithms which has the same characteristic idea have been developed [56] [128].

Ant colony optimisation ACO takes inspiration from the foraging behaviour of some ants or agents. These ants put pheromone on the ground in order to distinguish the shortest favourable path that should be followed by other members of the colony. By this procedure, ants able to adjust any change in the environment such as discovering a new shortest path when the old one is no longer available due to a new obstacle. Ant colony optimisation exploits a similar procedure for solving optimisation problems. Although the artificial ants are used to mimic the behaviour of real ants, there are some differences such as the "artificial ants are not completely blind, they have some memory and they live in an environment where time is discrete" [27].

4.10.4.3 Tabu Search

Tabu search has been first presented by Glover in 1986 [47]. The basic ideas of TS are taken from a meta-heuristic superimposed on another heuristic. This set of concepts has also been introduced by Hansen in 1986 [57]. TS are used set of concepts or algorithms which lead to different types of TS algorithms. In other words, there is no single algorithm called tabu [134]. "The overall approach is to avoid entrainment in cycles by prevent or penalising moves which take the solution, in the next iteration, to points in the solution space previously visited (hence "tabu")" [107]
TS are designed to solve hard optimisation problem. Therefore, several proposal of TS has been applied to different various scheduling problems such as job shop scheduling problems [112] [84].

4.10.4.4 Genetic Algorithm

GA as illustrated in chapter 3 is classified as a global search technique used in computing to solve an optimisation problem and search problem such as scheduling problems. It has been shown that GA cable to deal with hard various scheduling problems [79] [85] [89].

4.10.4.5 Particle Swarm Optimisation

One of the suitable methods to optimise the scheduling problem is Particle Swarm Optimisation (PSO). PSO was developed by Kennedy and Eberhart [68] in 1995 using the concept of social behaviour of bird flocking or fish schooling to invent PSO optimiser model. For more details see Chapter 3 where PSO has been illustrated.

4.11 Scheduling Benchmarks

Scheduling problems is ranking to NP-hard problems which mean that it is hard to solve these problems optimally. Therefore, different various algorithms have been proposed to be applied on scheduling problems. Each proposal requires measuring its capacity for improving its performance. Therefore, various scheduling benchmarks have been proposed [111] [25] [23]. Researchers have been used these benchmarks to know the capacity of their proposals. Many comparisons have been done between several algorithms [79] [85] [38] [23].

4.11.1 Opens Shop Benchmarks

Taillard [111] presented different sizes of open shop scheduling benchmarks with upper bound and lower bound. For example four machines and four jobs, five machines and five jobs, seven machines and seven jobs, ten machines and ten jobs, fifteen machines and fifteen jobs. However, Taillard benchmark is considered as the famous and most widely used benchmarks in open scheduling felid or other types of scheduling forms.

4.11.2 Single Resource Benchmarks

In 2001 Biskup and Feldmann [8] proposed 280 different size benchmarks for the restrictive common due-date problem with general earliness and tardiness penalties. These benchmarks can be used to determine the capacity of each proposal by comparing the results with optimal results which have been found by the authors.

4.12 Summary

In this chapter, an overview of scheduling problems has been given. In more details, the single resource scheduling problems and open shop scheduling problems have been discussed. The complexity of scheduling problems is illustrated which are generally classified as NP-hard problems. This complexity basically caused by various constraints, set of objectives and the size of the search space. However, the most common techniques which have been used to solve different types of scheduling problems in literature are discussed. Moreover, the scheduling benchmarks which have been used to measure the capacity of each optimiser are mentioned.

Finally, the advantages of AI algorithms and its capacity to solve scheduling problems are led to use some types of AI algorithm in this research. Moreover, open shop job scheduling problem benchmarks and single resource scheduling problem with tardiness and earliness penalties benchmarks have been chosen to be use in this research. This is because; an open shop scheduling is more difficult than other types of scheduling problems. This is due to size of search space is vast. Single recourse scheduling benchmarks is chosen because the similarity between these types of scheduling and heating treatment scheduling which is considered in this research.

Chapter 5 Intelligent Scheduling Systems Developments

5.1 Introduction

In industry, a scheduling system that is capable of scheduling jobs on available resources is highly required. In this research an intelligent scheduling system is developed where the capacity of the system is tested to prove its quality and robustness.

In this chapter, an outline of the system design and the different types of optimisation algorithms is given. The benchmarks models of open shop scheduling and single resource scheduling which have been created are illustrated. The results of the tested system are provided using the two NB-hard benchmarks.

5.2 Developed Optimisers

Developing optimisers is the first step in building the whole system. However, in this research, before developing the optimisers, a study of the previous approaches to solve scheduling problems using artificial intelligence methodologies such as GA and PSO have been undertaken. These approaches highlighted few areas that needed further research in terms of methods and suitability of the algorithms. Based on this comprehensive study, several algorithms have been developed in this research by combining some of the positive aspects of the approaches studied and incorporating a few unique points.

The optimisation method used in this research is based on an algorithm portfolio which consists of a collection of different algorithms or different copies of the same algorithm running on different processors [50]. The main objective behind the portfolios approach is the large variations in the performance of meta-heuristics for the same problem of varying complexity or using different random seeds on the same problem instance. The purpose of embedding the evolutionary algorithms to form portfolios is to reduce the computational time required in obtaining the optimal solutions. This strategy is useful when good solutions can be found (not necessarily optimal ones) within stipulated time frames. Various types of genetic algorithm are used in this research in order to obtain efficient optimisation.

5.3 Genetic Algorithm

GA's are exploratory search and optimisation methods that were devised on the principles of natural evolution and population genetics [48]. Unlike other optimisation techniques, GA does not require mathematical descriptions of the optimisation problem, but instead relies on a cost-function, in order to assess the fitness of a particular solution to the problem in question.

In this research, different derivations of GA were implemented such as the Age Genetic Algorithm (AGA) [46], Mimetic Genetic Algorithms (MGA) [24], Genetic Algorithms with Chromosome Differentiations (GACD) [5] and Sexual Genetic Algorithms (SGA) [79] [82], for constructing the algorithm portfolios.

5.3.1 The Fitness Function

The fitness of individuals are measured based on an object function that measures the quality of each individual representation and determine the local best in the generation. In this research the optimisers deal with multi objective function. Accordingly, Pareto Front (PF) and Weighted Sum Method (WSM) are used.

5.3.1.1 Pareto Front

Pareto Front method is applied using the concept of dominated and non-dominated solutions where the solution has to be non-dominated by another solution for all objective function until added to non-dominated set. This non-dominated set is called "pareto optimal" or "pareto front" when the plot of the objective functions whose non-dominated is applied. Unfortunately, pareto front always does not gives a single solution, but rather a set of solutions.

5.3.1.2 Weighted Sum Method

Weighted sum method is chosen to be the second method for measuring the quality of individuals in each generation. In this method each object function is scaled and multiplies by weight coefficient. Equation 5.1 shows the formulae which is applied to measure fitness of each individual.

$$WSM = \frac{scale\ of\ \ f_1 \times weight\ coefficient\ + \cdots +\ scale\ of\ \ f_n \times weight\ coefficient}{n} \text{.... (5.1)}$$

where f_{n} refer to the object function and *n* refers to number of objectives while WSM refers to weighted sum method.

5.3.1.3 Single Objective Fitness

In this work, single objective is considered only when benchmarks are used to examine the system as a result to the benchmarks design. However measuring fitness of single objective is very simple and straightforward where the fitness of each solution can be compared with optimal result. If the optimal result is unknown, it can be compared with other available results.

5.3.2 Encoding

The first step in modelling the genetic algorithm is the outline of an encoding proposal which makes it possible for representing the solutions as a whole on a set of strings. In fact, encoding is considered as a critical phase that conditions all the subsequent steps. For scheduling problems the first encoding technique used is binary code. Table 5.1 shows how to convert integer number to binary number.

Table 5.1 Converting integer number to binary.

Integer number	Binary number
1	1
2	10
3	011
4	100
5	101
6	110
7	111
8	1000
9	1001
10	1010

In this technique possible schedule (solution) candidates are represented by a population of individuals (generation) and each individual is encoded as a binary string containing a well-defined number of chromosomes (1's and 0's). The number of bits for each job is chosen according to the capacity of each bits orbit and the number of jobs in each schedule. Table 5.2 shows the capacity of each bets orbit and total bets number for each job. In order to explain each schedule and the whole generation, several schedules are taken as examples as shown in Table 5.2.

Table 5.2 The optimiser capacity and bits number.

Total bits for each job $n+1$	capacity of bits orbit 2^n	Total jobs in schedule $\sum_1^n 2^n$
1 bit	2^0	From 0 to 1
2 bits	2^1	From 1 to 3
3 bits	2^2	From 4 to 7
4 bits	2^3	From 8 to 15
5 bits	2^4	From 16 to 31
6 bits	2^5	From 32 to 63

The second encoding technique used is real number or integer number where each job encoded by real number. So in this method, it does not need to change the schedules to binary code except in mutation operation. Using real value representation, provide some benefits such as the requirement of less memory, greater freedom in applying different genetic operators and the elimination of the conversion step from chromosomes to phenotypes before the evaluation of each function.

Table 5.3 Encoding individuals using binary code.

Jobs no.	Jobs Schedule	Bits no.	Binary Coding
5	1 2 5 4 3	3	001010101100101
7	7 6 3 2 4 1 5	3	111110011010100001101
10	10 8 1 2 5 4 3 7 9 6	4	10101000001010101100101011110010110
15	13 12 10 8 1 2 5 4 3 7 9 6 15 11 14	4	10101000001010101100101011110010110

GA began with a randomly generated initial population consisting of a set of individuals that represent the solution of the problem. The number of individuals is chosen based on the capacity of the optimisers and the complexity of the object function to be optimised. Increasing the number of individuals may lead to make the system or the optimisers very slow. Decreasing the number of individuals may lead to take long time until the optimiser determines the optimal solution [107]. Table 5.4 shows example of four jobs generation with four individuals.

Table 5.4 Encoding generation of four jobs and four individuals.

Four Job Real Number Generation	Four Job Binary Generation
4 2 1 3	101010001011
3 1 2 4	011001010101
1 4 3 2	001101011010
2 3 4 1	010011101001

5.3.3 Reproduction Operators

As mentioned in Chapter Three, the reproduction operators are used to achieve better generations. In this work some operators are applied such as crossover, mutation, selection, and elitism. The outline of the design of these operators is explained in the following subtitles.

5.3.3.1 Crossover Operator

Crossover is considered as the main operator of GA where the performance of GA highly dependants on it. The crossover operator is applied to create a new generation by combining the features of two existing individuals (chromosomes). In this case, the crossover operation creates variations in the solutions by producing new solution strings that consist of parts taken from selected parent strings.

In this work, different types of crossover are used such as Classical Crossover, Symbolic Crossover and Multi Crossover. The outline of each crossover is given in subtitles below:

a) Classical Crossover

The first crossover developed is the classical crossover. In this crossover, the recombination between chromosomes is made by crossover operator where two chromosomes called offspring are created from a couple of chromosomes called parents where each offspring taking the first part from one parent and the second from the other [107].

In this work, the crossover operator operates based on crossing site to generate recombination between the individuals. The crossing site between the chromosomes is chosen randomly and could be single or multiple depending on the length of the chromosome to diversify the population where in the case of the chromosome being very long, several crossing sites are taken and in case of the chromosome being short, single crossing site is taken. To illustrate how the classic crossover operator operates some example are given in Table 5.5.

Table 5.5 Examples of the classical crossover operation.

Crossing site point	Random no.	Parent 1	Parent2	Offspring 1	Offspring 2
1	4	10001110	11110001	10000001	11111110
2	2,6	11111111	00000000	11000011	00111100
3	2,4,6	00000000	11111111	00110011	11001100

b) Symbolic Crossover

Symbolic Real number or integer number crossover does not deal with binary number. Therefore, each individual is encoded by real number where it represents specific job. In other words, in this crossover, each individual represents a schedule and the crossover's operator which works in the same way as Classical Crossover except that the generation does not change to binary code so there is no need to encode the generation to binary code [89] [133] [99].

c) Multi Crossover

Multi crossover consists of both real number crossover and classic crossover. This is to get the advantage of communication between the crossovers which can improve the searching abilities of the system [107].

5.3.3.2 Mutation

The main role of mutation operator is to maintain diversity in population. Mutation operator works randomly by mimicking or to simulating the phenotype of mutation. If the probability of mutation is satisfied, the operator alters one gene in the chromosome by replacing another gene [107]. Table 5.6 shows an example of the mutation operation.

Table 5.6 Examples of mutation operation.

gene no.	Before mutation operation	After mutation operation
4	110000011	11010011
7	110000011	110000111

The aim of this operation is to prevent the generation from sticking at any local optimal area in the search space. Therefore, mutation operator is used only when the probability of the operator is satisfied and this probability is always very low.

5.3.3.3 GA Selection Operator

The next step after the completion of measurement is the fitness function, which is the selection process with the aim of determining or choosing individuals in the recent generation that will create offspring in the next generation. The process of selection works to emphasise fitter individuals which will lead to achieve better individuals with higher fitness.

In this research, Stochastic Universal Sampling (SUS) [107] technique is chosen to be the selection operator. This is because of some advantages of SUS that enables it to determine highly fit individuals over the generation. In SUS each individual has a space based on its fitness. However, unlike roulette wheel selection, [107] SUS operator uses fixed random number to make each individual to have an equal space according to fitness. Figure 5.1 explain SUS selection operator used in this research for six jobs: A, B, C, D, E, and F. Equations 5.2 and 5.3 show how to adjust fixed random number. This adjustment for SUS process aims at improving the performance of GA.

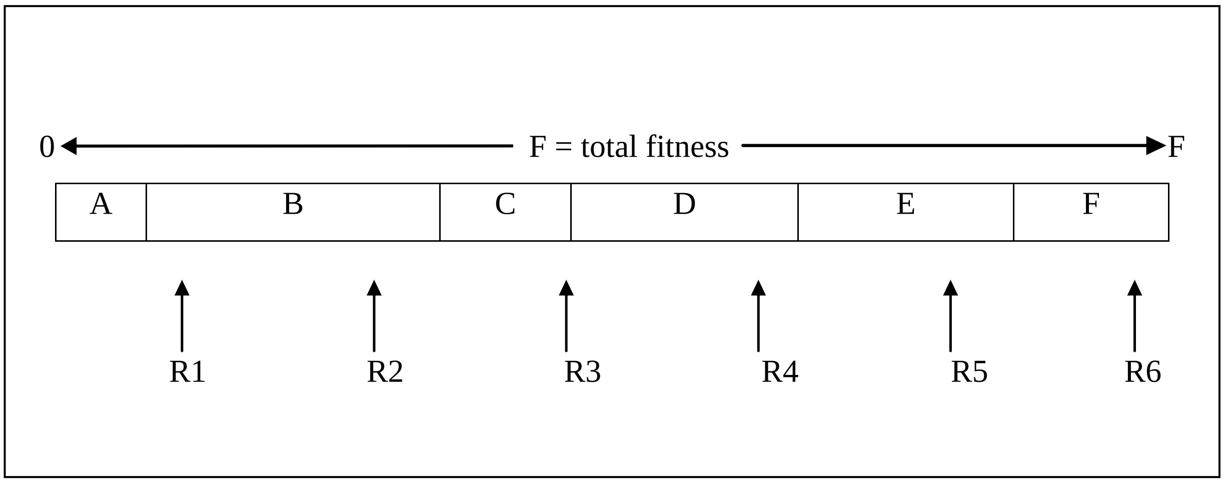


Figure 5.1 SUS selection for six jobs

$R = \frac{F}{N}$ ……………………………………………………………………….(5.2)

$R_n = R + \sum_{i=1}^{n} R_i$ ………………………………………………………………… (5.3)

where

F refers to the total fitness

N refers to the number of jobs

R is a fixed random number

The SUS selection process in the example of Figure 5.1 shows that a new generation which will include two individuals of B, one individual of C, D, E and F which will remove the lowest fit individual, A.

5.3.3.4 Ranking Selection

The selection operation comes after measuring each individual's fitness in the generation in order to rearrange them according to their fitness value. Based on the number of individuals in the generation, the ranking selection operator will remove the less fit individuals from the new generation and keep the highly fit individuals in the new generation. Table 5.7 explains how the ranking selection operator rearranges the individuals in the generation [107].

Table 5.7 Example of ranking selection operation.

Before Ranking Selection		After Ranking
Individual's order	Individual's fitness value	Individual's new order
1	30%	6
2	60%	4
3	82%	2
4	70%	3
5	55%	5
6	93%	1

5.3.3.5 Elitism

Elitism operation is used to keep the local optimal with the population of GA at each generation. This is because the local optimal might get lost during generating the new generation as a result of the operations of selection, crossover and mutation [107].

5.3.3.6 Decoding

Decoding operation is used to convert the generation from code such as binary into plain text. As a result of this operation many unfeasible solutions are generated which causes an increase in the search space. To fix this problem a scaling operation is developed [107].

5.3.3.7 Scaling Operation

Coding the GA operation in binary code or number code can lead to different solutions which might be repeated in the same set. As a result of crossover and mutation operations, many unfeasible solutions can be obtained in the new generation. In order to explain this, four operations are taken as an example in order to explain how crossover operation can produce unfeasible solutions:

Parent 1	1 2 | 3 4
Parent 2	4 3 | 2 1
Offspring 1	1 2 2 1
Offspring 2	3 4 3 4

This will lead to some operation being repeated in one job sequence. In order to prevent the unfeasible solution, a scaling process is being adopted in this work where each generation is scaled to obtain only feasible solutions which can be explained by the following:

The offspring before scaling.

Offspring 1	1 2 2 1
Offspring 2	3 4 3 4

The offspring after scaling

Offspring 1	1 3 4 2
Offspring 2	1 3 2 4

The scaling process takes the smallest value in a job to be the first operation, then take the second smallest value until all the operations are completed . In case there are two operations or more that have the same value, the scaling process takes the earliest operation first.

5.4 Classic GA Optimiser

The Classic GA optimiser is the first optimiser designed in this research. All previously mentioned operators are included in this optimiser. However, the optimiser starts the processing when it randomly generates an initial population of individuals. After that the fitness of the individuals are evaluated based on the objectives function. After encoding the generation, selection operator selects a pair of individuals from the current population using a rank order selection method to recombine and generate a pair of offsprings using crossover and mutation operators depending on their rates [48].

The outline of the proposed GA algorithm is listed as follows:

1. Generate the first generation randomly
2. Measure the fitness of each individual
3. Encode the generation
4. Ranking operation
5. Selection operation
6. Go to crossover and mutation operator to produce a new generation
7. Elitism operation
8. Decode and scale the generation
9. either the termination condition is satisfied or go to step 2

5.5 SGA Optimiser

SGA optimiser model has several common operations with the classical GA such as encoding crossover, mutation, elitism and decoding. But unlike classic GA, SGA has different type of selection technique where in case of GA, only one selection strategy is used. In SGA the individuals are divided to male and female. After that, the concept of male vigour and female choice is applied where the female chose the fittest between two males to produce a new individual by another operator such as crossover operator [79] [82].

An outline of the proposed SGA algorithm in this research is as follows:

1. Generate the first generation randomly
2. Measure the fitness of each individual
3. Encode the generation
4. Divide the individuals in the generation into males and females randomly
5. Applying the idea of male vigour and female choice
    a. Select two males and female randomly
    b. The female chose better male to generate a new offspring
6. Rearrange the individuals order in new generation according to the idea above
7. Go to crossover and mutation operator to produce a new generation
8. Decode and scale the generation
9. Either the termination condition is satisfied or go to step 2

5.6 AGA Optimiser

In designing the AGA optimiser, the age of each generation plays the main role in processing the generation. This is taken from the fact that the age of an individual affects its performance and hence it should be introduced in GA. Therefore, in model of AGA each individual is given an age. This age is assumed to be zero and every iteration the age of each individual is increased by one. As in natural genetic system, young and old individuals are assumed to be less fit compared to adult individuals. Figure 5.2 explains how to measure the fitness of individuals based on their age.

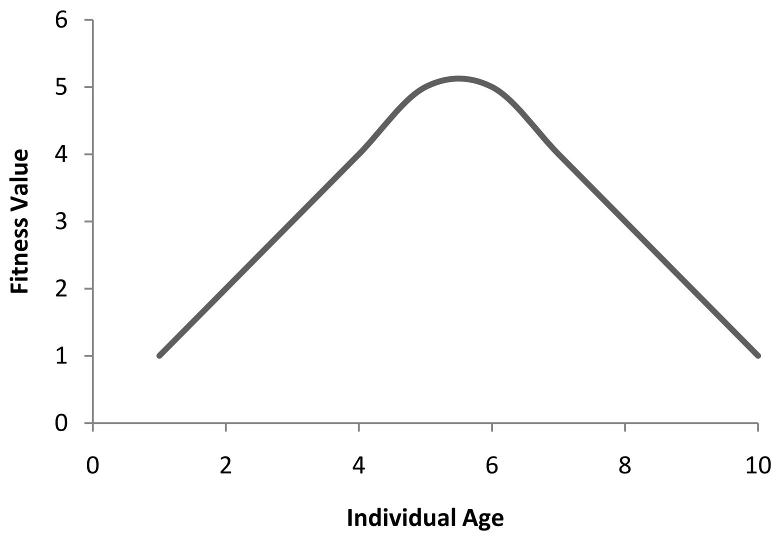


Figure 5.2 Fitness values of individuals based on the age.

In GA, selection operator selects the individuals according to their fitness value while in AGA, the selection operator select individuals based on two factors, their fitness value and their age fitness value [46].

An outline of the AGA algorithm is used in this research as follows:

1. Generate the first generation randomly
2. Count the age of individuals
3. Measure the fitness of the individuals
4. Encode the generation
5. Select the individuals based on:
    a. Age fitness value
    b. Individuals fitness value
6. Selected individuals go to crossover and mutation operators to produce a new generation
7. Decode and scale the generation
8. Either the termination condition is satisfied or go to step 2

5.7 GACD optimiser

In GACD model, individuals in the generation are divided into males and females population on the basis of sexual differentiation. Additional bits are added to the individuals for recognising the sex of the individuals. In fact, for giving a crossover a chance to make an exchange in the sex of the individuals, two bits are added. Table 5.8 outlines how the system identifies the sex of each individual from the additional bits.

Table 5.8 Additional bits for defining individuals' sex.

Additional bits	Sex of individual
0 0	male
0 1	female
1 0	female
1 1	male

In GACD, the selection operator rearranges the order of the generation based on the sex of the individuals where each pair of individuals is placed together which allow the crossover to produce a new generation from those selected [5].

An outline of the GACD algorithm used in this research is shown as follows:

1. Generate the first generation randomly
2. Measure the fitness of the individuals
3. Encode the generation
4. Add two bets to each individual
5. Select the individuals based on:
    a. Fitness value
    b. Sex of the individual
6. Selected individuals go to crossover and mutation operators to produce a new generation
7. Decode and scale the generation
8. Either the termination condition is satisfied or go to step 2

5.8 Particle Swarm Optimisation

PSO optimiser model is designed to deal with scheduling problem. Each schedule is represented by number of particles where each particle represents a job. Table 5.9 illustrates how the swarm represents a job schedule

Table 5.9 Example of PSO scheduling representation.

schedule	swarm
1 2 4 3	Particle1.1 Particle1.2 Particle1.3 Particle1.4
4 3 1 2	Particle2.1 Particle2.2 Particle2.3 Particle2.4
2 4 3 1	Particle3.1 Particle3.2 Particle3.3 Particle3.4

These particles move toward the local best position and global best position with each iteration. The local best represents the best schedule that has been found by particles at the current iteration while the global best refers to the best solution that has been found during iterations.

The initial velocities of the swarm are randomly generated which has been adjusted by maximum and minimum value toward a good solution. In this research, the maximum particle velocity value is +1 while the minimum is -1. Moreover, position of the particles has been adjusted between 1 to total jobs in schedule to prevent any particle going out of the search space. In case of the swarm being stuck in local optimal, penalties are applied to the best local particle for releasing the swarm [68]. The velocity and position of particles can be updated using the following formulae:

$$v_{id}(k+1) = \chi[\omega_{\kappa}(k)v_{id}(k)+c_1 r_1[p_{id}(k)-x_{id}(k)]+c_2 r_2[p_{gd}(k)-x_{id}(k)]] \quad \ldots\ (5.4)$$

$$x_{id}(k+1) = x_{id}(k)+v_{id}(k+1) \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ (5.5)$$

where ω is the inertia weight, which represents the particle's preference to continue moving in the same direction it was going on the previous iteration, as introduced by Eberhart and Shi [32], χ is the constriction coefficient, which serves as a balancing factor for the local and global search, c1 and c2 are cognitive and social factors respectively, often set equal to 2, k represents the iteration number, r1 and r2 are random numbers between [0, 1], i (i = 1, 2, . . . , n) is the index representing the particles in the swarm and d (d = 1, 2, . . . , N) is the index for dimensions of the searching space. These formulae allow the communication between the particles in whole swarm and allowed the nearest particle to the optimal solution for leading the whole swarm to the best solution.

5.9 Mimetic Genetic Algorithms

In the design of MGA all chromosomes and offsprings are allowed to gain some experience before being involved in the process of evolution. The experience of the chromosomes is simulated by incorporating local search operation [24].

In this research, PSO has been chosen to be a local search for its advantages such as the following:

- It has common properties with GA such that both algorithms begin with a group of a randomly generated population and searches for optimal solution by updating the generations and both algorithms have fitness values for each individual to evaluate the population.
- It is easy to implement
- It takes less time
- There are few parameters to adjust.
- GA is better in global search while PSO is better in local search

Accordingly the MGA local search engine is based on PSO. When the population is generated, it is passed to PSO for gaining some experience. The PSO will train the individuals to find local solutions to the problem within a constrained environment. Once the individuals are trained, they are passed back to the GA for performing the mating operations, and consequently finding solutions for the optimisation problem.
An outline of the proposed MGA algorithm used in this research as is listed as following:

1. Generate the first generation randomly
2. Measure the fitness of each individual

3. Go to PSO
4. Measure the fitness again
5. Encode the generation
6. Ranking operation
7. Selection operation
8. Go to crossover and mutation operator to produce a new generation
9. Elitism operation
10. Decode and scale the generation
11. Either the termination condition is satisfied or go to step 2

5.10 Algorithms Portfolios Optimiser

As mentioned in Chapter Three, algorithms portfolio consists of a collection of different algorithms or different copies of the same algorithm running on different processors. The main objective behind the portfolio approach is the large variations in the performance of meta-heuristics for the same problem of varying complexity or using different random seeds on the same problem instance [50].

In this work, all optimisers, classical GA, AGA, GACD, and MGA are used to design the algorithms portfolio model. In this design, communication between different algorithms are considered at early stages, where the fast and less accurate algorithm can pass its results to the slow and more accurate algorithm, which will benefit from the good results at an early stage. An outline of the proposed algorithms portfolio used in this research is listed as follows:

1. Generate the first generation randomly
2. Measure the fitness of each individual
3. Encode the generation
4. Ranking operation
5. Selection operation
6. Go to all optimisers to produce a new generation
7. Decode and scale the generation
8. Measure the fitness again

9. Send the most fit individuals to all optimisers
10. Elitism operation
11. Either the termination condition is satisfied or go to step 2

In this technique, each optimiser takes the generation after encoding and modifying this generation based on its principles such as in GACD, two bits are added, in AGA, the age of individuals is considered. After that the optimisers work individually in parallel to find a good solution during the iterations. The number of iterations for chromosomes exchange is chosen experimentally after several trails. The best iterations number that improves the performance of algorithm was selected. When the optimisers finish from the exact iterations, the best individuals are provided to the optimisers by the system while the worse individuals are removed. In the next generation, the optimisers will benefit from this advantage where good individuals have been harvested from other optimisers during their search for the optimal result. In order to illustrate the mechanism of the communication between the optimisers Figure 5.3 is given.

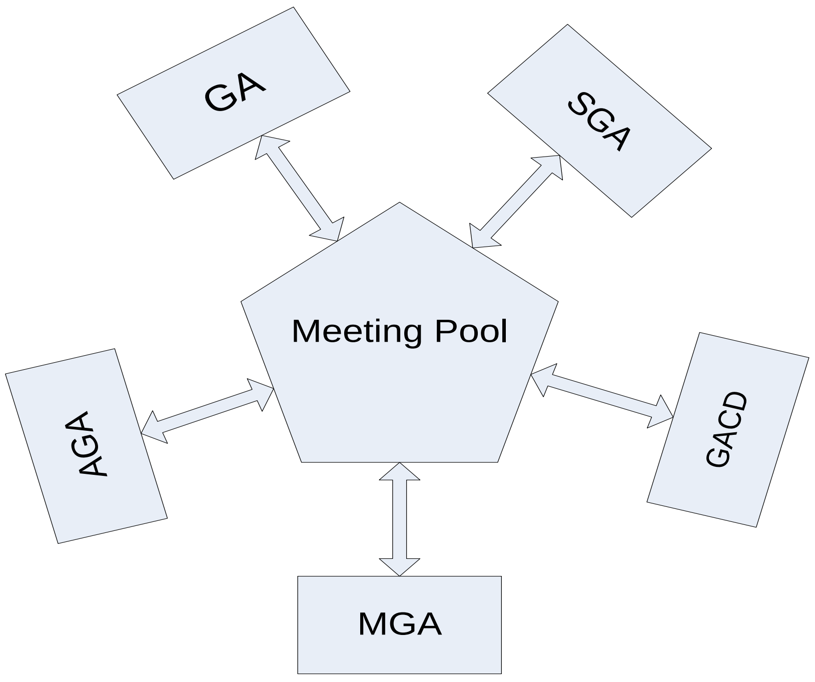


Figure 5.3 Communication between the optimisers.

In this work, after several simulations experiments are done, the number of iterations adjusting to be 4 where it gives the best performance for the algorithm. Hence, after each these iterations, the best generation for each optimiser is sent by the system to the meeting pool. The additional bets of GACD are removed by the system before send the generation to the meeting pool. In the meeting pool the order of individuals (chromosomes) are rearranged based on their fitness. After that the best individuals are chosen to send them to the optimisers while the less fit individuals are removed. The number of individuals that are chosen must be equal to the number of the previous individuals. Anew additional bets will added for GACD and the age of the individuals will be zero for these individuals that will send to AGA. SGA will divide the new individuals to male and female randomly. After that each optimiser will work for the specific iterations then the same processing will be done.

5.11 Benchmarking

Evaluating the various algorithms which have been developed particularly in comparison with each other is required. This lead to search for suitable benchmark which enable the authors to compare the optimisers' processes, performance and quality to others that are widely considered to be an industry standard benchmark or best practice.

As a result this research being focussed on scheduling problems; several scheduling benchmarks are studied so that suitable benchmarks for testing these optimisers can be selected. Based on relationships between these benchmarks and the subject of this research in processing, performance and complexity, two NB-hard benchmarks have been chosen for this proposal. The first chosen benchmark is a single resource benchmarks designed by Biskup and Feldmann [8] while the second benchmark is an open shop benchmark designed by Taillard [111].

5.11.1 Single Resource Benchmark

Biskup and Feldmann's single resource benchmarks has many common properties with multi objectives heating treatment scheduling which is considered in the research where both consider only one resource and both classified as NB-hard to solve problem. Both are made with consideration to common due-date where each job has to be finished at exact time or penalties will be applied. In common due date scheduling, there are two types of penalties, tardiness and earliest penalties [8].

a) *Single Machine Model*

Biskup and Feldmann [8] propose different size benchmark problems applied to single machine against common due dates with respect to earliness and tardiness penalties. To apply these benchmarks, the first step is to build the model of the problem.

The model of the single machine against common due date is developed with consideration that there are several jobs which have to be processed on a single machine where each job has only one operation. All jobs must be ready to be processed at time zero and any job finish before common due date an earliest penalty will be applied. Likewise any job finish after common due date, tardiness penalty will be applied to the cost function of schedule. Equation 5.6 outline how to count the cost function of jobs in both cases of job finish before common due date or job finish after common due date

$$f(s) = \sum^{n} \alpha_i E_i + \sum^{n} \beta_i T_i \quad \dots\dots\dots\dots\dots\dots\dots\dots\dots\dots \text{(5.6)}$$

where

$E_i = d - C_i$

$T_i = C_i - d$

$f(s)$ refers to the makespan (cost function) for feasible schedule.

C_i refers to completion time of a job

d refers to due date time

α_i refers to tardiness penalty of a job

β_i refers to earliest penalty of a job

p_i refers to the processing time of a job

The job processing times must be deterministic and known and the job pre-emption is not allowed. Moreover, some restrictions are applied as the following:

$T_i \geq s_i + p_i - d, \qquad i = 1, \dots, n$

$E_i \geq d - s_i - p_i \qquad i = 1, \dots, n$

$T_i,\ E_i, s_i\ \geq 0 \qquad i = 1, \dots, n$

$s_i +\ p_i\ \leq\ s_k +\ R\,(1 -\ x_{ik})\,, \quad i = 1, \dots, n-1;\ \ k = i+1, \dots, n$

$s_i +\ p_i\ \leq\ s_k +\ R\,x_{ik}, \qquad i = 1, \dots, n-1;\ \ k = i+1, \dots, n$

$x_{ik} \in \{0, 1\}, \qquad i = 1, \dots, n-1;\ \ k = i+1, \dots, n$

where

s_i refers to start time of job i

x_{ik} refers to value 1 if job i is sequenced prior to job k and value of 0 otherwise

The benchmark of eight jobs solved by Biskup and Feldmann [8] using heuristic algorithm is given in order to illustrate how benchmark is implemented. Table 5.10 shows details of each job such as the processing time of a job, tardiness penalty of a job and earliest penalty of a job. The result of this heuristic algorithm is shown in Figure 5.4.

Table 5.10 details of eight jobs benchmark.

p_i	7	1	18	6	13	14	5	6
x_i	2	8	4	9	5	5	7	4
β_i	14	7	8	9	7	9	5	14

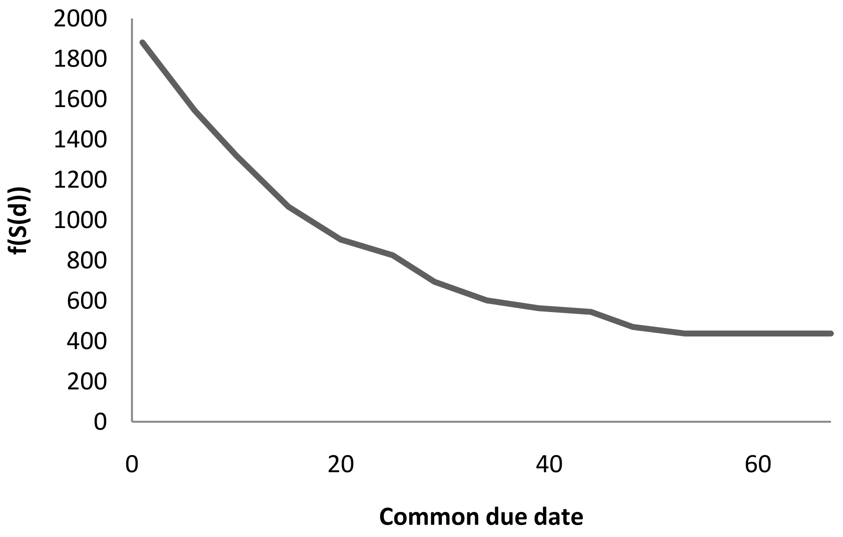


Figure 5.4 Results of eight jobs is done by Biskup and Feldmann [8].

5.11.2 Open Shop Scheduling Benchmark

The open shop scheduling problem in practice is considered as one of the most complex job shop scheduling problems, which are generally formulated by conventional approaches. There are many proposals that have been developed and in order to measure the capacity of these proposals benchmarks are required. However, Taillard benchmarks are considered as the famous and most widely used benchmarks in open scheduling felid. Therefore, these benchmarks have been chosen to test the capacity of our optimisers [111].

a) Open Shop Scheduling Model

The formal description of the open shop has to be applied to a model for generating open shop problem. In this work, m_i refers to the number of machines; n_j refers to the number of jobs where each job consists of at most O_{ij} operations and these operations are ordered to process on available machine. p_{ij} refers to the processing time for an operation O_{ij}; i refers to machine number and j refers to job number. An example of four jobs is given in Figure 5.5 for explaining how to open shop model is implemented.

	Job1	Job2	Job3	Job4
m_1	O_{ij}	O_{ij}	O_{ij}	O_{ij}
m_2	O_{ij}	O_{ij}	O_{ij}	O_{ij}
m_3	O_{ij}	O_{ij}	O_{ij}	O_{ij}
m_4	O_{ij}	O_{ij}	O_{ij}	O_{ij}

Figure 5.5 Example of open shop model schedule.

To generate a feasible schedule, all rules and constrains which are mention in Chapter Four must be satisfied such that it is not possible to process two operation or more from one job in at same time. Therefore, some operations have to wait until the process is completed and some machines are able to finish early while others may be getting some delay. However, the upper bound means the total time is spent for finishing all operation by latest machine while the lower bound means the total time of earliest machine. Equation 5.7 shows how to count the total time of each machine.

$$Total\ machines\ time(i) = machine\ waiting\ time(i) + \sum_{j=1}^{n} p_{ij} \ \ldots\ldots\ldots\ldots\ldots\ (5.7)$$

The object function of open shop scheduling is to minimise the upper and lower bound by giving a suitable schedule for ordering operations and jobs.

5.12 Integrated System Model for Scheduling Benchmark

The final step is done to examine the optimisers developed in this research, by integrating the optimisers with the scheduling model. The integration process is achieved as follows:

1. Generate the number of scheduling problem (generation) randomly
2. Measure the fitness of each individual using the benchmarks results as reference
3. Select the optimiser method and benchmark
    - Select the type and size of benchmark
    - Select optimiser technique
4. Select the encoding technique
    - Binary
    - Real number
5. Decode and scale the generation
6. Either the termination condition is satisfied or go to step 2

5.12.1 Parameters Adjustment

The GA's parameters are set as follows:
The population size is set to 40-100; the crossover rate is 94 – 99 %, the mutation rate is 0.1 – 1 %.
The PSO parameters are set as follows: population size of 40-100, while the inertial cognitive and social constants are as follows:
W(min) = 0.4, W(max) = 0.9, c1 = 1.4, c2 = 1.4, Velocity constraints = ±1.

5.12 Results of Single Machine Benchmarks

The experimental results of GA's on single machine benchmarks against common due dates with respect to earliness and tardiness penalties shows that GA's are able to converge to the optimal solutions faster than Biskup's heuristic as shown clearly in Figure 5.6 and Table 5.11. The experimental results of the optimisers applied to single machine benchmark [8] are recorded in sequence from Tables 5.12 to 5.23. In this benchmark the due date time (D) equal the total processing time.

Table 5.11 Results of GA and Biskup's heuristic for 8 jobs.

Common due date (D)	Biskup's Heuristic	GA
1	1881	1808
6	1544	1471
10	1320	1276
15	1066	1042
20	904	883
25	826	805
29	694	684
34	602	595
44	545	514
48	470	470
53	438	438
58	438	438
63	438	438
67	438	438

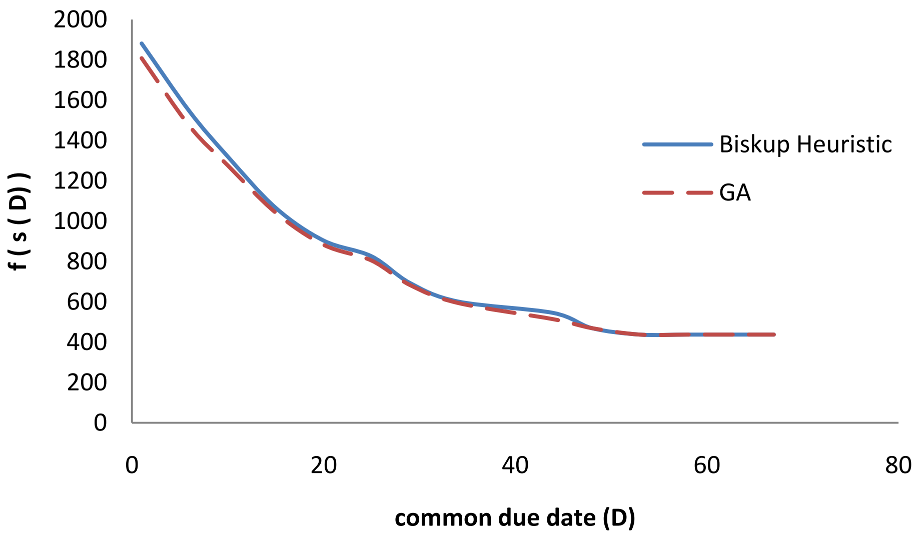


Figure 5.6 Results of GA and Biskup's Heuristic for 8 jobs.

Table 5.12 Results of 8 jobs with multi crossover.

	Crossover Rate	Mutation Rate	Result	Max Iterations
Optimal	-	-	438	-
GA	98	0.4	438	30
AGA	97	0.5	438	30
SGA	97	1	438	30
GACD	95	4	438	30
MA	97	0.1	438	20
Portfolio	97	1	438	10

Table 5.13 Results of 8 jobs using classical crossover.

	Crossover Rate	Mutation Rate	Result	Max Iterations
Optimal	-	-	438	-
GA	98	0.4	438	50
AGA	97	0.5	438	50
SGA	97	1	438	50
GACD	95	4	438	50
MA	97	0.1	438	30
Portfolio	97	1	438	20

Table 5.14 Results of 8 jobs using symbolic crossover.

	Crossover Rate	Mutation Rate	Result	Max Iterations
Optimal	-	-	438	-
GA	99	0.6	438	50
AGA	98	1	438	55
SGA	95	0.5	438	45
GACD	90	7	438	60
MA	96	1	438	40
Portfolio	97	1	438	20

Table 5.15 Results of 10 jobs using multi crossover.

Optimiser	Result of D=0.8	Result of D=0.6	Result of D=0.4	Result of D=0.2	Max Iterations
Optimal	818	841	1025	1936	
GA	818	841	1025	1936	100
AGA	818	841	1025	1936	100
SGA	818	850	1025	1936	100
GACD	818	844	1044	1936	100
MA	818	848	1014	1936	100
Portfolio	818	841	1009	1925.2	50

Table 5.16 Results of 10 jobs using classical crossover.

Optimiser	Result of D=0.8	Result of D=0.6	Result of D=0.4	Result of D=0.2	Max Iterations
Optimal	818	841	1025	1936	
GA	818	862.8	1044	1936	200
AGA	818	860	1025	1936	200
SGA	818	854	1044	1936	200
GACD	818	862.8	1025	1936	200
MA	818	841	1025	1928	200
Portfolio	818	841	1009	1936	100

Table 5.17 Results of 10 jobs using symbolic crossover.

Optimiser	Result of D=0.8	Result of D=0.6	Result of D=0.4	Result of D=0.2	Max Iterations
Optimal	818	841	1025	1936	
GA	818	862.8	1025	1930	200
AGA	818	841	1044	1936	200
SGA	818	862.8	1025	1936	200
GACD	818	862.8	1037	1936	200
MA	818	848	1025	1936	200
Portfolio	818	841	1014	1925.2	100

Table 5.18 Results of 20 jobs using multi crossover.

Optimiser	Result of D=0.8	Result of D=0.6	Result of D=0.4	Result of D=0.2	Max Iterations
Optimal	2986	2986	3066	4431	
GA	3075	3118	3071	4418	600
AGA	3086	3134	3077	4453	600
SGA	3132	3132	3072	4421	600
GACD	3118	3072	3089	4416	600
MA	2987	2991	3068	4384	400
Portfolio	2986	2986	3066	4372	300

Table 5.19 Results of 20 jobs using classical crossover.

Optimiser	Result of D=0.8	Result of D=0.6	Result of D=0.4	Result of D=0.2	Max Iterations
Optimal	2986	2986	3066	4431	
GA	3128	3114	3092	4431	600
AGA	3092	3112	3088	4431	600
SGA	3078	3010	3090	4431	600
GACD	3086	3072	3149	4431	600
MA	2987	3072	3072	4384	400
Portfolio	2986	2986	3066	4384	300

Table 5.20 Results of 20 jobs using symbolic crossover.

Optimiser	Result of D=0.8	Result of D=0.6	Result of D=0.4	Result of D=0.2	Max Iterations
Optimal	2986	2986	3066	4431	
GA	3086	3114	3071	4431	600
AGA	3108	3089	3092	4466	600
SGA	3185	3136	3072	4431	600
GACD	3131	3170	3149	4455	600
MA	2991	2991	3068	4384	400
Portfolio	2986	2986	3066	4372	300

Table5.21 Results of 50 jobs using multi crossover.

Optimiser	Result of D=0.8	Result of D=0.6	Result of D=0.4	Result of D=0.2	Max Iterations
Optimal	17990	17990	24868	42363	
GA	18850	18496	24868	42363	2000
AGA	18308	18308	24868	42363	2000
SGA	18326	18450	24868	42363	2000
GACD	18353	18,488	24868	42363	2000
MA	18303	18290	24065	41648	1500
Portfolio	18243	18195	24057	41636	1000

Table 5.22 Results of 50 jobs using classical crossover.

Optimiser	Result of D=0.8	Result of D=0.6	Result of D=0.4	Result of D=0.2	Max Iterations
Optimal	17990	17990	24868	42363	
GA	18458	18694	24868	42648	2000
AGA	18648	18488	24868	42363	2000
SGA	18556	18526	24868	42363	2000
GACD	18424	18445	24868	42363	2000
MA	18352	18303	24070	41707	1500
Portfolio	18266	18243	24063	41672	1000

Table 5.23 Results of 50 jobs using symbolic crossover.

Optimiser	Result of D=0.8	Result of D=0.6	Result of D=0.4	Result of D=0.2	Max Iterations
Optimal	17990	17990	24868	42363	
GA	18850	18479	24868	42462	2000
AGA	18402	18398	24868	42363	2000
SGA	18556	18450	24868	42363	2000
GACD	18494	18488	24868	45469	2000
MA	18352	18445	24070	41747	1500
Portfolio	18266	18243	24069	41674	1000

The results of the optimisers as recorded in the previous tables are used to evaluate the optimisers' capacity and robustness. In general, all optimisers were able to achieve a good solution which is not necessary to be the optimal one. Several new optimal results have been achieved by these algorithms such as 10 jobs, 20 jobs and 50 job for due date of 0.2 and 0.4. In most cases multi crossover can give better results. The symbolic crossover and binary crossover gives good results but with less robustness than multi crossover. There are different results for each optimiser to converge to the good solution. To analyses these differences, results of 10 jobs using multi crossover is selected to be a case study. Figure 5.7 illustrates the results for this case. In this case, MGA and GA optimisers were quick to find the optimal solution but with low accuracy where they began with a solution that is considered far from the optimal one. On the other hand, some optimisers were slow to achieve the optimal solution but started with better solution such as SGA and GACD. AP was collecting both advantages of fast convergence speed and accurate during the chromosomes exchange event when communication between the optimisers accrue. This gives the capability for the AP to be better.

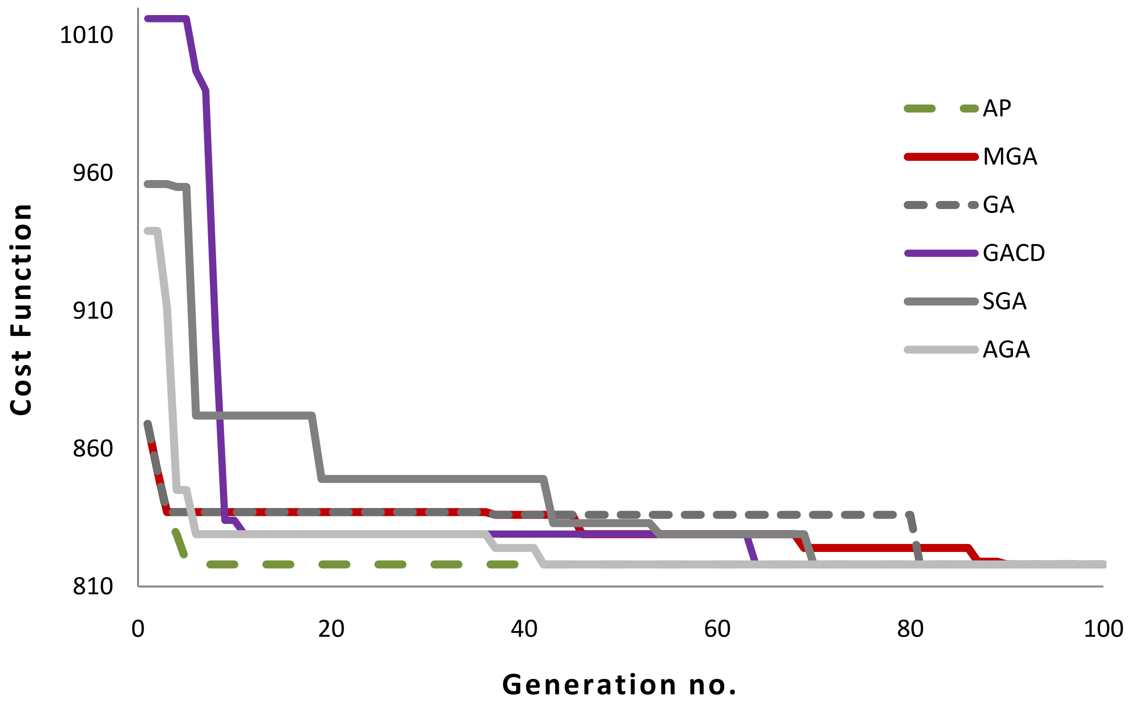


Figure 5.7 Results of 10 jobs single machine benchmark.

5.13 Results of Open Shop Benchmarks

Experiments results of the optimisers applied to scheduling of open shop benchmarks are recorded in sequence in Tables 5.24 to 5.32. UB refers to the upper bound of the schedule while LB refers to lower bound.

Table 5.24 Results of 4 jobs and 4 machines using multi crossover.

Optimiser	LB	UB	Iteration no.
Optimal	186	193	---
GA	186	193	58
AGA	186	193	67
SGA	186	193	102
GACD	186	193	76
MA	186	193	45
Portfolio	186	193	22

Table 5.25 Results of 4 jobs and 4 machines using classical crossover.

Optimiser	LB	UB	Iteration no.
Optimal	186	193	---
GA	186	193	79
AGA	186	193	65
SGA	186	193	98
GACD	186	193	145
MA	186	193	86
Portfolio	186	193	28

Table 5.26 Results of 4 jobs and 4 machines using symbolic crossover.

Optimiser	LB	UB	Iteration no.
Optimal	186	193	---
GA	186	193	45
AGA	186	193	163
SGA	186	193	77
GACD	186	193	148
MA	186	193	96
Portfolio	186	193	34

Table 5.27 Results of 5 jobs and 5 machines using multi crossover.

Optimiser	LB	UB	Iteration no.
Optimal	295	300	---
GA	274	300	387
AGA	274	300	285
SGA	274	300	398
GACD	274	300	380
MA	274	300	331
Portfolio	274	300	201

Table 5.28 Results of 5 jobs and 5 machines using classical crossover.

Optimiser	LB	UB	Iteration no.
Optimal	295	300	---
GA	274	300	345
AGA	274	300	325
SGA	274	300	369
GACD	274	300	393
MA	274	300	289
Portfolio	274	300	267

Table 5.29 Results of 5 jobs and 5 machines using symbolic crossover.

Optimiser	LB	UB	Iteration no.
Optimal	295	300	---
GA	274	300	325
AGA	274	300	374
SGA	274	300	392
GACD	274	300	359
MA	274	300	366
Portfolio	274	300	245

Table 5.30 Results of 7 jobs and 7machines using multi crossover.

Optimiser	LB	UB	Iteration no.
Optimal	435	438	---
GA	392	460	765
AGA	392	460	690
SGA	392	460	958
GACD	392	460	865
MA	418	455	699
Portfolio	367	451	423

Table 5.31 Results of 7 jobs and 7 machines using classical crossover.

Optimiser	LB	UB	Iteration no.
Optimal	435	438	---
GA	417	470	856
AGA	417	470	764
SGA	417	470	645
GACD	417	470	953
MA	434	454	691
Portfolio	428	450	532

Table 5.32 Results of 7 jobs and 7machines using symbolic crossover.

Optimiser	LB	UB	Iteration no.
Optimal	435	438	
GA	417	470	821
AGA	417	470	687
SGA	434	454	712
GACD	417	470	856
MA	434	454	576
Portfolio	367	451	571

The results as recorded in the previous tables, the optimisers were able to find the optimal solution for 4×4 open shop benchmark and 5×5 benchmark while for 7×7 benchmark, the optimisers achieved good solutions near the optimal results. As a result to the extent in the search space, the optimisers need to increase the number of iterations to achieve good solution. To evaluate the optimisers' capacity and robustness, 5×5 open shop benchmark is chosen as a case study. The result of this case is shown in Figure 5.8. It can be noticed that AP was the fastest optimiser which achieved the optimal result while GA, GACD and SGA were the slowest. AGA could achieve good solution at early stage but it took more than 200 iterations to converge to the optimal solution.

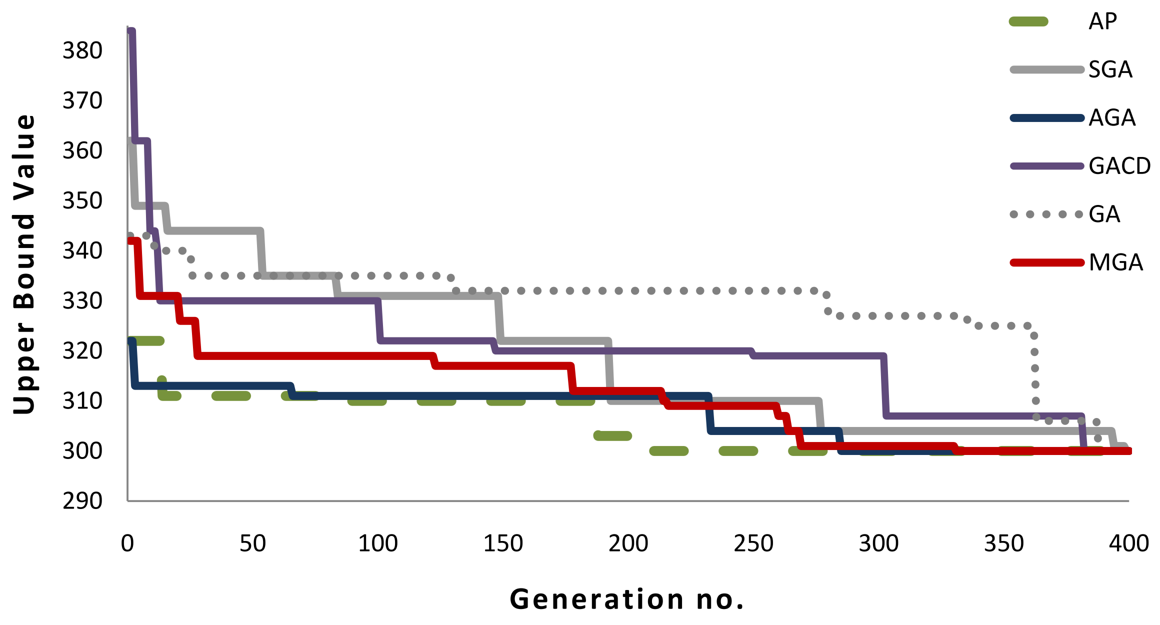


Figure 5.8 Results of 5×5 open shop benchmark.

5.14 Summary

GA's are not guaranteed to find the global optimum solution to a problem, but they are generally good at finding "acceptably good" solutions to problems. Where specialised techniques exist for solving particular problems, they are likely to out-perform GA's in both speed and accuracy of the final result.

In this research, many types of GA's are proposed for job scheduling using an effective crossover operation and scale operation for representation and decoding the solution into an active schedule during the search process.

The purpose of embedding the evolutionary algorithms to form portfolios is to reduce the computational time required in obtaining the optimal solutions. This strategy is useful when good solutions can be found (not necessarily optimal ones) within stipulated time frames. Various types of genetic algorithm are used in this work in order to obtain efficient optimisation.

All algorithms which have been developed applied to NP- hard problems. Results of these algorithms was excited and new optimal result such as d=0.2 and d=0.4 for ten, twenty and fifty jobs found by these optimisers. Moreover, GA's converge to optimal solutions faster than Bskup's heuristic technique.

It is clearly shown that the Portfolio and MA are superior to other optimisers such as GA, GACD, AGA and SGA. Moreover the experimental results show that using multi crossover technique was able to find better result and faster than other crossovers.

Chapter 6 Heat Treatment of Metals

6.1 Introduction

In the steel industry, determining the optimal heat treatment regime that is required to obtain the desired mechanical properties of steel is considered important so the optimal regime region can be set. Therefore, it is important to develop a system that is capable of selecting the optimal heat treatment regime so the required metal properties can be achieved with the least energy consumption and the shortest time. In order to develop this system, several models have to be developed such as furnace model, heat treatment model and optimiser model. In this work all of these models have been developed and integrated using intelligent systems.

In this chapter, the outline of developing a furnace model and heat treatment system is given. The design of intelligent system featuring the furnace model and heat treatment system with integration to the optimisers which are developed as in Chapter Five are illustrated.

6.2 Steel and its Heat Treatment Operation

Steel is related to a large family of iron-carbon alloys, which are able to be hammered or presses into shape without breaking or cracking, within some temperature range, during the casting operation. The principal raw materials used in steelmaking are iron ore, coal, and limestone. Using blast furnace these materials are converted into a product known as "pig iron," which contains considerable amounts of carbon, manganese, sulphur, phosphorus, and silicon. Pig iron is hard but able to break, and not appropriate for direct processing or machining. To refine pig iron as well as iron and steel, steel-makings use two methods. The first method is removing undesirable elements from the melt and then adding desirable elements in predetermined amounts while the second method using heat treatment operation.

The common content of steel is carbon element which its percentage plays a main role in heating treatment operation. Figure 6.1 shows steel's temperature against Carbon curve where this curve can be used to implement a specific heat treatment operation such as annealing and hardening. Steel can be classified based on grades ranges from a few hundredths of a percent to about 1 percent. Several elements are added to the steel to improve its properties such as Manganese, which acts as a deoxidizer and make hot working easier. Silicon, Chromium, Nickel and Sulphur are also added according to the utilisation. Other elements may be present, either as residuals that are not intentionally added, but result from the raw materials or steelmaking practice, or as alloying elements added to effect changes in the properties of the steel [119].

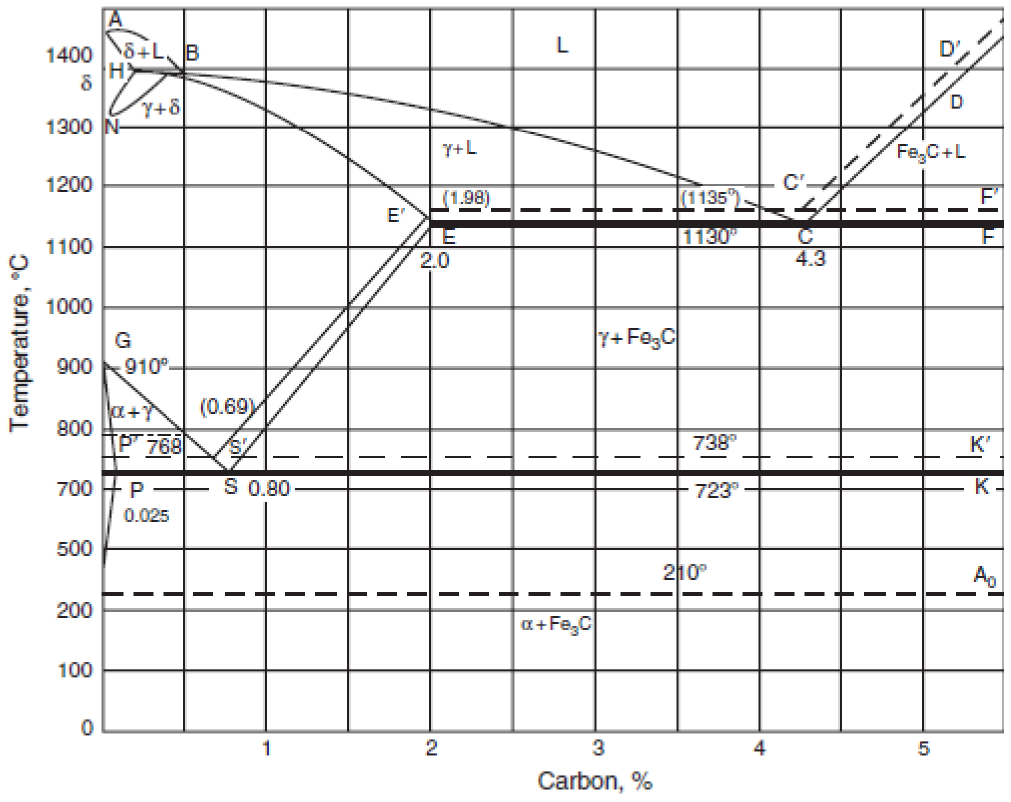


Figure 6.1 Steel's temperature against Carbon curve [119].

In manufacturing, steel can be shaped by the casting or using other manufacturing operation such as "hot working". Hot working means the operation which done when heating steel up until it reaches a critical stage suitable for rolling, forging and extrusion. The processes of a wrought mill shaping can be considered as hot working. Wrought steels are the most widely used of engineering materials, offering a multitude of forms, finishes, strengths, and usable temperature ranges. No other material offers comparable versatility for product design.

6.2.1 Heat Treatment Operation of Steel

Heating treatment operations aim to improve the martial properties such as strength and hardness without changing the shape of the products. Therefore, heating treatment can be defined as the process that is used to alter the physical and mechanical properties of the material without changing the product shape by controlling the heating and cooling rates where this processing lead to cause desired changes in the metallurgical structure. The properties of most metals and alloys can be affected by heating treatment operation [119].

In case of pure steel (cast iron) in solid state condition, there are two allotropic cases based on its temperature. The first case which is called Body Centred Cubic (BCC) placed in steel's temperature between low temperatures and till 910°C. It is also known as (α-Fe). The second case is called Face Cantered Cubic (FCC) or (γ-Fe). It is placed between 911°C to 1400°C. Above this temperature they again acquire a bcc lattice and are usually called δ crystals. The δ crystals differ from α crystals only in the temperature region of their existence. Pure steel has the following lattice constants: 0.286 nm for BCC lattices (α -Fe, δ -Fe) and 0.364 nm for FCC lattices (γ -Fe). β crystals come just above 770°C [119].

The lattice of paramagnetic β crystals is identical to the lattice of α crystals.. It can be said that pure steel can be changed from crystals case to another using heating temperature and cooling rate. Figure 6.2 outline the transformation between the cases of crystals which is done based on cooling rate [119].

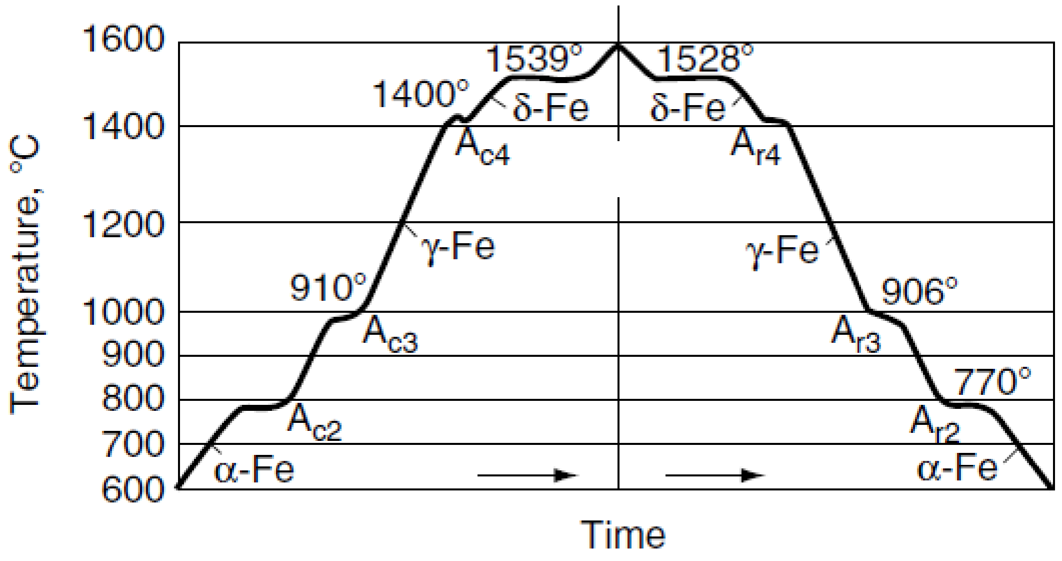


Figure 6.2 Heating and cooling curves for pure iron [119].

In Figure 6.2, c and r refer to whether the transformation is caused by heating or cooling. A change in the density of α -Fe as it transforms to γ-Fe results in an abrupt change in the volume of the material. Sometimes this gives rise to stresses that exceed the elastic limit and lead to failure. The density of γ-Fe is about 4% higher than that of α-Fe. However, steel heating treatment operations invest these transformations in the cases of crystals to get specific mechanical properties. There are several types of heating treatment operation such as hardening, annealing and tempering.

- Hardening (Quenching)

  It can be introduced as the process in which steel and cast iron alloys are heated into the austenitic crystal phase and then quickly cooled to produce a hardened structure. There are two types of hardening, surface hardening and through-hardening both are done based on the cooling rate that can be achieved [119].

- Annealing

  Stresses may be caused by previous processing operations such as welding, cold working, casting, forging, or machining. Annealing helps relieve internal stresses and reduce the chances for distortion and cracking. Depending on the temperature and time of annealing, various structural changes take place in a cold-deformed material. In annealing, steel is heated to a temperature above or within the critical range usually around 750°C, then cooled at a predetermined slow rate (usually in a furnace) to produce a coarse pearlite structure.

- Tempering

  Hardening causes decreasing in toughness as result of this, tempering is used to impart some toughness so it is always used after hardening. Therefore, the main objective of tempering is to provide a disperse structure at a preset degree of cooling for increasing the toughness.

6.3 Furnace Model Design

Furnace designs vary as to its function, heating operations, type of fuel and method of introducing combustion air; in spite of this most process furnaces have some common features. Figure 6.3 shows a furnace model designed by Yoshitani and Hasegawa. The quality of furnace design depends on fuel type, combustion efficiency, standby losses cycling losses and heat transfer. However, for applying optimisation techniques such as scheduling of heating treatment, a furnace model is required [131].

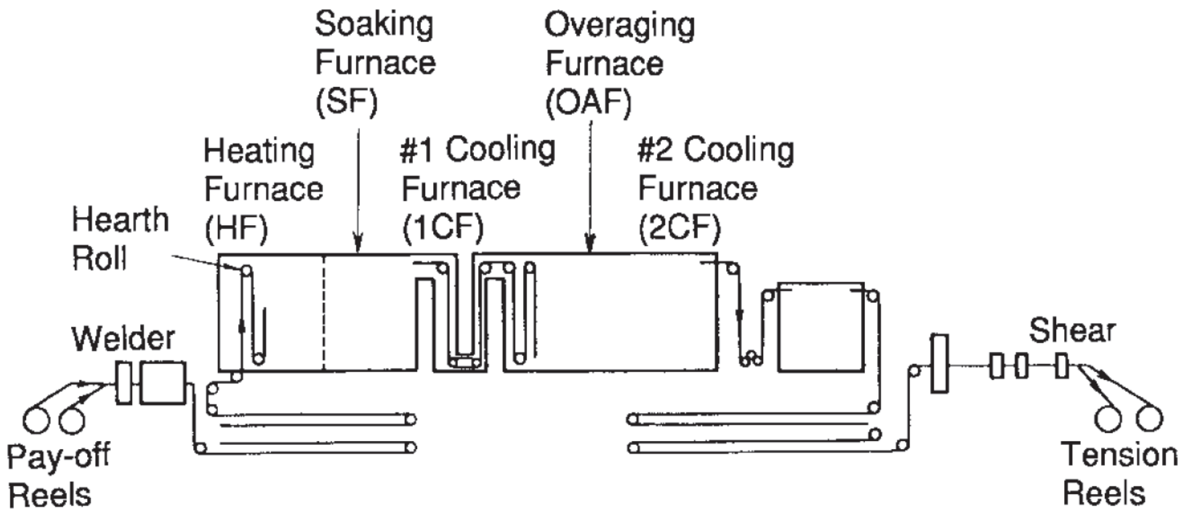


Figure 6.3 Outline of a continuous annealing process [131].

Indeed, in this work the furnace model should provide all requirements accurately such as the amount of consumed time and the amount of consumed fuel. Therefore, standard data [116] [26] are used to develop a furnace model for optimisation purposes. However this furnace is designed to heat martial to approximately to 1850°C therefore the walls are chosen to be of high alumina. This is because its bricks have a high density, Pyrometric Cone Equivalent (PCE) value higher than fire clays, good mechanical strength and creep resistance, and can be used up to 1850°C. High Alumina Al_2O_2 contains 73-85% Alumina. Its density 2750 kg/m² and its specific heat is between 0.8 - 1.2 kJ/kg °C [26], see Table 6.1. The shape of high alumina is 230mm×11mm×65mm and so its volume can be determined as 171925 mm³. This information and the size of the furnace are used to determine the total number of bricks and the total amount of heating required to heat up the wall of the furnace. Moreover, all gases that might be inside the furnace or to be used are considered during the modelling. Tables 6.1 and 6.2 show an example of standard data have been used to model the gases.

Table 6.1 Properties of dry air at P_B =760 mm Hg [26].

Temp (° C)	Density (kg/m³)	Specific Heat (kJ/kg.K)	Thermal conductivity	Diffusivity (m²/sec)
0	1.293	1.005	2.44	18.8
10	1.247	1.005	2.51	20.0
20	1.205	1.005	2.59	21.4
30	1.165	1.005	2.67	22.9
40	1.128	1.005	2.76	24.3
50	1.093	1.005	2.83	25.7
60	1.060	1.005	2.90	27.2
70	1.029	1.009	2.96	28.6
80	1.000	1.009	3.05	30.2
90	0.972	1.009	3.13	31.9
100	0.946	1.009	3.21	33.6
120	0.898	1.009	3.34	36.8
140	0.854	1.013	3.49	40.3
160	0.815	1.017	3.64	43.9
180	0.779	1.022	3.78	47.5
200	0.746	1.026	3.93	51.4
250	0.674	1.038	4.27	61.0
300	0.615	1.047	4.60	71.6
350	0.566	1.059	4.91	81.9
400	0.524	1.068	5.21	93.1
500	0.456	1.093	5.74	115.3
600	0.404	1.114	6.22	138.3
700	0.362	1.135	6.71	163.4
800	0.329	1.156	7.18	188.8
900	0.301	1.172	7.63	216.2
1000	0.277	1.185	8.07	245.9
1100	0.257	1.197	8.50	276.2
1200	0.239	1.210	9.15	316.5

Table 6.2 Specific heats of selected gases (kJ/m³ °C) [26].

Temp °C	CO_2	N_2	O_2	H_2O	Air (dray)	CO	H_2	H_2S	CH_4
100	1.720	1.301	1.320	1.502	1.305	1.265	1.290	1.541	2.106
200	1.808	1.303	1.337	1.517	1.310	1.302	1.297	1.574	2.330
300	1.881	1.310	1.358	1.538	1.320	1.310	1.302	1.607	2.530
400	1.944	1.317	1.380	1.560	1.330	1.323	1.304	1.645	2.721
500	2.045	1.330	1.400	1.583	1.344	1.331	1.306	1.683	2.893
600	2.060	1.342	1.415	1.608	1.360	1.344	1.310	1.721	3.048
700	2.108	1.353	1.437	1.638	1.372	1.361	1.315	1.758	3.190
800	2.152	1.370	1.453	1.660	1.382	1.373	1.318	1.796	3.341
900	2.205	1.382	1.466	1.686	1.400	1.390	1.323	1.830	3.450
1000	2.226	1.394	1.480	1.713	1.411	1.402	1.327	1.863	3.567
1100	2.263	1.405	1.493	1.740	1.424	1.415	1.336	1.892	—
1200	2.288	1.407	1.506	1.765	1.435	1.428	1.344	1.922	—
1300	2.316	1.430	1.512	1.791	1.440	1.440	1.352	1.947	—
1400	2.340	1.437	1.522	1.815	1.455	1.449	1.361	1.972	—
1500	2.366	1.447	1.531	1.840	1.464	1.461	1.369	2.000	—
1600	2.385	1.455	1.540	1.862	1.473	1.470	1.377	—	—
1700	2.404	1.462	1.548	1.884	1.481	1.478	1.386	—	—
1800	2.423	1.470	1.556	1.905	1.490	1.486	1.404	—	—
1900	2.440	1.478	1.564	1.952	1.496	1.494	1.412	—	—
2000	2.455	1.485	1.571	1.945	1.503	1.498	1.420	—	—

In this furnace model, to avoid the discontinuously during the processing the standard data, some improvement and treatment is done for these data such as treating the original data statistically. Scaling the data and taking exponential vector of these data is one methodology has been used in this work. The improvement and treatment in the data such as scaling is subject to the accuracy where any change must be within the data range. In case of unacceptable error, another statistic methodology is considered. To illustrate this improvement, dry air is given as in example. Figure 6.4 shows this modification where dry air specific heat data is replaced by exponential of these data. It can be noticed that this modification does not affect the original standard data which has been used in this work.

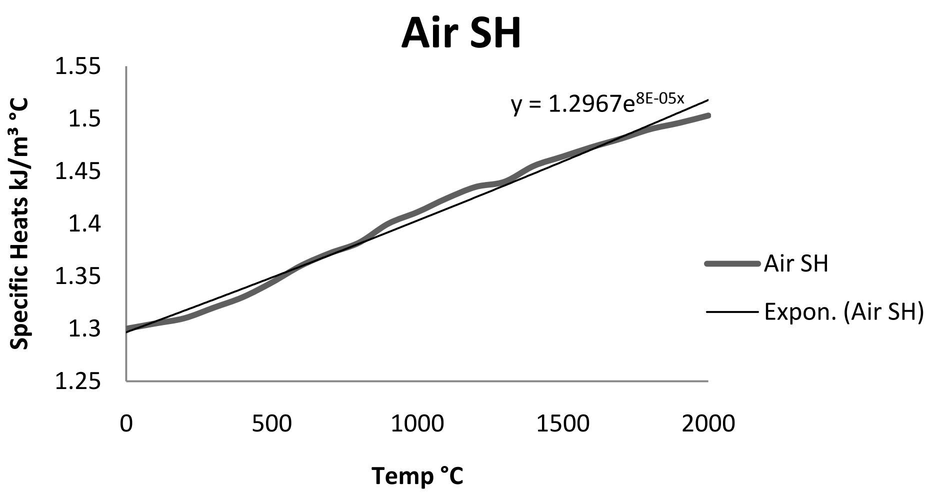


Figure 6.4 Specific heat of dry air and its exponential curve.

In this model cooling system is designed to reduce the furnace temperature to a desired setting point for other products. Drying air system works to cool the furnace after the products are taken out of the furnace. This is because, the system design considers the cooling rate for heat treatment operation is done in another section of the furnace using oil. This cooling system is modelled using concept of heating exchange. Figure 6.5 outlines the zones of the furnace where there are two zones, the first zone for heating up jobs and another zone for cooling the jobs based on the specific heat treatment operation. The size of heating zone is designed to be 4m length, 3m width and 2m height.

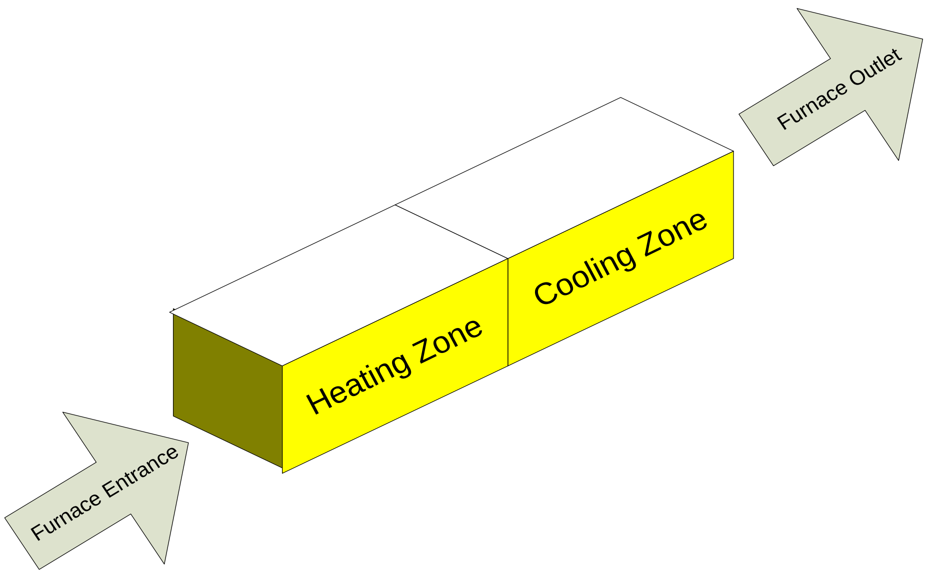


Figure 6.5 Furnace design.

This model also, is designed for several different fuels including various types of gas fuel and various grades of oil fuel. However, to model a furnace for optimisation proposes, it is very important to count the amount of heating which can be offered by the resource accurately. In this work, accurate measurements have been done; the details of these calculations are given in the fuel example as the following:

Carbon	86.2 %Wt
Hydrogen	12.3 %Wt
Sulphur	1.5 %Wt
Calorific value	43.401 MJ/kg
Theoretical air for combustion	10.714 m³/kg oil
Volume of combustion products	11.41 m³/kg

Composition of combustion product:

CO_2 (13.501 %)

H_2O (12.294 %)

N_2 (74.205 %)

While the combustion air will contain:

$O_2 \rightarrow 2.25$ m³

$N_2 \rightarrow 8.46$ m³

Where the theoretical air for combustion 10.71 m³ /kg oil and only 90% available air is given by the following:

$0.9 \times 10.71 = 9.64$ m³

This air will contain:

$O_2 \rightarrow 2.025$ m³

$N_2 \rightarrow 7.615$ m³

If 1 kg oil need 2.25 m³ O_2 to complete combustion. Hence only 2.025 m³ O_2 available

$\text{Unburned Oil} = 1 - \frac{2.025}{2.25} = 0.1\text{kg}$

It will contain 0.0862 Carbon.

The calorific value of oil is 43.4 MJ/kg which 90% is burned.

Calorific value = $43.4 \times 0.9 = 39.06$ MJ/kg

For Carbon reaction the absorbed heat is 9.92MJ/kg and unburned fuel is 0.1 kg which contains 0.0862 kg Carbon, hence:

Heat Absorbed = 9.902 × 0.062 = 0.853 MJ

Net Heat available = 39.06 – 0.853 = 38.207 MJ

In the case where the energy resource is electrical power then it can calculate the net heat available straightforward from the capacity of the heating system where the system can offer amount of heating per unit of time and no loss energy in this case during generating the heating.

The first priority of the furnace model is to calculate the consumed fuel and time for each job during the heating treatment operation. However, in this work for calculating consumed fuel, it is assumed that the fuel rate is constant. From the amount of the required heating for each operation to the desired temperature point, it is possible to calculate the fuel consumption. From the rate of fuel, the total amount of heat required and the time required for the cooling system, it is possible to calculate the time of each operation.

The simulation depends on the initial temperature point and the desired temperature point to calculate the fuel and time. In case of the initial temperature point is smaller than the desired temperature point, the simulation is set to measure the total amount of heat required and fuel consumption as Equation 6.1.

$$Q_t = Q_f + Q_g + Q_p + Q_l \quad \text{(6.1)}$$

$$C_f = \frac{Q_t}{H_{avi}} \quad \text{(6.2)}$$

$$T_t = \frac{Q_t}{F_{rate} \times H_{avi}} + T_{treat} \quad \text{(6.3)}$$

For the consuming time as Equations 6.2 and 6.3. In case of the initial temperature point is bigger than desired temperature point. The cooling system will be activated using dry air to decrease the temperature of the furnace until desired temperature point is reached by the concept of heating exchange. This means no fuel will be used in this stage. Except when the temperature reaches to constant level for the purpose of heat treatment, the fuel will feed the furnace to keep the temperature at the desired point. In this case, the total amount of consumed heat will be equal to the mount of the lost heat plus the amount of heat absorbed by the workspaces as illustrated Equation 6.4.

$Q_t = Q_p + Q_l$ ……………………………………………………………………….(6.4)

The treatment time will be the time of cooling plus the time of the heat treatment operation as Equation 6.5 [131] [26].

$T_t = T_{cool} + T_{treat}$ ……………………………………………………………...(6.5)

where

Q_t refers to the total amount of heat that is required for an operation.

Q_f refers to the amount of heat that is required for getting furnace the desired temperature point.

Q_g is the mounts of heat is absorbed by gases to get desired temperature point.

Q_p refers to the amount of heat that is absorbed by workspaces to get the desired temperature point.

Q_l refers to the amount of lost heat.

T_t refers to the total time required for an operation

F_{rate} refers to the fuel rate

C_f the fuel consumption

H_{avi} refers to the net heat available of burning fuel

T_{treat} refers to the time required for heating treatment operation.

T_{cool} refers to the time of cooling system

An unscheduled 10 jobs sequence simulation is shown in Figure 6.6. Due to the large differences between the sequenced jobs temperature, the furnace temperature has to be raised and lowered to meet the required temperature for each piece. Since the furnace has to be heated and cooled to meet the required piece temperature, this will cause the process to take a long time and high energy consumption. The need for optimisation to schedule for shortest time and lower energy consumption will be achieved through the optimisation system.

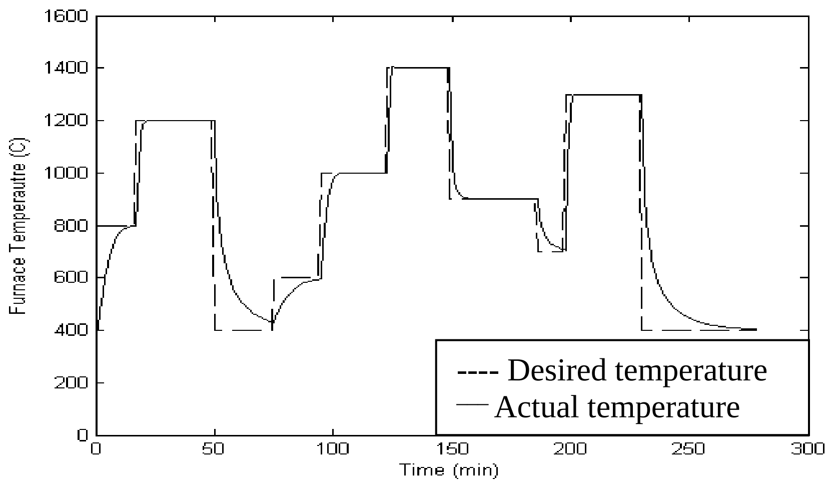


Figure 6.6 Unscheduled 10 jobs heating sequence.

6.4 Heat Treatment Model Design

The next step in building the intelligent system is to develop a model that is capable of selecting the optimal heat treatment regime which is required for heating treatment operation that is able to give desired mechanical properties. In fact classic methods fail to generate accurate model. This is due to the complexity of materials behaviour during heat treatment operation under different conditions. Moreover, there is a large space search area to find heat treatment regime which is capable to give the best results. This lead to select a suitable methodology based on an NF system and PSO for mechanical properties of the steel. The detail and explanation are given in next subtitles.

6.4.1 Neuro-Fuzzy Model

In the field of artificial intelligence, Neuro-Fuzzy (NF) refers to combinations of artificial neural networks and fuzzy logic [105]. A NF system is defined as fuzzy system which employs a self learning algorithm derived and inspired by neural network concept to achieve its fuzzy sets and fuzzy rules using processing data samples. In this work, experimental heat treatment data is used to develop the NF system that models the materials properties such as the hardness and its depth. The heat treatment data that is used to train the NF model is organised as input parameters (diameter, austenitic temperature, tempering temperature, time of heating treatment, pressure, and sub-zero treatment) which the outputs are the hardness and its depth. Figure 6.7 outline the inputs and outputs for NF system model.

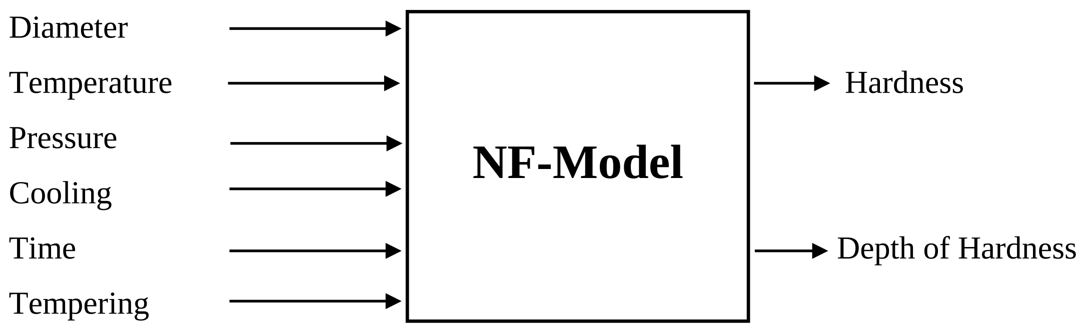


Figure 6.7 Design of NF model.

The heat treatment data are taken from industrial company called MIC in Saudi Arabia. Nevertheless, not much choice is given to the author due to some restrictions such as limited materials are used and limited heat treatment operations available, several experiments have been carried out that later used to develop the NF model. The descriptions of these data is the following

- 4 types of steal have been used and their composition are given in Table 6.3
- The heat treatment operation were done called hardening (quenching)
- A classic furnace with two zones are used
- The materials is quenched using oil in cooling zones
- The length of the parts is fixed to be 30 cm.
- The ranges of heat treatment operation regime and their outputs results are given in Table 6.3

Table 6.3 Ranges of experimental data.

Material	Diameter (mm)	Austenitic Temperature (°C)	Pressure (Pa)	Sub-zero treatment (°C)	Time (min)	Hardness results (HV)	Depth hardness results (mm)
14 Ni Cr 14	80 - 100	850 - 900	8 - 12	50 - 70	20 - 45	407 – 440	23 – 61
16 Mn Cr 5	60 - 120	850 - 900	8 - 12	50 - 70	20 - 45	330 – 423	23 – 59
30 Cr Ni Mo 8	60 – 90	850 - 900	8 - 12	50 - 70	20 - 45	542 – 610	23 – 57.5
20 Mn45	60 - 75	850 - 900	8 - 12	50 - 70	20 - 45	735 - 794	23 – 85.5

In this work, two membership functions are chosen to design the NF system. After this treatment the model is trained and tested using NF model which give very good prediction accuracy. To evaluate the NF model, the surfaces of NF model are examined. These surfaces have shown that the system works effectively where the system cover the range of heating treatment operations which are considered in this work. In this chapter, some examples of these surfaces' figures are given. Figures 6.8, 6.9 and 6.10 show some surfaces of the model. In Figures 6.8 and 6.10 the output is the hardness which is measured using Vickers Hardness while in Figure 6.9 the output is depth of the harness. Several inputs are used to options these figures consequently, temperature, pressure, time, and diameter

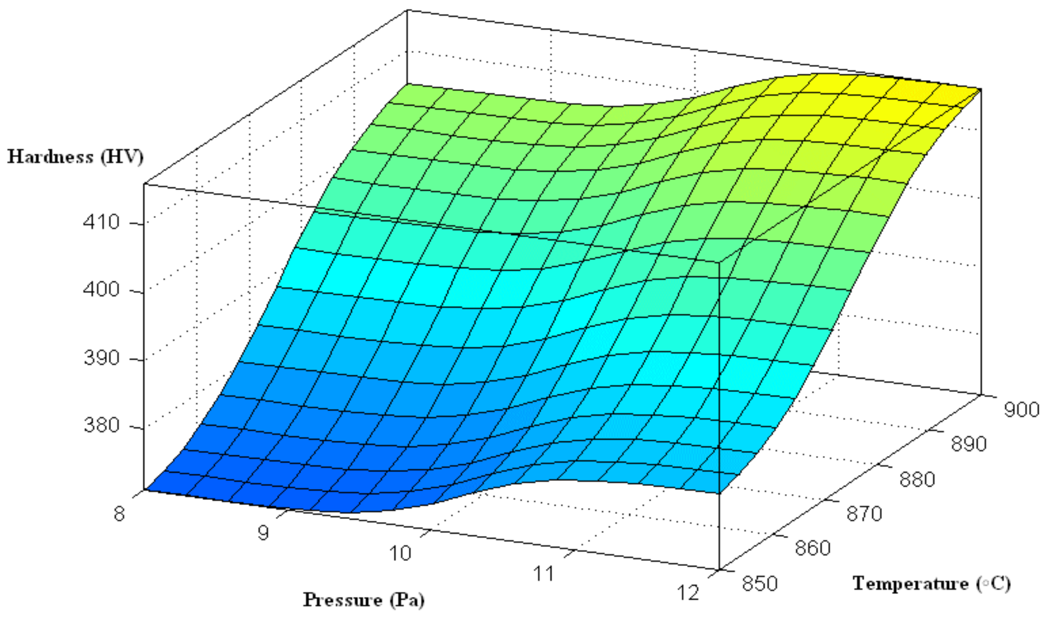


Figure 6.8 NF surface for temperature and pressure inputs and the hardness as output.

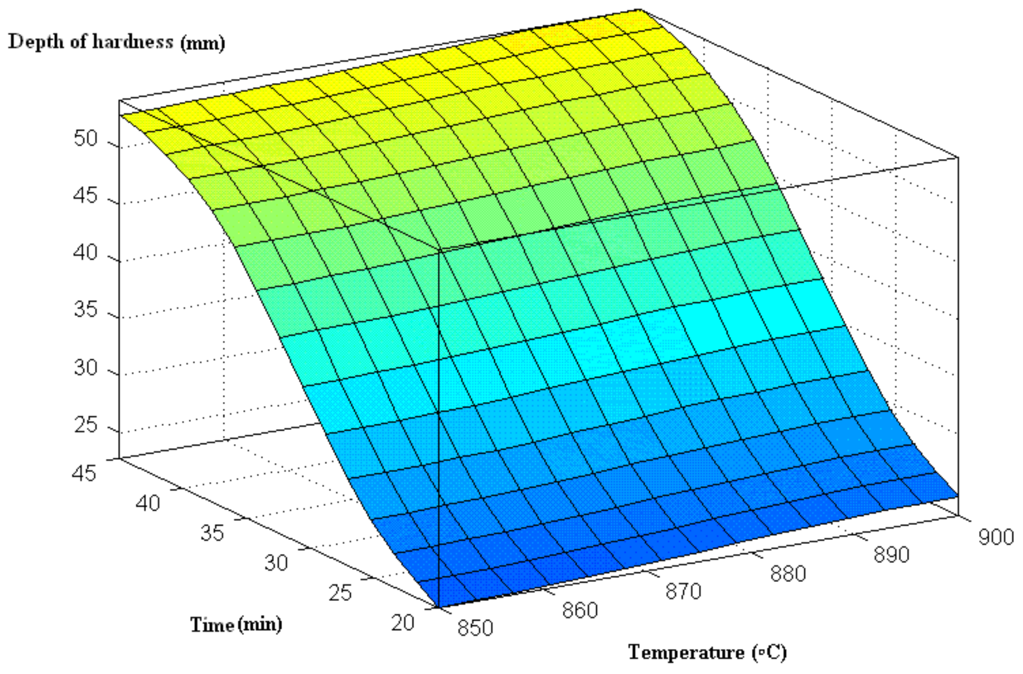


Figure 6.9 NF surface for temperature and time inputs and the output is the depth of hardness.

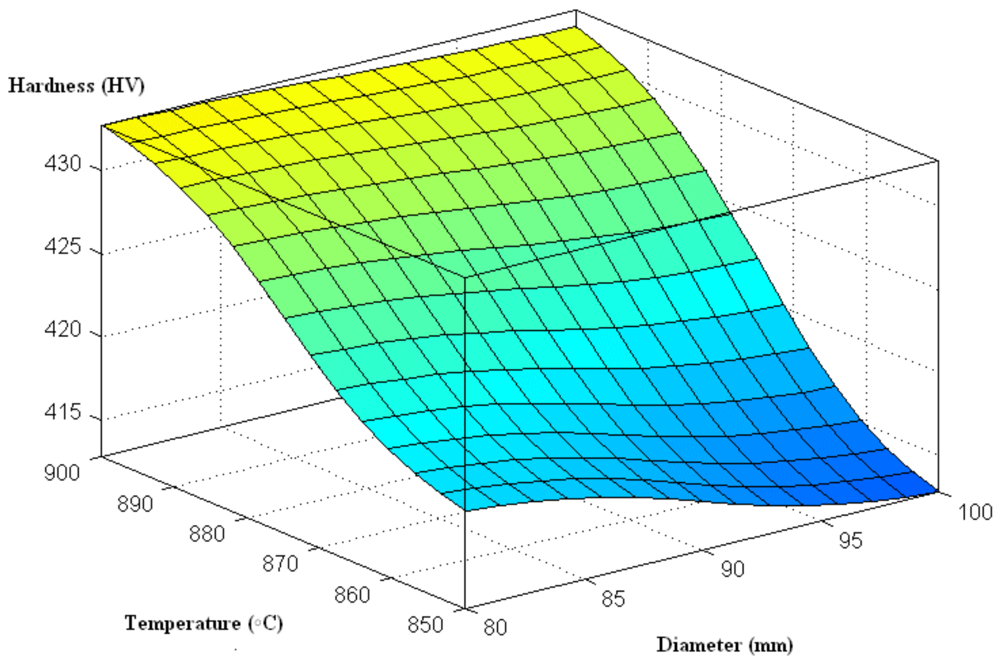


Figure 6.10 NF surface for temperature and diameter inputs and the hardness output.

The NF model deals with deferent types of steel which means different composition. In order to develop this model that is capable to give very good prediction accuracy, it is required large amount of data points to develop this model. For this reason, several NF models have been developed for different types of steel which are considered in this work. Two NF models are developed for each type of steel, one model for predicting the hardness while the other one for predicting the depth of hardness. The examples of NF models rules are shown in Figures 6.10 and 6.12. The figures of the NF models rules are shown in index A. NF models are designed for four types of steel. Table 6.4 outline the steel's compositions that are considered in this work.

Table 6.4 Steel types their components.

Steel Type	C wt %	Si wt%	Mn wt%	Cr wt%	Mo wt%	Ni wt%
14 Ni Cr 14	0.14	0.25	0.55	0.75	0	3.25
16Mn Cr5	0.16	0.2	1.15	0.95	0	0
30 Cr Ni Mo 8	0.30	0.2	0.45	2.0	0.40	2.0
20 Mn45	0.20	0.25	0.45	0	0	0

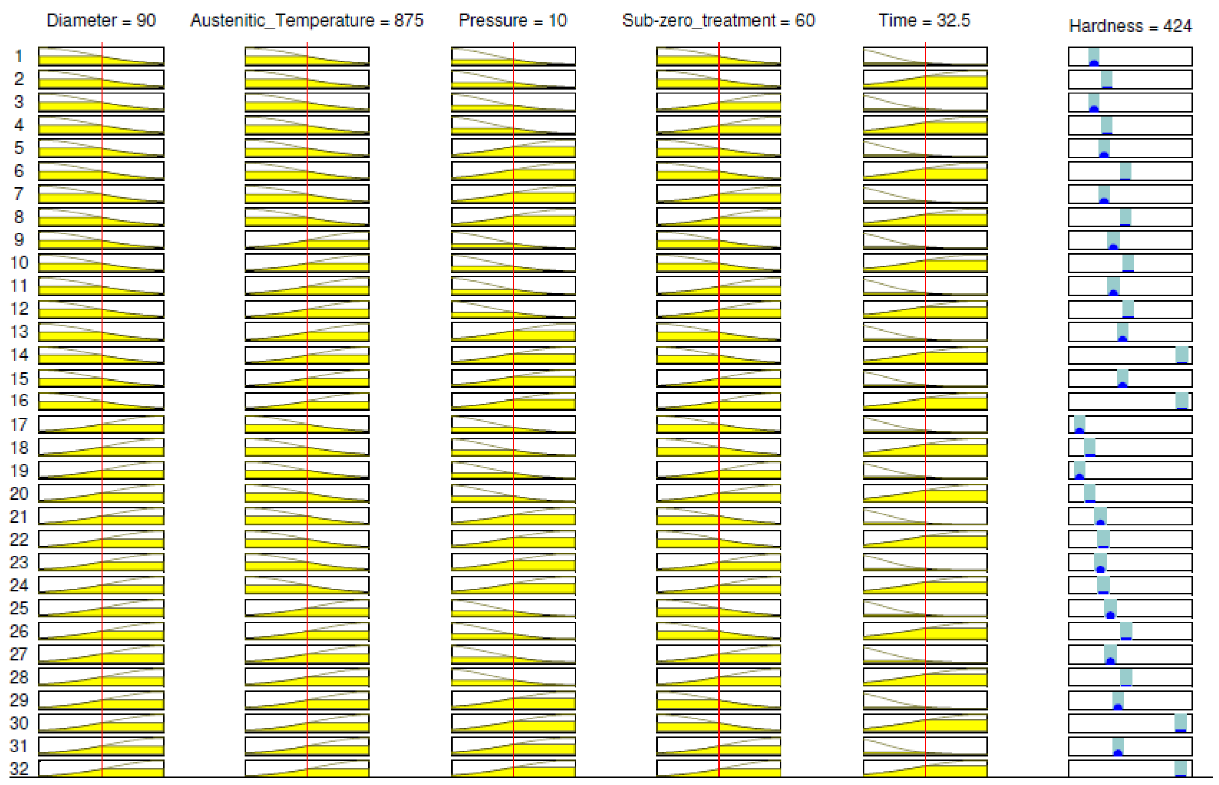


Figure 6.11 NF model rules for predicting steel's hardness.

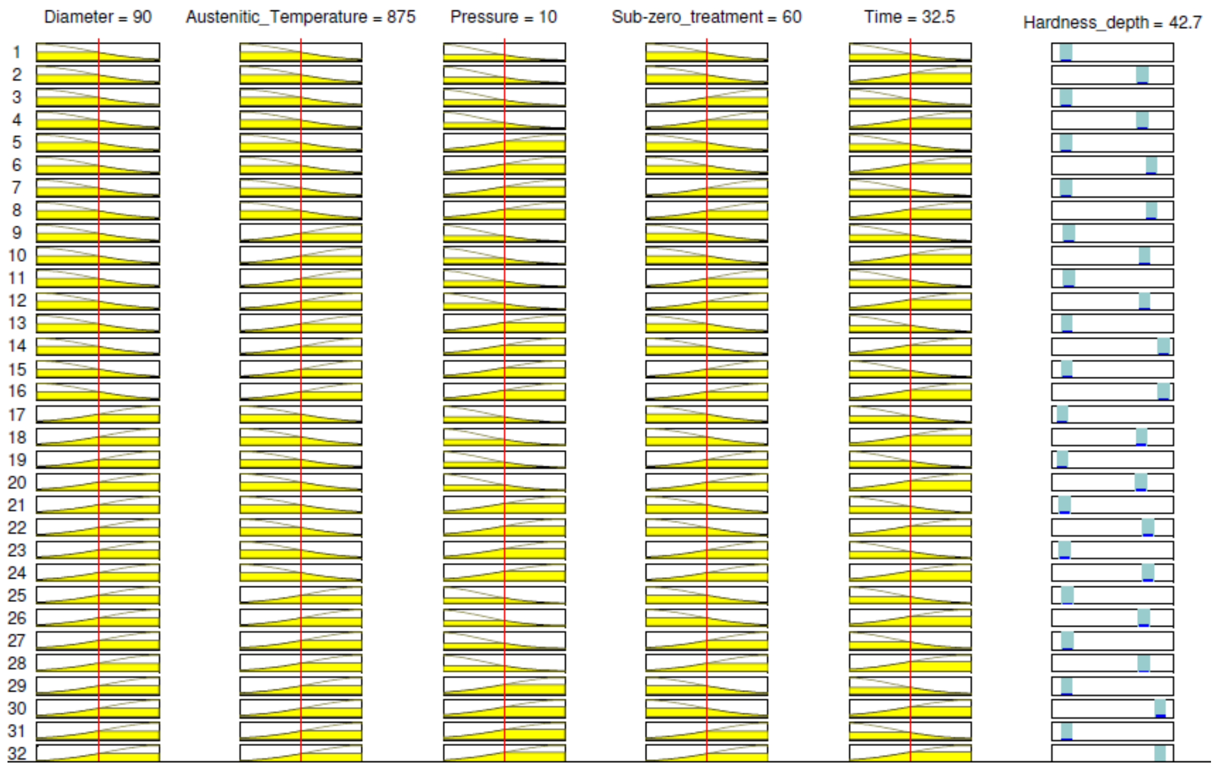


Figure 6.12 NF model rules for predicting steel's depth hardness.

In order to show the accuracy of each NF model, Figures 6.13 to 6.20 show the comparison between the actual and the predicted values of hardness or its depth that are given by the NF models.

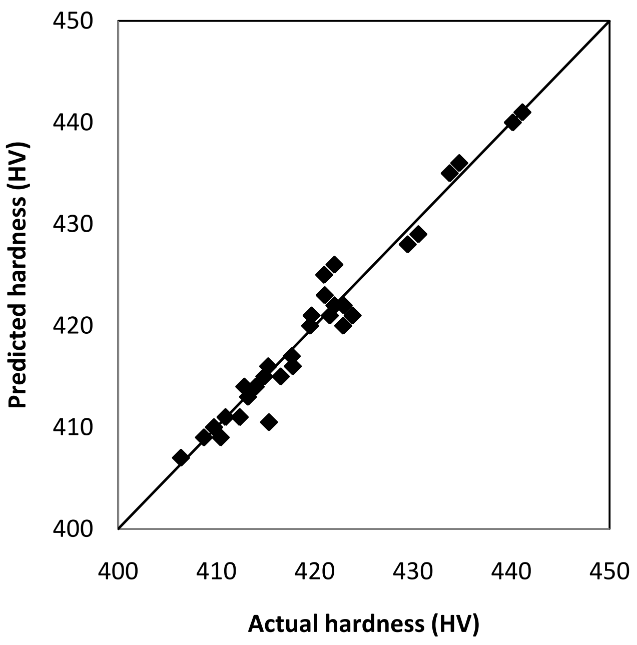


Figure 6.13 Actual and predicted hardness for steel (14 Ni Cr 14).

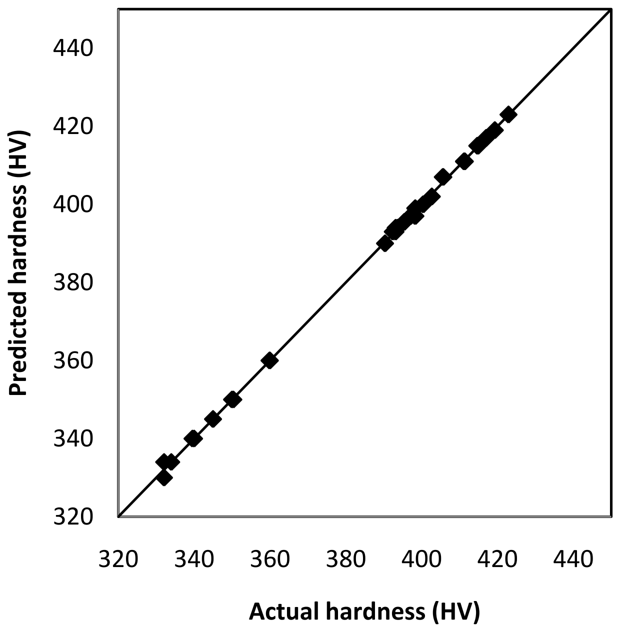


Figure 6.14 Actual and predicted hardness for steel (16Mn Cr5).

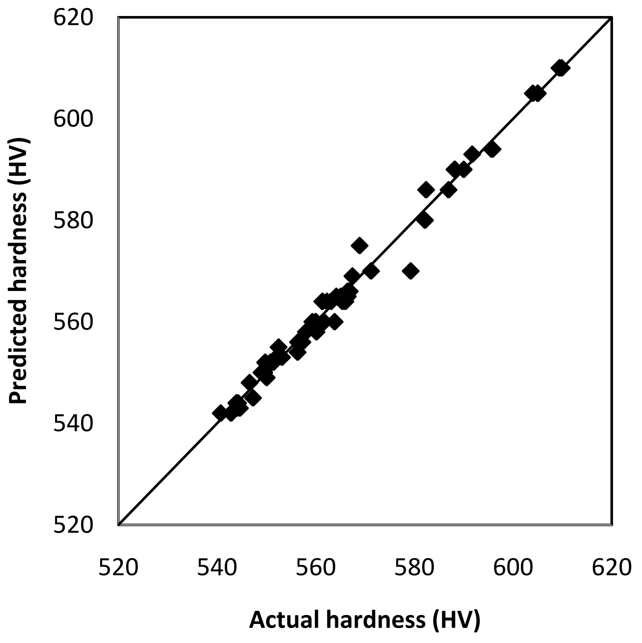


Figure 6.15 Actual and predicted hardness for steel (30 Cr Ni Mo 8).

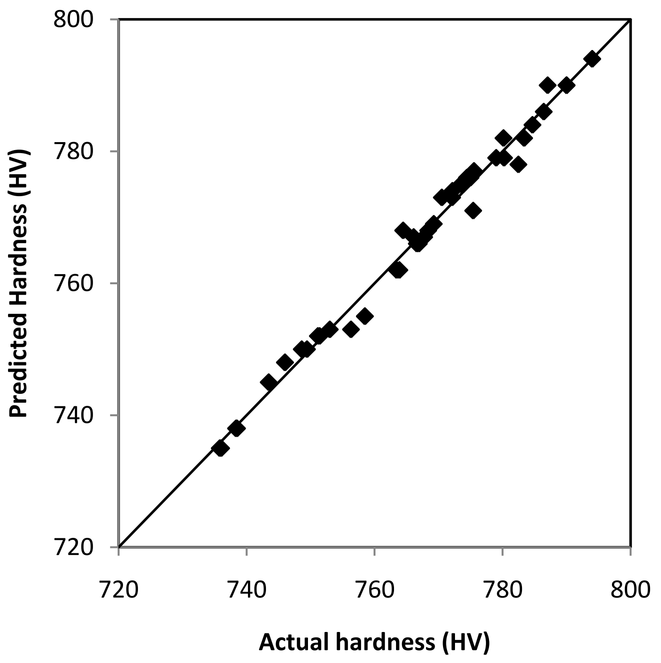


Figure 6.16 Actual and predicted hardness for steel (20 Mn45).

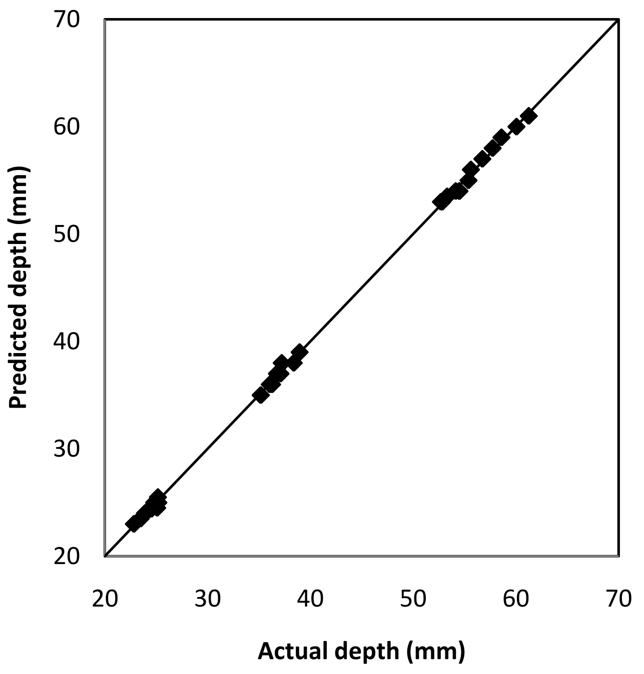


Figure 6.17 Actual and predicted depth hardness for steel (14 Ni Cr 14).

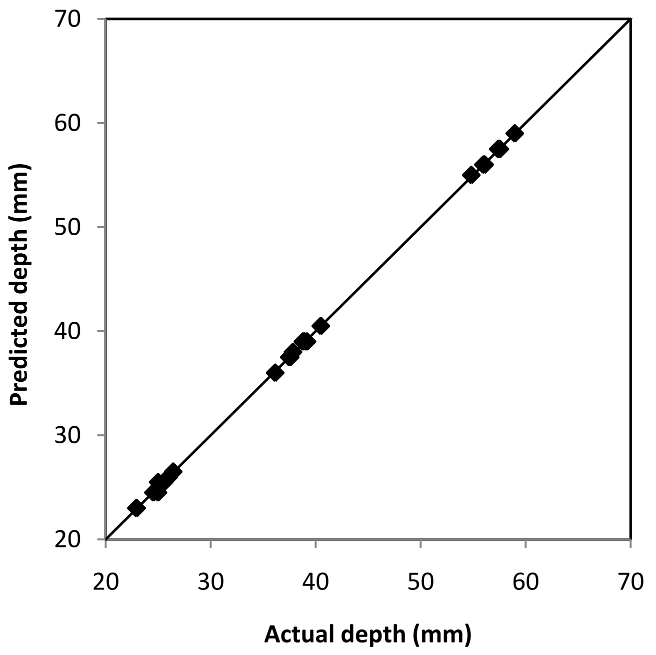


Figure 6.18 Actual and predicted depth hardness for steel (16Mn Cr5).

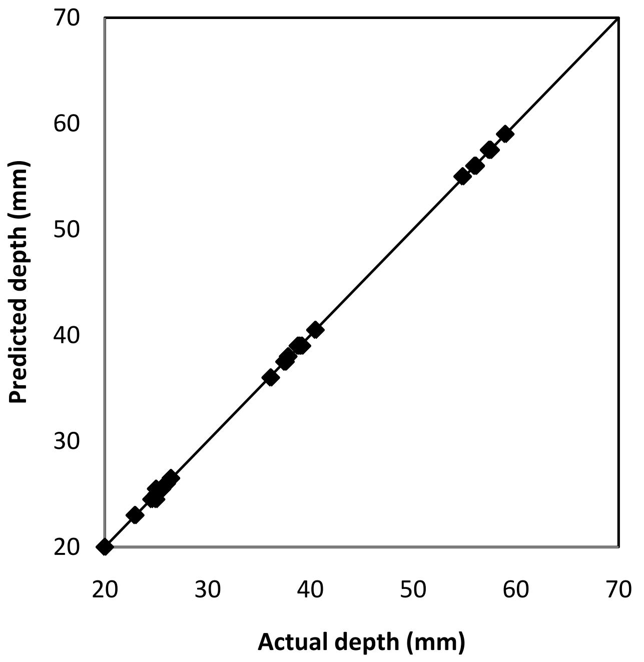


Figure 6.19 Actual and predicted depth hardness for steel (30 Cr Ni Mo 8).

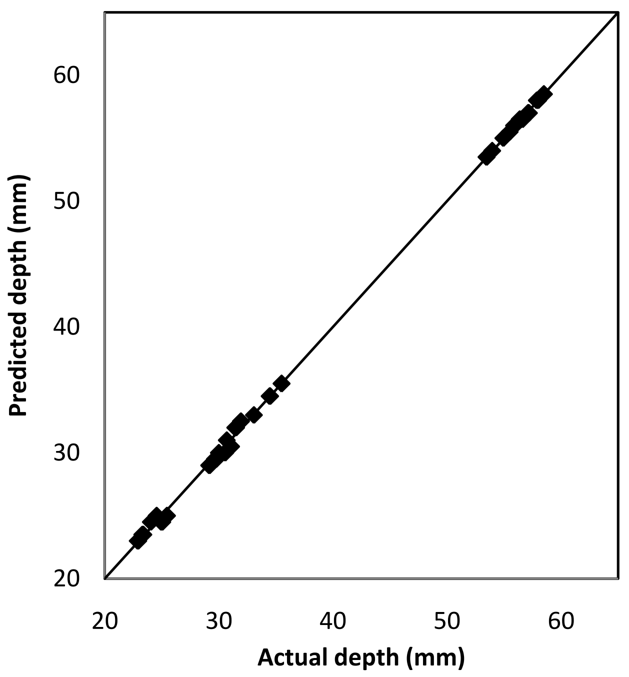


Figure 6.20 Actual and predicted depth hardness for steel (20 Mn45).

The RMS error for each NF model is given in Table 6.5.

Table 6.5 RMS error for NF models.

Steel Type	Model of hardness	Model of depth of hardness
14 Ni Cr 14	1.6366	0.3004
16Mn Cr5	0.6600	0.1404
30 Cr Ni Mo 8	1.7353	0.3082
20 Mn45	1.7830	0.2979

6.4.2 Integrated PSO to NF System

The next step in building the heat treatment system is the integration of the NF systems with PSO methodology [68] [105] so that the system can predict the optimum heat treatment regime. PSO is chosen to be the optimisers because its advantages over other optimisers which are developed in this work such as it is fast convergence to optimal solution and it does not need to encoding or decoding operator for solutions.

PSO is utilised to find the optimal heat treatment conditions which will provide the desired mechanical properties such as hardness and depth of hardness with the aid of the NF model to predict the metal properties. Figure 6.21 outlines how the heating treatment model is integrated using NF model and PSO.

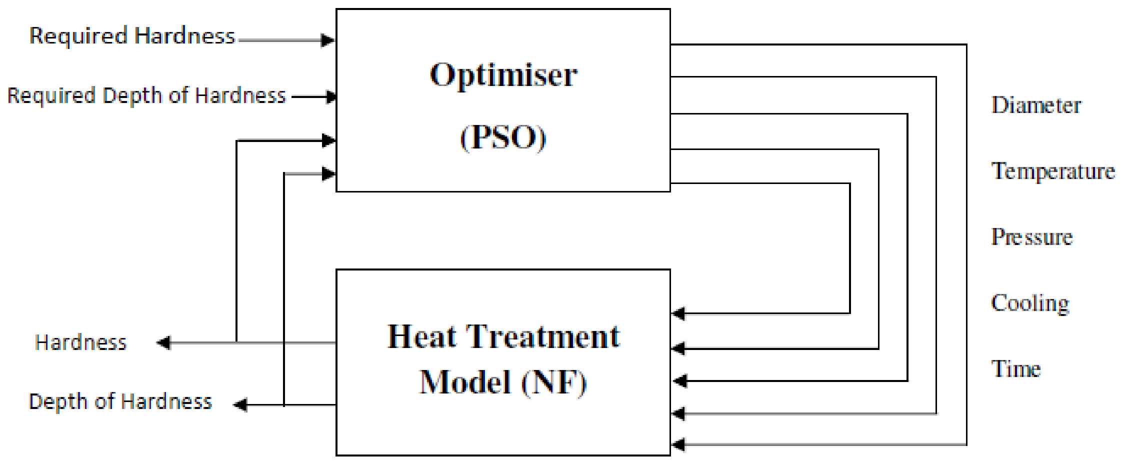


Figure 6.21 Block diagram of integration heating treatment model and PSO.

Accordingly, there are two objectives that need to be achieved, the hardness and its depth which lead to the need of a multi-objective optimisation algorithm. To evaluate both objectives Pareto Front function is used to measure each objective. The results of integrating NF model with PSO are recorded in Table 6.6. The experimental data are used to evaluate this system (see Table 7.1)

Table 6.6 Optimisation results of the NF and PSO using Pareto-Front.

Diameter (mm)	Austenitic Temperature (°C)	Pressure (Pa)	Sub-zero treatment (°C)	Time (min)	Hardness (HV)	Depth of Hardness (mm)	Desired Hardness (HV)	Desired Depth of Hardness (mm)
100	850	8	54	30	408.73	35.19	409	35
60	900	12	70	20	588.13	26.76	590	26.5
120	850	10	59	45	350.21	54.83	350	55
90	900	12	70	28	776.53	31.91	776	32
80	900	10	66	45	434.70	58.60	436	59
60	900	10	50	31	417.38	40.47	417	40.5
60	850	8	50	42	562.07	53.58	560	54
75	850	10	70	45	778.10	55.52	778	56
100	854	12	55	45	416.11	55.77	416	56
60	854	8	50	20	392.98	24.59	393	24.5

In this work, to help the PSO optimiser to find the optimal result, reducing the search space is required. In order to reduce the search space the initial search area is constrained and the optimisers has been restricted to prevent generating unfeasible solutions. In fact, this constrains leads to get accurate result as in Table 6.5 comparing to the requirement by the industry.

6.5 Intelligent System for Heat Treatment Scheduling

The aim of this work is to develop an intelligent system which is capable to provide all requirements of scheduling heat treatment operations such as job heat treatment regime and jobs schedule that is able to give the least energy consumption and the shortest time with consideration to due date time.

In order to realise this system and implement it in industry, the furnace model, the integration of NF model with PSO and GA optimisers are used to build the system. However, all the models that have been developed in this work are utilised and integrated re in this system where each model contributes to the final outcome. Heat treatment model is responsible to provide the system an optimal heat treatment regime for each job. On the other hand, the optimal jobs schedule is supplied to the system by optimisers that their role is searching to find this schedule using furnace model. All requirements that the optimisers need for measuring the object function such as time and fuel consumption for jobs are given by furnace model. Figure 6.22 outlines how the intelligent system integrates the heat treatment system, furnace model and optimisers for giving optimal schedule and heat treatment regime.

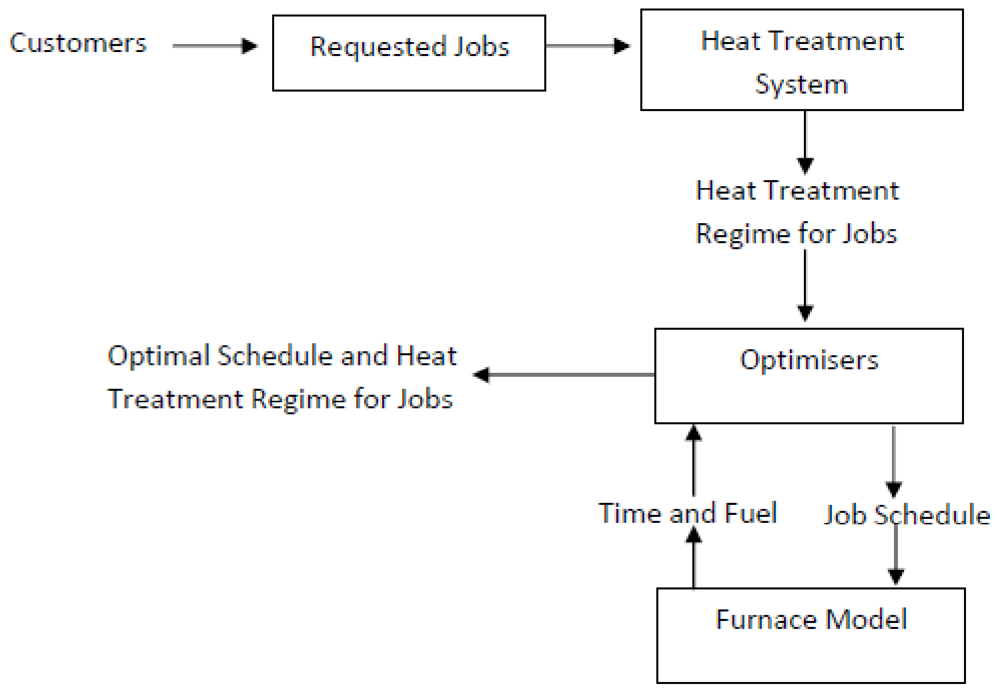


Figure 6.22 Block diagram for the developed intelligent system.

The target of this system is to be a helpful tool for industry where it being capable to provide heat treatment regimes for heat treatment operations that get mechanical properties as is requested. Moreover, this system able to provide jobs schedule which minimises the consumed fuel and time with consideration to the common due date. The mechanism of the system and the outline of the system are illustrated in the next subsection.

6.5.1 The Mechanism of System

The mechanism of this system starts when the customer’s requests received as jobs. It supposes each job to be a type of steel and need to improve its properties by heating treatment operations. Therefore, the system sends information about each job such as type of steel, its components, desired hardness and desired depth of hardness to the heat treatment model which process these information using NF model and PSO to find the optimal heat treatment regime which is capable to give the mechanical properties of steel as the customer request. The jobs heat treatment regimen which is determined by heat treatment model is given to the optimisers by the system. Using these information the optimisers search for the optimal schedule for the jobs that minimising the consumed fuel and time with consideration of common due date where the time and fuel consumption for jobs schedule are provided by furnace model. Furthermore, this work considers common due date where each job finish before or after due date specific penalty is applied for each job. So in this work, there is a function which is working to apply this concept and measuring the time‘s cost function as illustrated in Chapter 4 and 5 by several equations and constraints. In this work, earliness penalty and tardiness penalty for the jobs are given. After the optimisers find the optimal schedule, the system handles this schedule from the optimisers and records it with heating treatment regime for each job to provide them to current customers.

6.6 Summary

In this chapter, some manufacturing information about steel and heating treatment operations are given such as the main types of steel and the main types heat treatment operation. To simulate the heat treatment operations a model of the furnace is developed successfully and is illustrated where an explanation of how this model count the time and fuel consumption are given in details. On the other hand, the NF system and PSO have been developed successfully and working effectively. This system has been able to determine the heat treatment regime that is required for obtaining the desired mechanical properties of steel which is considered as a part of the intelligent model which is developed in this research. The design of this system and its mechanism are illustrated where furnace model, heat treatment model and optimisers integrated to develop this system. Finally, this intelligent system is designed to be helpful tool to the industry in field of scheduling and heating treatment operations of steel and its applications.

Chapter 7   Simulation Results

7.1 Introduction

In this chapter, testing results of the heat treatment model are provided. Furthermore, in this work, several algorithms have been developed. In order to evaluate these algorithms and measuring their capacity, these algorithms have to be tested. So in this chapter, the results of each algorithm are given with some analyses and comparison to other algorithms. Finally, the best algorithm will be chosen to be the recommended intelligent system which is developed by this research. After the development of the intelligent system furnace model, heat treatment model and optimisers, the simulation results of this system have to be analysed and evaluated. So the results discussion for the system whether considering the due date time or without are given.

7.2 Heating Treatment System Simulation Results

In order to evaluate the heat treatment model, different tests have been conducted; the tests emulate different jobs specifications and type, such as 10 jobs, 20 jobs and 50 jobs. The system has been implemented on a PC using Matlab programming language. The furnace model, the materials model and the PSO optimiser has been implemented and tested, then the models were integrated with the optimisers and tested. All the simulations have been conducted on Pentium processor. All the results were recorded and saved under the Matlab environment.

These jobs are selected from the experimental data. Table 7.1 shows the jobs details which have been chosen for the simulation tests.

Table 7.1 Experimental data for the heat treatment model.

Job no.	C wt %	Si wt%	Mn wt%	Cr wt%	Mo wt%	Ni wt%	Diameter (mm)	Hardness (HV)	Depth Hardness (mm)
1	0.14	0.25	0.55	0.75	0	3.25	100	409	35
2	0.3	0.2	0.45	2	0.4	2	60	590	26.5
3	0.16	0.2	1.15	0.95	0	0	120	350	55
4	0.2	0.25	0.45	0	0	0	90	776	32
5	0.14	0.25	0.55	0.75	0	3.25	80	436	59
6	0.16	0.2	1.15	0.95	0	0	60	417	40.5
7	0.3	0.2	0.45	2	0.4	2	60	560	54
8	0.2	0.25	0.45	0	0	0	75	778	56
9	0.14	0.25	0.55	0.75	0	3.25	100	416	56
10	0.16	0.2	1.15	0.95	0	0	60	393	24.5
11	0.3	0.2	0.45	2	0.4	2	90	565	55.5
12	0.2	0.25	0.45	0	0	0	75	771	55
13	0.14	0.25	0.55	0.75	0	3.25	80	411	24
14	0.16	0.2	1.15	0.95	0	0	120	415	57.5
15	0.3	0.2	0.45	2	0.4	2	90	543	24.25
16	0.2	0.25	0.45	0	0	0	60	784	57
17	0.14	0.25	0.55	0.75	0	3.25	80	414	36
18	0.16	0.2	1.15	0.95	0	0	60	407	56
19	0.3	0.2	0.45	2	0.4	2	90	542	23
20	0.2	0.25	0.45	0	0	0	60	738	23.5
21	0.14	0.25	0.55	0.75	0	3.25	80	436	58.5
22	0.16	0.2	1.15	0.95	0	0	120	397	26.5
23	0.3	0.2	0.45	2	0.4	2	60	590	26.5
24	0.2	0.25	0.45	0	0	0	75	773	32
25	0.14	0.25	0.55	0.75	0	3.25	80	441	61
26	0.16	0.2	1.15	0.95	0	0	120	397	26.5
27	0.3	0.2	0.45	2	0.4	2	60	544	23.5
28	0.2	0.25	0.45	0	0	0	75	753	30
29	0.14	0.25	0.55	0.75	0	3.25	100	425	38
30	0.16	0.2	1.15	0.95	0	0	60	411	26
31	0.3	0.2	0.45	2	0.4	2	60	590	26.5
32	0.2	0.25	0.45	0	0	0	60	790	35.5
33	0.14	0.25	0.55	0.75	0	3.25	100	428	38
34	0.16	0.2	1.15	0.95	0	0	60	415	39
35	0.3	0.2	0.45	2	0.4	2	90	605	57.5
36	0.2	0.25	0.45	0	0	0	60	794	58.5
37	0.14	0.25	0.55	0.75	0	3.25	100	435	58
38	0.16	0.2	1.15	0.95	0	0	120	399	39
39	0.3	0.2	0.45	2	0.4	2	60	594	39
40	0.2	0.25	0.45	0	0	0	75	782	56.5
41	0.14	0.25	0.55	0.75	0	3.25	100	440	60
42	0.16	0.2	1.15	0.95	0	0	60	394	25.5
43	0.3	0.2	0.45	2	0.4	2	90	590	38.5
44	0.2	0.25	0.45	0	0	0	60	777	56.5
45	0.14	0.25	0.55	0.75	0	3.25	100	440	60
46	0.16	0.2	1.15	0.95	0	0	60	419	57.5
47	0.3	0.2	0.45	2	0.4	2	60	610	57.5
48	0.2	0.25	0.45	0	0	0	75	775	56
49	0.14	0.25	0.55	0.75	0	3.25	80	410.5	53.5
50	0.16	0.2	1.15	0.95	0	0	120	345	37.5

In order to test the intelligent system model with consideration of the due date time, earliness penalty and tardiness penalty are given to each job. In this work, the penalties are the same as those which are used for single machine benchmarks in Chapter 5. Table 7.2 shows the penalties for specific jobs.

Table 7.2 Jobs penalties.

Job no.	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Earliness penalty	10	3	9	6	10	6	3	6	7	4	6	3	7	4	2	4	4	1	5	8
Tardiness penalty	5	13	7	4	1	15	2	13	10	14	12	4	4	10	13	9	5	7	7	14

Job no.	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40
Earliness penalty	7	5	8	4	3	1	9	9	2	5	4	5	7	1	5	3	6	1	4	1
Tardiness penalty	12	11	9	7	9	6	10	8	12	12	13	11	9	4	2	7	3	11	14	12

Job no.	41	42	43	44	45	46	47	48	49	50
Earliness penalty	7	3	6	6	6	2	10	2	8	8
Tardiness penalty	9	10	5	9	6	10	8	7	9	2

The heat treatment regime for each job is determined by the heat treatment system which contains NF model and PSO. The results of the jobs are given in Table 7.3. The PSO optimiser is set to find the results in 4 iterations based on 20 particles. Although small number of iterations and agents is used, the model could find good results with a small error. In case of the higher accuracy results is required; it is possible to improve the results accuracy by increasing the number of iterations or the number of agents.

Table 7.3 Results of the heat treatment system.

Job no.	Diameter (mm)	Austenitic Temperature (°C)	Pressure (Pa)	Sub-zero treatment (°C)	Time (min)	Hardness (HV)	Hardness Required (HV)	Depth of Hardness (mm)	Depth of Hardness Required (mm)
1	100	850	8	54	30	408.73	409	35.19	35
2	60	900	12	70	20	588.13	590	26.76	26.5
3	120	850	10	59	45	350.21	350	54.83	55
4	90	900	12	70	28	776.53	776	31.91	32
5	80	900	10	66	45	434.7	436	58.6	59
6	60	900	10	50	31	417.38	417	40.47	40.5
7	60	850	8	50	42	562.07	560	53.58	54
8	75	850	10	70	45	778.1	778	55.52	56
9	100	854	12	55	45	416.11	416	55.77	56
10	60	854	8	50	20	392.98	393	24.59	24.5
11	90	850	12	67	45	565.02	565	55.52	55.5
12	75	878	8	61	45	770.966	771	54.92	55
13	80	850	8	70	20	410.92	411	23.94	24
14	120	900	12	70	45	414.9	415	57.4	57.5
15	90	873	8	63	21	543.057	543	24.14	24.25
16	60	850	11	50	45	783.63	784	56.89	57
17	80	850	8	50	29	413.64	414	34.27	36
18	60	850	10	70	45	405.54	407	55.97	56
19	90	850	8	58	20	542.11	542	23.02	23
20	60	850	8	50	20	738.34	738	23.38	23.5
21	80	900	10	50	45	434.71	436	58.6	58.5
22	120	900	12	70	20	397.1	397	26.44	26.5
23	60	900	12	70	20	588.13	590	26.76	26.5
24	75	897	10	50	30	771.9	773	33.0	32
25	80	894	12	70	45	440.06	441	61	61
26	120	900	12	70	20	397.10	397	26.44	26.5
27	60	850	8	63	20	544	544	23.48	23.5
28	75	866	9	50	30	755.80	753	30.1	30
29	100	890	10	70	30	427.52	425	38.14	38
30	60	900	8	50	20	411.15	411	26.022	26
31	60	900	12	70	20	588.13	590	26.76	26.5
32	60	900	12	50	30	790	790	35.5	35.5
33	100	890	10	70	30	427.52	428	38.15	38
34	60	899	8	50	30	414.55	415	38.95	39
35	90	900	12	50	45	605	605	57.14	57.5
36	60	900	12	50	45	794	794	58.5	58.5
37	100	900	10	50	45	433.73	435	57.74	58
38	120	900	10	50	30	398.31	399	39.18	39
39	60	900	12	70	30	595.63	594	38.74	39
40	75	856	11	65	45	782.08	782	56.48	56.5
41	100	900	12	70	45	440.16	440	60.06	60
42	60	850	10	70	20	393.93	394	25.33	25.5
43	90	900	12	70	30	590	590	38.19	38.5
44	60	900	8	70	45	775.53	777	56.38	56.5
45	100	900	12	50	45	440.15	440	60.06	60
46	60	900	10	70	45	419.29	419	57.54	57.5
47	60	900	12	66	45	609.48	610	57.52	57.5
48	75	883	9	70	45	775.35	775	55.8	56
49	80	850	8	70	45	415.35	410.5	53.33	53.5
50	120	850	10	69	30	345	345	37.5	37.5

7.3 Intelligent System Results

The intelligent system developed in this work is based on different algorithms and several optimisers, different types of crossover and fitness function techniques. To compare and analysis the results of each algorithm, simulation of each technique was undertaken. In order to achieve a fair analysing and comparison, the same number of jobs, same number of individuals and same number of iterations was run for each algorithm. The same parameters of optimality and robustness will be used for discussing the results of each algorithm. The simulations of the intelligent system are divided into two main sections. The first section is applied without consideration to the due date time while in the second section the due date time is considered.

7.3.1 System Results without Due Date Time

In this section, the simulation results of the intelligent system using different algorithms are provided without consideration to due date time. The results of each algorithm are recorded in subsection where the results are divided based on the techniques used such as the type of the crossover and the method of measuring the function for individuals.

7.3.1.1 GA Results

The classic GA simulation results are given in Table 7.4 where 10, 20 and 50 scheduling jobs problems are chosen to test the algorithm. Both techniques Pareto Front (PF) and Weighted Sum Method (WSM) used to measure the quality of each individual (solution). The results of GA are recorded in Table 7.4

Table 7.4 GA optimisation results.

crossover	PF			WSM		
	10 jobs	20 jobs	50 jobs	10 jobs	20 jobs	50 jobs
Multi	285.5746	512.3784	1253.326	285.6459	512.6619	1254.293
Binary	285.8313	519.9241	1268.059	275.7552	517.1907	1269.102
Symbol	285.6174	523.117	1257.04	285.6726	512.574	1266.111

The best result of the system for 10 jobs scheduling problem is done using binary crossover and WSM. The system using multi crossover and PF method could achieve the best result for 20 and 50 job scheduling problems. In case of using PF method, the best results are achieved by the system using multi crossover for 10, 20 and 50 jobs problems while the system using symbol crossover has achieved the best result for 20 jobs scheduling problems when using WSM.

7.3.1.2 AGA Results

The simulation results of AGA are given in Table 7.5 where the results are distributed based on algorithms which have been applied such as cost function measurement method and crossover type

Table 7.5 AGA optimisation results.

crossover	PF			WSM		
	10 jobs	20 jobs	50 jobs	10 jobs	20 jobs	50 jobs
Multi	285.575	498.384	1271.942	275.371	506.208	1263.427
Binary	285.5746	509.049	1259.375	274.887	506.431	1283.433
Symbol	285.602	512.316	1266.261	285.602	498.384	1266.216

Using AGA and binary crossover with WSM method, the system could determine the best result for 10 jobs scheduling problem while both algorithms using multi crossover with PF and symbol crossover with WSM, have achieve the best result for 20 jobs scheduling problem. For 50 jobs scheduling problem, the best result is found using binary crossover and PF method. In general using WSM technique gives better results than PF in case of using AGA optimiser.

7.3.1.3 GACD Results

The simulation results of the system using GACD optimiser are recorded in Table 7.6 where different techniques have been used such as PF or WSM for measuring the fitness function of each solution.

Table 7.6 GACD optimisation results.

crossover	PF			WSM		
	10 jobs	20 jobs	50 jobs	10 jobs	20 jobs	50 jobs
Multi	285.575	512.272	1261.818	285.548	509.211	1262.385
Binary	285.548	504.866	1276.042	285.575	510.553	1262.101
Symbol	274.887	509.127	1255.053	285.548	512.570	1279.010

The results using GACD show that using multi crossover gave less robustness where the system got the worse result for 10 and 20 jobs using PF method. The best result for 50 and 10 jobs problems are determined by the system using symbol crossover and PF. Using binary crossover and PF the system could determine the best result for 20 jobs scheduling problem. It can be noticed that PF and symbol crossover gave better performance using GACD.

7.3.1.4 SGA Results

The simulations results for 10, 20 and 50 jobs scheduling problems using SGA algorithms are shown in Table 7.7.

Table 7.7 SGA optimisation results.

crossover	PF			WSM		
	10 jobs	20 jobs	50 jobs	10 jobs	20 jobs	50 jobs
Multi	285.548	509.049	1263.552	274.887	504.894	1263.515
Binary	285.575	504.977	1284.053	275.371	506.286	1282.608
Symbol	274.913	516.841	1279.133	285.602	517.147	1280.828

The best performance using SGA allows determining the best solution for 50, 20 and 10 jobs scheduling problems. Using multi crossover and WSM method the best results for the system have been achieved. Hence, it is clear that multi crossover and WSM have been improved the capability of SGA.

7.3.1.5 MGA Results

The results of the system using MGA methodology that contains a global searcher (GA) and a local searcher (PSO) are given in Table 7.8.

Table 7.8 MGA optimisation results.

crossover	PF			WSM		
	10 jobs	20 jobs	50 jobs	10 jobs	20 jobs	50 jobs
Multi	274.886	505.290	1268.727	275.030	498.445	1250.460
Binary	275.326	520.720	1268.376	274.941	504.480	1263.649
Symbol	274.967	512.346	1273.261	275.030	509.090	1252.410

The simulation results show that multi crossover has improved the performance of the system using MGA where the best results are achieved. For example the best result for 50 jobs scheduling problem is determined using multi crossover and WSM method while the best results for 10 and 20 jobs scheduling problems have been using multi crossover and PF technique.

7.3.1.6 AP Results

The results of the intelligent system applying AP methodology for 10, 20 and 50 jobs problems have been recorded in Table 7.9.

Table 7.9 AP optimisation results.

crossover	PF			WSM		
	10 jobs	20 jobs	50 jobs	10 jobs	20 jobs	50 jobs
Multi	274.886	498.384	1252.853	274.886	498.384	1238.636
Binary	274.886	498.384	1243.099	274.886	498.411	1253.551
Symbol	274.886	509.143	1245.792	274.886	509.103	1255.792

As shown in Table 7.6, all AP techniques could find the same result which is the best known result for 10 jobs. Three techniques of AP could determine the best result for 20 jobs using multi crossover with PF or WSM and AP using binary crossover with WSM method.

7.3.2 Results Discussion

Several algorithms and several techniques have been used for the intelligent system. However, in order to evaluate them, a comparison between those algorithms and techniques is done based on their final results. The aim from this evaluation is to provide a good performance of the intelligent system for scheduling heat treatment jobs.

Table 7.10 show the result of all algorithms using multi crossover, symbol crossover and binary crossover. The results show the capably of these algorithm to determine good results. Not necessary to be the optimal one. In this work evaluating each algorithm individually is done based on compression to other results. The results are measured depend on PF method. In order to explain how the PF method is applied, Table 7.11 and Figure 7.1 show an example of 7 jobs schedule.

Table 7.10 Cost function results using PF.

optimiser	Multi Crossover			Symbolic Crossover			Binary Crossover		
	10 jobs	20 jobs	50 jobs	10 jobs	20 jobs	50 jobs	10 jobs	20 jobs	50 jobs
GA	285.574	512.378	1253.326	285.617	523.117	1257.040	285.831	519.924	1268.059
AGA	285.574	498.384	1271.942	285.602	512.315	1266.261	285.574	509.049	1259.375
GACD	285.574	512.272	1261.818	274.886	509.126	1255.053	285.547	504.865	1276.042
SGA	285.547	509.048	1263.552	274.913	516.840	1279.133	285.574	504.976	1284.053
MGA	274.886	505.289	1268.727	274.967	512.346	1273.261	275.326	520.720	1268.376
AP	274.886	498.384	1252.853	274.886	509.103	1245.792	274.886	498.384	1243.099

Table 7.11 Results of seven jobs schedules.

Schedule	Fuel (kg)	Time (min)
A	2306.723	484.962
B	2308.326	475.4297
C	2313.563	473.268
D	2315.035	483.7419
E	2316.907	486.2013
F	2323.723	496.9256
G	2309.642	493.8532

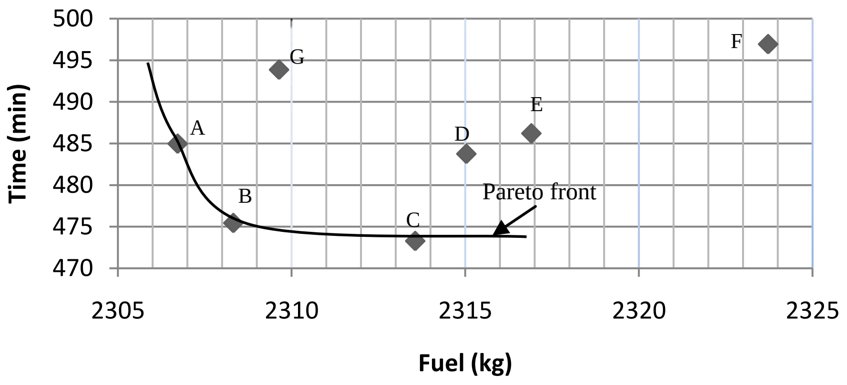


Figure 7.1 Results seven jobs using PF.

In this example, it is clear that schedules A, B and C dominate other schedules G, D, E and F. However as a result, there is no single solution in this method that is leading to choose the best solution according to the priority of each objective function. In this research, each objective is given the same priority.

Table 7.12 to 7.14 shows the time and fuel consumption of each algorithm and the generation in which the optimisers could determine the best result. In fact a good optimisation has been achieved by the optimisers. For example time and fuel consumption of random schedule for 50 jobs are 5789.5 kg, 1480.6 min that can be reduced to 5622 kg of fuel and 1208.5 minutes by the optimisers.

Table 7.12 Fuel and time optimisation results for 10 jobs using PF.

optimiser	Multi Crossover			Symbolic Crossover			Binary Crossover		
	fuel (kg)	time (min)	iteration no.	fuel (kg)	time (min)	Iteration no.	fuel (kg)	time (min)	iteration no.
GA	1273.202	281.785	3	1273.442	281.816	3	1274.613	281.977	1
AGA	1273.202	281.785	2	1273.748	281.816	3	1272.967	281.785	19
GACD	1273.202	281.785	1	1273.509	260.393	20	1272.967	281.785	16
SGA	1273.202	281.785	1	1273.509	260.393	9	1273.202	281.785	1
MA	1273.274	260.392	19	1273.989	260.393	11	1275.517	260.762	2
AP	1273.274	260.392	16	1273.274	260.393	13	1273.274	260.392	3

Table 7.13 Fuel and time optimisation results for 20 jobs using PF.

optimiser	Multi Crossover			Symbolic Crossover			Binary Crossover		
	fuel (kg)	time (min)	iteration no.	fuel (kg)	time (min)	iteration no.	fuel (kg)	time (min)	iteration no.
GA	2322.893	496.891	34	2323.061	518.266	32	2331.183	510.034	22
AGA	2307.090	472.430	35	2322.292	496.860	36	2309.651	496.323	13
GACD	2321.814	496.860	33	2307.497	493.822	19	2314.365	483.740	39
SGA	2307.228	493.836	23	2314.883	507.572	36	2315.23	483.764	36
MA	2316.467	484.110	27	2322.468	496.860	29	2334.511	510.870	38
AP	2307.090	472.430	21	2307.645	493.822	7	2307.09	472.430	37

Table 7.14 Fuel and time optimisation results for 50 jobs using PF.

optimiser	Multi Crossover			Symbolic Crossover			Binary Crossover		
	fuel (kg)	time (min)	iteration no.	fuel (kg)	time (min)	iteration no.	fuel (kg)	time (min)	iteration no.
GA	5631.314	1226.807	89	5640.146	1232.228	95	5653.216	1251.297	99
AGA	5646.318	1260.629	100	5650.666	1248.279	103	5645.874	1235.598	109
GACD	5635.231	1242.902	102	5625.225	1231.645	108	5651.368	1267.682	101
SGA	5640.991	1245.061	100	5652.989	1273.495	106	5675.102	1278.311	103
MA	5640.779	1255.459	98	5658.271	1260.552	97	5654.996	1251.525	100
AP	5629.344	1226.309	76	5623.957	1213.413	86	5622.103	1208.448	88

The results of the system using WSM for measuring the fitness of each solution are shown in Table 7.15. The results are distributed based on the type of crossover and the algorithm of the optimiser that have been used for each case.

Table 7.15 Cost function results using WSM.

optimiser	Multi Crossover			Symbolic Crossover			Binary Crossover		
	10jobs	20jobs	50 jobs	10jobs	20jobs	50 jobs	10jobs	20jobs	50 jobs
GA	285.645	512.661	1254.293	275.755	517.190	1269.102	285.672	512.574	1254.94
AGA	275.370	506.207	1263.427	285.602	498.384	1266.216	274.886	506.431	1283.433
GACD	285.547	509.210	1262.385	285.547	512.570	1279.010	285.574	510.552	1262.101
SGA	274.886	504.893	1263.515	285.602	517.147	1280.828	275.370	506.285	1282.608
MGA	275.030	498.444	1250.460	275.030	509.089	1252.410	274.941	504.479	1263.649
AP	274.886	498.384	1238.636	274.886	498.384	1245.792	274.886	498.411	1253.551

Using WSM the system could achieve good minimisation for the time and fuel consumption. Tables 7.16, 7.17 and 7.18 show the results of the time and fuel consumption for the system using different algorithms types.

Table 7.16 Fuel and time optimisation results for 10 jobs using WSM.

optimiser	Multi Crossover			Symbolic Crossover			Binary Crossover		
	fuel (kg)	time (min)	iteration no.	fuel (kg)	time (min)	iteration no.	fuel (kg)	time (min)	iteration no.
GA	1273.829	281.785	17	1277.952	261.067	2	1274.065	281.785	6
AGA	1275.905	260.763	2	1273.445	281.785	16	1273.274	260.393	4
GACD	1272.967	281.785	3	1272.967	281.785	9	1273.202	281.785	4
SGA	1273.274	260.393	9	1273.445	281.785	10	1275.905	260.763	2
MGA	1274.320	260.443	4	1274.32	260.443	4	1273.753	260.393	4
AP	1273.274	260.393	2	1273.274	260.393	8	1273.274	260.393	7

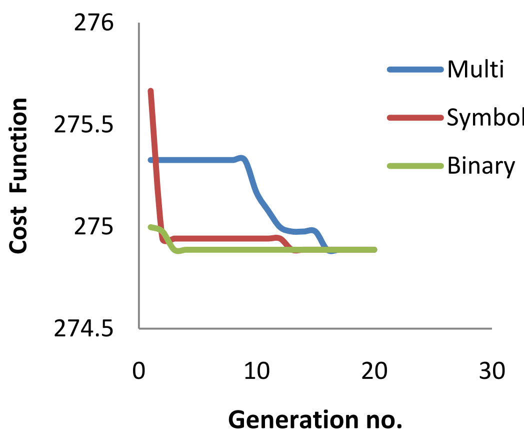
Table 7.17 Fuel and time optimisation results for 20 jobs using WSM.

optimiser	Multi Crossover			Symbolic Crossover			Binary Crossover		
	fuel (kg)	time (min)	iteration no.	fuel (kg)	time (min)	iteration no.	fuel (kg)	time (min)	iteration no.
GA	2324.595	497.007	15	2317.503	507.676	5	2324.254	496.909	3
AGA	2315.520	486.161	39	2307.407	493.853	21	2316.947	486.284	33
GACD	2308.008	493.875	35	2324.123	496.931	19	2309.119	496.306	17
SGA	2314.609	483.740	34	2317.119	507.676	14	2316.134	486.1773	36
MGA	2307.521	472.452	12	2307.166	493.822	39	2313.696	483.1189	31
AP	2307.090	472.430	18	2307.090	472.423	34	2307.326	472.430	32

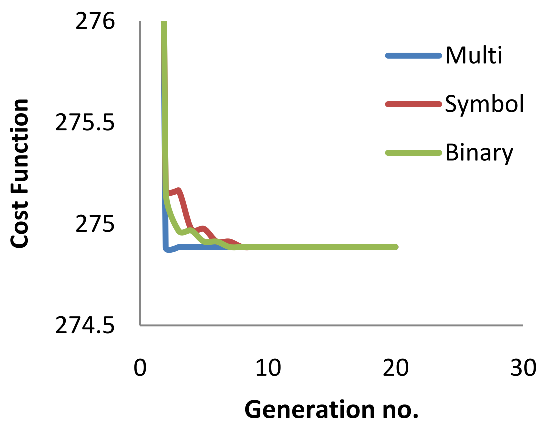
Table 7.18 Fuel and time optimisation results for 50 jobs using WSM.

optimiser	Multi Crossover			Symbolic Crossover			Binary Crossover		
	fuel (kg)	time (min)	iteration no.	fuel (kg)	time (min)	iteration no.	fuel (kg)	time (min)	iteration no.
GA	5637.683	1227.295	94	5652.507	1253.544	54	5637.282	1228.679	83
AGA	5650.344	1242.684	75	5637.712	1251.134	84	5669.667	1278.306	8
GACD	5644.385	1241.955	52	5664.384	1270.661	92	5635.783	1243.343	31
SGA	5647.662	1243.471	66	5666.556	1273.803	54	5659.639	1278.933	57
MGA	5633.388	1220.604	100	5636.167	1223.874	85	5649.646	1247.245	83
AP	5618.612	1200.314	43	5627.756	1232.549	73	5636.694	1226.036	41

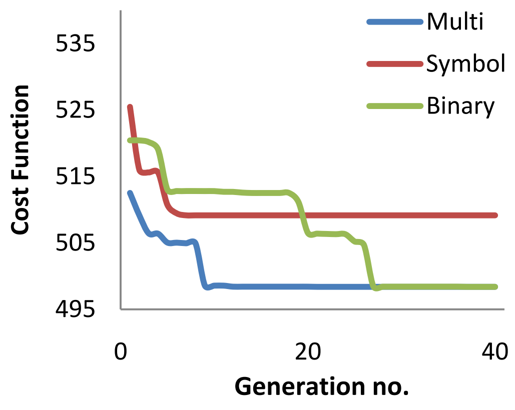
It is clear to notice that the results of the system using AP have outperformed other algorithms where AP could determine the best result for 10, 20 and 50 jobs scheduling problem using both techniques for measuring the fitness of the solutions. In other words, the best intelligent system performance has been achieved using AP algorithm where the most minimisation for time and fuel consumption are achieved. In order to evaluate which AP performance is faster and more accurate Figure 7.2 shows the cost function results of the system using AP optimiser with respect to the generation number.



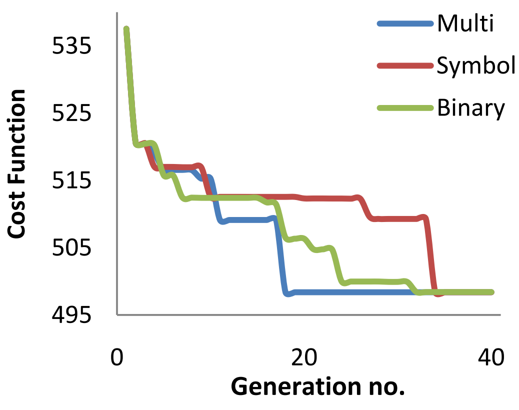
(A) System results using PF for 10 jobs.



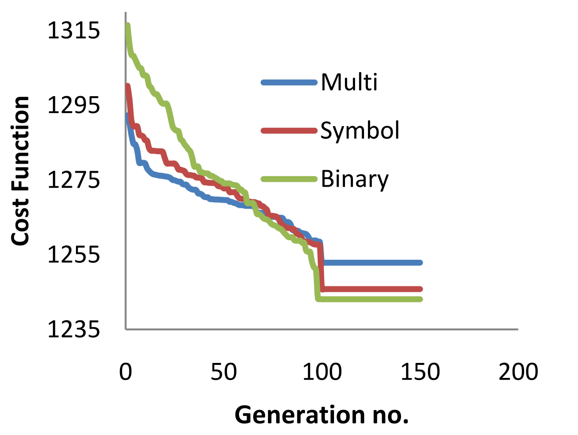
(B) System results using WSM for 10 jobs.



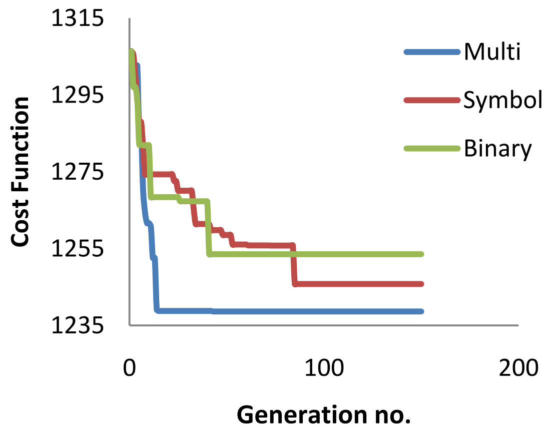
(C) System results using PF for 20 jobs.



(D) System results using WSM for 20 jobs.



(E) System results using PF for 50 jobs.



(F) System results using WSM for 50 jobs.

Figure 7.2 Results of AP optimisation.

There is an increasing in the robustness of the system using AP with multi crossover where it was able to converge to the optimal result faster than other crossovers in all job schedule problems such as 10, 20 and 50 jobs using WSM and 20 jobs using PF except the 10 and 50 jobs using PF. In these cases binary crossover work affectively where it could achieve the best result and it was faster than other crossovers to get the optimal result. In case of using WSM, symbol crossover has improved the accuracy of the AP optimiser to find the optimal result compared to the binary crossover for 50 jobs problem. While in the case of using the PF cost function, the best performance of AP is achieved using the binary crossover where the best results of 10, 20 and 50 job problems were determined.

Other algorithms work affectively and their results are different. For example, in the case of using PF, MGA has determined the same result as AP for 10 jobs problem but it failed to achieve the best for 20 and 50 jobs problems. AGA could achieve the best results for 20 jobs while it could not find the optimal for 10 and 50 jobs problem. GACD could achieve the best for 10 jobs problem and determine the second best result for the 20 and 50 jobs problem.

7.3.3 System Results with Due Date Time

The simulations of the system are run with considering the due date time where the penalties of earliness or tardiness are added to the jobs based on the due date time and finishing time of each job. So it will not be time period but the time cost as a result to the penalties that have been added to each job. Equation 7.1 outline how the time cost is determined.

$$f(s) = \sum^{n} \alpha_i E_i + \sum^{n} \beta_i T_i \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \text{(7.1)}$$

where

$E_i = d - C_i$

$T_i = C_i - d$

$f(s)$ refers to the makespan (cost function) for feasible schedule.

C_i refers to completion time of a job

d refers to due date time

α_i refers to tardiness penalty of a job

β_i refers to earliest penalty of a job

p_i refers to the processing time of a job

7.3.3.1 GA Results

Simulations of the system are conducted using the classical GA and the due date time. All crossovers have been tested and the two techniques of measuring the quality of each solution are tested as well. The simulation results are given in Table 7.19.

Table 7.19 Optimisation Results.

crossover	PF			WSM		
	10 jobs	20 jobs	50 jobs	10 jobs	20 jobs	50 jobs
Multi	1109.708	2579.228	4831.731	1157.019	2926.606	4952.428
Binary	1122.992	2577.710	4831.731	1143.180	2808.326	4822.760
Symbol	1157.687	2776.713	5100.873	1170.309	2798.604	5025.388

In general, the best performance of the system using classical GA is to apply PF and binary crossover where the results show that the best crossover techniques of the system using classical GA optimiser for 20 and 50 jobs problems is binary crossover while the best for 20 jobs problem is multi crossover. It is clear that using PF gives good results where the system can achieve better results in 10 and 20 jobs problems.

7.3.3.2 AGA Results

The results of the system using AGA optimiser are recorded in Table 7.20. The simulations are conducted for each technique individually such as PF and WSM using different types of crossovers.

Table 7.20 Optimisation Results.

crossover	PF			WSM		
	10 jobs	20 jobs	50 jobs	10 jobs	20 jobs	50 jobs
Multi	1117.854	2495.810	5196.445	1131.829	2652.099	5230.619
Binary	1110.206	2535.369	5203.485	1152.416	2634.960	5392.972
Symbol	1116.254	2466.012	4955.351	1171.075	2491.609	5396.638

The best performance of the system using AGA optimiser can be achieved using PF method. For example, the best result achieved for 10 jobs problem has been determined using binary crossover with PF method while the best results for 20 and 50 jobs problems are found using PF and symbol crossover. Generally, the system could make more optimisation using PF method.

7.3.3.3 GACD Results

The system is tested using GACD optimisers and the results are shown in Table 7.21. Several crossovers have been used such as multi, binary, and symbol crossovers. The results are listed based on the method used to measure the fitness of the results.

Table 7.21 GACD optimisation results

crossover	PF			WSM		
	10 jobs	20 jobs	50 jobs	10 jobs	20 jobs	50 jobs
Multi	1120.251	2484.575	4945.21	1127.862	2541.943	5174.248
Binary	1137.722	2613.262	5273.970	1132.175	2564.439	5217.650
Symbol	1130.178	2505.285	5218.488	1180.030	2627.664	5108.957

Using PF and multi crossover the system achieved the best results for 10, 20 and 50 jobs problems. So it can be said that the best performance of the system using GACD optimiser is when PF and multi crossover is used.

7.3.3.4 SGA Results

The results of applying SGA methodology for 10, 20 and 50 jobs problems are given in Table 7.22. The cost function of each solution is measured using PF and WSM. Moreover different types of crossover have been used such as multi, binary and symbol crossover.

Table 7.22 SGA optimisation results.

crossover	PF			WSM		
	10 jobs	20 jobs	50 jobs	10 jobs	20 jobs	50 jobs
Multi	1117.982	2584.274	5195.679	1141.256	2487.998	4996.683
Binary	1122.992	2523.255	5216.448	1154.911	2539.338	5454.463
Symbol	1135.516	2513.691	4975.228	1156.200	2710.855	5193.776

PF gives better performance for the system where the best results for 10 and 50 jobs problems have been found using this method. Multi crossover works effectively and the best results for 10 and 20 jobs problems are determined. Using PF and symbol crossover the system has achieved the best result for 50 jobs problem.

7.3.3.5 MGA Results

The results of system using MGA that employs PSO as a local searcher to improve the optimiser performance are given in Table 7.23

Table 7.23 MGA optimisation results.

crossover	PF			WSM		
	10 jobs	20 jobs	50 jobs	10 jobs	20 jobs	50 jobs
Multi	1122.229	2622.730	5213.439	1122.090	2667.140	4860.360
Binary	1122.992	2523.255	5216.448	1143.180	2636.255	4860.360
Symbol	1130.913	2685.954	5127.735	1143.678	2683.262	4812.878

The best system performance can be achieved using WSM where the MGA optimiser could find the best results for 10 jobs 50 jobs using multi crossover and binary crossover. The best result for 20 jobs problem is achieved using multi crossover and PF method.

7.3.3.6 AP Results

Using the AP methodology as an optimiser, the system simulations results are shown in Table 7.24. PF and WSM techniques with multi, binary and symbol crossover are used and tested.

Table 7.24 AP optimisation results.

crossover	PF			WSM		
	10 jobs	20 jobs	50 jobs	10 jobs	20 jobs	50 jobs
Multi	1109.708	2420.111	4727.586	1109.708	2474.456	4486.274
Binary	1109.708	2459.693	4764.083	1109.708	2426.712	4804.967
Symbol	1109.708	2427.797	4822.907	1116.361	2449.503	4809.201

PF and WSM have been alternated to determine the best results for all scheduling problems that are considered in this work, namely 10, 20 and 50 jobs using multi crossover. For 10 jobs problem all methods could find the best result except in the case of WSM and symbol crossover.

7.3.4 Results Discussion

Considering the due date times the system has been tested using different types of algorithms and has been used for measuring the cost function. Hence their affect on the system has been tested using the algorithms. To evaluate these algorithms and their techniques, a comparison between these algorithms and techniques are done based on their simulation results. The aim from this evolution and comparison is to provide a good performance of the scheduling heat treatment jobs and minimise the fuel and time with considering the due date time.

Tables 7.25 and 7.26 show the results of all algorithms where these results distributed based on the algorithm that has been used.

Table 7.25 PF cost function results.

optimiser	Multi Crossover			Symbolic Crossover			Binary Crossover		
	10 jobs	20 jobs	50 jobs	10 jobs	20 jobs	50 jobs	10 jobs	20 jobs	50 jobs
GA	1109.708	2579.228	4831.731	1157.687	2776.713	5100.873	1122.992	2577.710	4831.731
AGA	1117.854	2495.810	5196.445	1116.254	2466.012	4955.351	1110.206	2535.369	5203.485
GACD	1120.251	2484.575	4945.210	1130.178	2505.285	5218.488	1122.824	2587.240	5216.650
SGA	1117.982	2584.274	5195.679	1135.516	2513.691	4975.228	1137.722	2613.262	5273.970
MGA	1122.229	2622.730	5213.439	1130.913	2685.954	5127.735	1122.992	2523.255	5216.448
AP	1109.708	2420.111	4727.586	1109.708	2427.797	4822.907	1109.708	2459.693	4764.083

Table 7.26 WSM cost function results.

optimiser	Multi Crossover			Symbolic Crossover			Binary Crossover		
	10 jobs	20 jobs	50 jobs	10 jobs	20 jobs	50 jobs	10 jobs	20 jobs	50 jobs
GA	1157.019	2926.606	4952.428	1170.309	2798.604	5025.388	1143.180	2808.326	4822.760
AGA	1131.829	2652.099	5230.619	1171.075	2491.609	5396.638	1152.416	2634.960	5392.972
GACD	1127.862	2541.943	5174.248	1180.030	2627.664	5108.957	1132.175	2564.439	5217.650
SGA	1141.256	2487.998	4996.683	1156.200	2710.855	5193.776	1154.911	2539.338	5454.463
MGA	1122.090	2667.140	4860.360	1143.678	2683.262	4812.878	1143.180	2636.255	4860.360
AP	1109.708	2474.456	4486.274	1116.361	2449.503	4809.201	1109.708	2426.712	4804.967

The results show the capably of these algorithm to determine good optimisation for different sizes of scheduling problems. Indeed, it not necessary to be the optimal one. For 10 jobs problems, the best result is achieved (1109.708) using AP and GA optimisers where AP could find it several times using several techniques while GA could find it once using PF and multi crossover. The best results for 20 and 50 jobs problem are determined among the algorithms using AP. Generally, the system could achieve best performance using AP methodology where the most minimisation for time and fuel consumption is achieved using AP algorithm. In other words, the system was accurate and faster converging to the best result among algorithms. Tables 7.27 to 7.32 show the time cost function and fuel consumption of the system.

Table 7.27 Fuel and time optimisation results for 10 jobs using PF.

optimiser	Multi Crossover			Symbolic Crossover			Binary Crossover		
	fuel (kg)	time (min)	iteration no.	fuel (kg)	time (min)	iteration no.	fuel (kg)	time (min)	iteration no.
GA	1305.950	3379.82	42	1306.253	3733.74	42	1305.937	3478.17	7
AGA	1307.251	3435.29	55	1305.950	3428.26	50	1305.950	3383.51	29
GACD	1306.108	3457.26	20	1323.391	3466.77	12	1306.253	3475.76	39
SGA	1322.488	3379.86	44	1307.156	3566.34	6	1307.156	3582.67	6
MA	1322.330	3411.87	17	1305.937	3536.79	23	1305.937	3478.17	38
AP	1305.950	3379.82	4	1305.950	3379.82	57	1305.950	3379.82	8

Table 7.28 Fuel and time optimisation results for 20 jobs using PF.

optimiser	Multi Crossover			Symbolic Crossover			Binary Crossover		
	fuel (kg)	time (min)	iteration no.	fuel (kg)	time (min)	iteration no.	fuel (kg)	time (min)	iteration no.
GA	2323.905	17007.30	58	2340.567	19277.16	61	2323.601	16990.91	11
AGA	2323.128	16010.95	37	2342.974	15534.30	79	2308.705	16572.21	58
GACD	2323.352	15874.79	67	2324.400	16117.02	85	2309.168	17191.87	98
SGA	2324.673	17063.25	43	2324.641	16216.45	82	2307.228	17515.78	72
MA	2323.692	17530.60	24	2339.289	18195.71	47	2322.408	16344.61	68
AP	2323.641	15099.48	59	2324.321	15187.64	66	2324.610	15568.65	35

Table 7.29 Fuel and time optimisation results for 50 jobs using PF.

optimiser	Multi Crossover			Symbolic Crossover			Binary Crossover		
	fuel (kg)	time (min)	iteration no.	fuel (kg)	time (min)	iteration no.	fuel (kg)	time (min)	iteration no.
GA	5674.466	68211.83	71	5669.231	77505.98	107	5674.466	68211.83	71
AGA	5684.182	80518.92	117	5654.295	72784.55	79	5687.252	80707.18	106
GACD	5667.982	72203.70	91	5693.378	81115.53	96	5700.807	80925.62	57
SGA	5693.910	80326.36	143	5688.113	72886.07	137	5707.975	82763.41	121
MA	5665.552	81418.67	135	5683.037	78188.58	98	5695.132	81015.78	73
AP	5706.152	64164.90	49	5651.429	68304.01	64	5657.426	66189.65	30

Table 7.30 Fuel and time optimisation results for 10 jobs using WSM.

optimiser	Multi Crossover			Symbolic Crossover			Binary Crossover		
	fuel (kg)	time (min)	iteration no.	fuel (kg)	time (min)	iteration no.	fuel (kg)	time (min)	iteration no.
GA	1289.144	3792.11	41	1306.456	3826.40	54	1305.426	3629.46	4
AGA	1305.535	3545.06	4	1307.992	3826.38	9	1307.347	3690.70	32
GACD	1305.742	3514.93	52	1306.945	3896.53	30	1307.042	3542.04	28
SGA	1306.993	3609.42	9	1307.867	3716.78	47	1290.985	3769.70	13
MA	1305.937	3471.50	11	1305.426	3633.14	7	1305.426	3629.46	3
AP	1305.950	3379.82	11	1322.488	3367.87	21	1305.950	3379.82	11

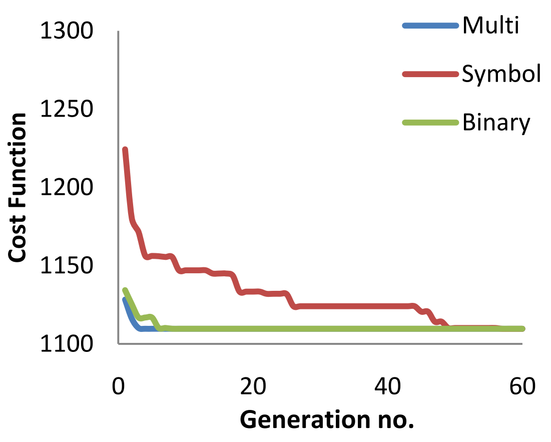
Table 7.31 Fuel and time optimisation results for 20 jobs using WSM.

optimiser	Multi Crossover			Symbolic Crossover			Binary Crossover		
	fuel (kg)	time (min)	iteration no.	fuel (kg)	time (min)	iteration no.	fuel (kg)	time (min)	iteration no.
GA	2357.244	20975.80	81	2350.123	19482.51	81	2348.260	19610.35	30
AGA	2308.252	17975.68	80	2341.320	15851.38	96	2342.270	17565.90	75
GACD	2325.573	16549.88	86	2357.146	17389.09	98	2341.400	16724.87	45
SGA	2324.805	15907.15	80	2340.916	18484.76	29	2350.660	16368.10	73
MA	2339.572	17968.24	81	2366.210	18001.89	52	2339.668	17597.05	92
AP	2325.685	15742.57	94	2340.714	15349.75	93	2324.321	15174.62	90

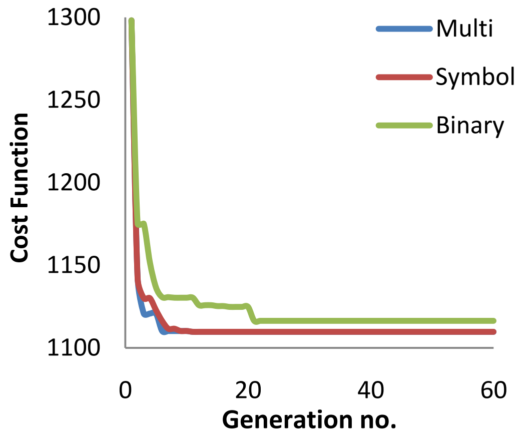
Table 7.32 Fuel and time optimisation results for 50 jobs using WSM.

optimiser	Multi Crossover			Symbolic Crossover			Binary Crossover		
	fuel (kg)	time (min)	iteration no.	fuel (kg)	time (min)	iteration no.	fuel (kg)	time (min)	iteration no.
GA	5685.205	72156.03	137	5709.732	74231.84	108	5661.471	68127.24	137
AGA	5732.825	80855.85	120	5724.302	86679.44	85	5706.864	86852.27	133
GACD	5661.117	80154.16	111	5690.110	77425.45	26	5699.062	80989.68	128
SGA	5714.953	73160.86	139	5699.543	80164.93	84	5745.202	88299.67	61
MA	5666.796	69322.10	91	5698.471	67156.56	145	5666.796	69322.10	153
AP	5686.668	56188.54	119	5661.794	67513.20	142	5661.794	67513.20	149

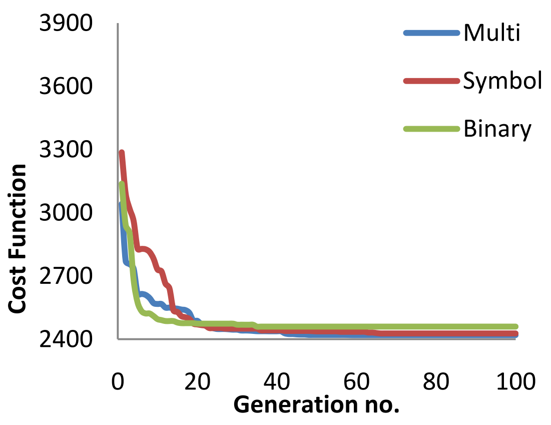
Finally, the results of the system show that AP can give the best performance for the system where the most minimisation has been achieved for time cost function and fuel. Figure 7.3 shows the results of the system using AP methodology.



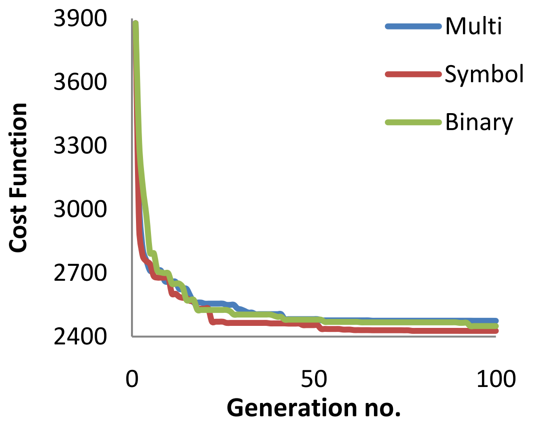
(A) System results using PF for 10 jobs.



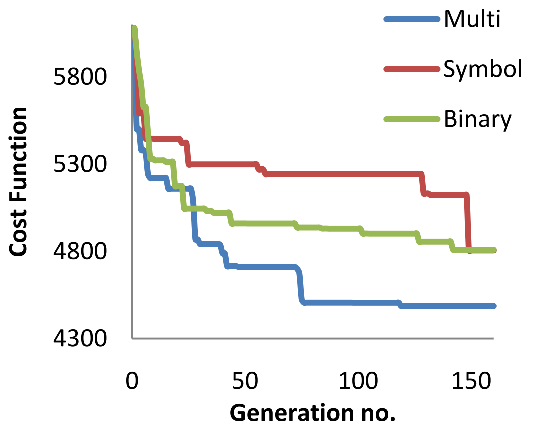
(B) System results using WSM for 10 jobs.



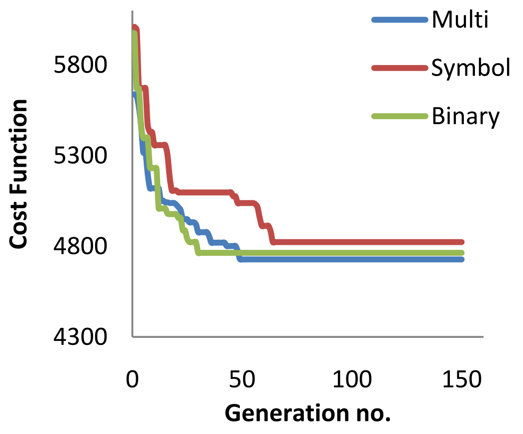
(C) System results using PF for 20 jobs.



(D) System results using WSM for 20 jobs.



(E) System results using PF for 50 jobs.



(F) System results using WSM for 50 jobs.

Figure 7.3 intelligent system results using AP.

AP works affectively using multi crossover where it could to converge the optimal result faster than other crossovers for 10 and 50 jobs problems using WSM or PF. In the case of using PF, AP using binary crossover gives more improvement in the results of the system than another case when using WSM technique. The results of the system using AP with symbol crossover was very slow in 50 jobs problem but in 20 jobs problem was fast and get better accuracy.

7.4 Summary

In this chapter, the experimental data that have been used and the penalties of common due date are given. The results of the test heat treatment system are provided which demonstrated the good accuracy of system.

The intelligent system is implemented based on different algorithms and several optimisers, different types of crossover and fitness function techniques. Tests for each methodology were conducted. The results of the tests show good optimisation have been achieved using different methods. Among all algorithms the system using AP are superior to other algorithms where the best result have been achieved and the biggest amount of optimisation is determined. Moreover, using multi crossover gives more robustness than other crossover. The results of using binary crossover exhibit good capacity in most cases than symbol crossover. The performance of the system using symbol crossover could achieve good optimisation but it is not guaranteed to find the global optimum solution.

Chapter 8 Conclusions and Future Work

8.1 Conclusions

This Chapter presents the main conclusions of this thesis and the future work possibilities. The research works that have been presented in the previous chapters will be summarised and concluded along with the overall achievement to fulfil the aims and objectives of the research. After that, all the possible modification which could give moral or intellectual benefit to the performance of the presented methodologies that are used for this work will be discussed in details as the future work.

This research is focused on the integration problem of process planning and scheduling in steel heat treatment operations environment using artificial intelligence techniques that are capable to deal with such problems rather than on theoretical developments which are as yet to be applied.

In this thesis, the literature review of artificial intelligence applications such as GA, PSO and ACO to scheduling problems such as JSSP, FSSP, OSSP and SMSP are given. The majority of the researches show that a lot of improvement is done using artificial intelligence techniques. Furthermore, the encoding methodology for different types of scheduling problems has played a main role in improving the performance of the scheduling algorithm using artificial intelligent to solve these problems.

In this research the overview of scheduling problems has been given. In more details, the single resource scheduling problems and open shop scheduling problems have been discussed. The complexity of scheduling problems is illustrated which are generally classified as NP-hard problems.

Several algorithms have been developed such as GA, SGA, AGA, GACD, MGA and Algorithm Portfolio to solve scheduling problems that are classified as complex problems. In fact, this complexity is basically caused by various constraints, set of objectives and the size of the search space. These algorithms have been developed successfully using new methodology called scale operation that is capable to decode and represent the solutions for representing an active schedule during the search process. Several effective crossovers have been used to improve the performance of these algorithms. After that these algorithms have been tested using two types of benchmarks, namely open shop benchmarks and single resource scheduling with tardiness and earliest penalties benchmarks. The results of these algorithms demonstrated that they are working effectively where good results can be determined. Several new optimal results have been determined for single machine benchmarks using these algorithms. Among algorithms the results of the Algorithm Portfolio which outperforms other algorithms in the accuracy to find the optimal result and in the speed of convergence to the optimal result. This algorithm is developed using the purpose of embedding the evolutionary algorithms to form portfolios for reducing the generation number that is required in obtaining the optimal solutions. This strategy is useful when good solutions can be found (not necessarily optimal ones) within stipulated time frames. Various types of genetic algorithm have been used in order to develop the Algorithm Portfolio. On the other hand, the results of using scale operation technique for representing and decoding were encouraging where it was able to avoid the system from creating unfeasible solutions and it could improve the performance of the optimisers using binary code which researchers avoids for scheduling problem due to the unfeasible solutions. Instead, symbol is usually used.

Hence, in industry, the regime that is used for conducting the heat treatment operation to realise the specific mechanical properties of steel is very important and not easy to obtain. In this research, a system that is capable to determine the optimal heat treatment regime for different type of steel is developed. This system is implemented using NF and PSO methodologies. NF model is developed to predict the regime while PSO is utilised to find the optimal conditions which will provide the desired mechanical properties such as hardness and depth of hardness with the aid of the NF model to predict the metal properties.

After the heat treatment system model is developed, the system is tested and the tests results show that the system works properly and is able to predicate the optimal regime that leads to achieve the desired mechanical properties of steel. In this research four types of steel are considered. Therefore, several NF models have been developed to allow the system to deal with each type of steel.

In industry heat treatment operation scheduling for steel is highly desired as a result of the consumption of time and fuel that is considered as numerous and costly. To schedule the heat treatment operations, a furnace model has been developed that is capable to provide time and fuel consumption accurately. On the other hand, furnace model designs vary as to its function, heating operations, type of fuel and method of introducing combustion air. The furnace model provides all requirements accurately such as the amount of consumed time and the amount of consumed fuel using standard data. This model has two zones; the first zone is designed for heating the steel until it reaches a critical degree point while the second zone is designed for cooling the materials for applying the heat treatment operation such as quenching and annealing. In the heating zone, cooling system is designed to reduce the furnace temperature to the desired setting point for other products. Dry air system is used to cool the furnace after the products are taken out. Moreover, the furnace model is designed for several different fuels including various types of gas fuel and various grades of oil fuel. The model only needs to know the composition rate of the element where the system calculates the amount of heating using calorific value concept.

Using the optimisers, furnace model and heat treatment system model, the intelligent system model is developed and implemented successfully. In this system the heat treatment system model provides the optimal regime for each steel job that is able to determine the desired mechanical properties of the steel job. After that the furnace model is used to simulate the heat treatment for the jobs and will count the amount of fuel and time consumption for jobs schedule using the heat treatment model that account for austenitic temperature, pressure, time and cooling rate. The optimiser will search for the schedule that gives the least amount of fuel and least consuming time. The due time has been considered in this work where any job that finishes before the due date, the earless penalty is applied and any job that finishes after the due date time, the tardiness penalty is applied. In fact, this leads to make the problem to be hard and more complex.

As a result, this work deals with multi objectives, two methods to measure the quality of the fitness for each solution are selected, namely Pareto Front and Weighted Sum Method. Both techniques are applied successfully and the effect of using these methods on the performance of the algorithms are shown in the results of each algorithm

The testing results of the intelligent system have been given and the performance of each algorithm has been measured. However, in this work, using the AP is recommended as it features the advantages of fast convergence speed and better accuracy. This gives the advantage for the AP to be better. Moreover, using multi crossover gives more robustness than other crossover. The results of using binary crossover exhibit good capacity in most cases than symbol crossover. The performance of the system using symbol crossover could achieve good optimisation but it is not guaranteed to find the global optimum solution. Furthermore, based on the system results the best performance for each algorithm is given and recommended.

8.2 Future Work

Following the investigations described in this thesis, a number of projects could be taken up, involving improving and extending some parts of this work. In the following subsections, extending works are given based on the model function such as the optimisers, the NF model, heat system model, and furnace mode.

8.2.1 Optimisers

Better optimisation can be achieved by enhancing the performance of the optimiser. In order to improve the optimiser performance, there are several techniques can be used such as using effective algorithm. In this thesis, dispatching rules methodologies are suggested where there are different types of rule that can be applied as follows:

- Process-time based rules
- Due-date based rules
- Combination rules
- Rules that are neither process-time based nor due-date based.

In fact, using dispatching rules methodologies with AI such as GA will lead to decrease in the search space which will enhance the performance of the algorithm. Moreover, using heuristic algorithms [86] such as developing a new algorithm that contains different methodology the performance of this algorithm may be improved compared to other algorithms. If this algorithm performance is satisfied, it can be added to Algorithm Portfolio as a new optimiser or instate of one of the optimisers that Algorithm Portfolio hosts.

Furthermore, other new optimisers can be developed using another methodology such as the bacterial foraging optimisation algorithm [97] that is based on biologically inspired computation technique which depends on simulation the foraging behaviour of some types of bacteria. Based on the performance of this algorithm, this algorithm can be used for this work as an individual or it can be added to Algorithm Portfolio optimisers.

Finally, the Algorithm Portfolio can be implemented in parallel with allowing the communication between the optimisers. This technique may lead to an improvement in the performance of algorithm.

8.2.2 NF System Model

In this work several NF systems models have been used to predict the heating treatment regime of steel for each job where two NF systems models have been developed for each type of steel. In fact, the NF system model can be improved using more experimental data of steel heat treatment operation. Based on extra data, one model for different types of steel can be developed. In this case, the composition of steel will be considered in the design of NF system model where the composition of steel will be included with inputs to the model. The experimental data of steel heat treatment should be sufficient to develop an accurate system that is capable to predict the materials properties changing between their compositions. Indeed, this need large amount of data which sometimes is hard to get. Wherefore, a method is needed that is able to minimise the amount of experiment required. Several methods can be used to reduce the data that is required for developing an accurate NF system model for several types of steel. Byrne and Taguchi methodology [14] can be used to reduce the amount of experiments. Taguchi methods can be defined as a statistical technique that is developed based on orthogonal array which gives much reduced variance for the system to get the best settings. In deed this will help to select the data that is capable to give an accurate prediction and dispense some of the data that is not needed.

8.2.3 Furnace Model

In this work, the system considers different types of fuel for generating the heat in the furnace model. Hence this work used fuel as energy resource to furnace model where in industry; there are different types of furnaces. Some of these furnaces use electrical power as energy resource. So the furnace model can be modified to consider the electrical power energy resource. The furnace model can count the net heat available straightforward from the capacity of the heating system where the system can offer amount of heating per unit of time no energy loss in this case during generating the heating.

8.2.4 Benchmarks

In this work several benchmarks have been used to test the algorithms that have been developed. However, still some benchmarks have not been used for testing the system such as 10×10 and 20×20 open shop benchmarks or 100, 200 and 500 jobs of single machine benchmarks. Moreover, although the algorithms could find the optimal solutions for most cases, in some cases some algorithms failed to find the optimal one. It is possible to improve the performance of these algorithms using some ideas that are suggested in this chapter or changing some algorithms parameters setting. After that testing these algorithms is conducted again to know the affect of these algorithms on the benchmarks.

8.2.5 Real Time Implementation

Real time implementation of the system, testing, and validation can part of the future work. This work can be implemented in the industry such as steel manufacturing companies that applies heat treatment operations. This work will verify and validate the capacity of the system in real time. In this case, some parts of the model may need setting or modification such as the furnace model and the heat treatment model.

Real time scheduling problems such as batch scheduling problems [90] and scheduling production orders [113] can be implemented as future work where the algorithms can be applied to such problems. Moreover, these algorithm can also be applied to other hard problems such just-in-time scheduling problems. Just-in-time scheduling problem can be defined as the problem that is determining the best schedule such that each job finishes exactly at its due date time. This problem can be in car or airplane manufacturing. Furthermore, dynamic manufacturing scheduling problems is considered to be realistic real time scheduling problems. Therefore this problem can be solved using the algorithms. However some modification might be required such as prioritisation.

References

[1] K.R. Baker., “Introduction to sequencing and scheduling”, John Wiley & Sons, New York, 1974.

[2] K. Baker and G. Scudder., “Sequencing with earliness and tardiness penalties”, Review, Operations Research, vol 38, pp 22-36, 1990.

[3] J.E Baker., “Reducing bias and inefficiency in the selection algorithm”, In International Conference GA2, pp. 14-21, 1987.

[4] J.E. Baker., “Adaptive selection methods for genetic algorithms”, Proceedings of the 1st International Conference on Genetic Algorithms, In Grefenstette, pp 101-111, 1985.

[5] S. Bandyopadhyay, S.K. Pal, U. Mulak., “Incorporating chromosome differentiation in genetic”, Information Science, vol 104, no 8, pp 293–319, 1988.

[6] R. Bellman., “Dynamic programming”, Princeton University Press, Dover paperback edition, 2003.

[7] V. Bergh, A.P. Engelbrecht., “Cooperative learning in neural network using particle swarm optimisers”, South African Computer Journal, vol 26, pp 84–90, 2000.

[8] D. Biskup, M. Feldmann., “Benchmarks for scheduling on a single machine against restrictive and unrestrictive common due dates”, Computers & Operations Research, vol 28, pp 787-801, 2001.

[9] J.H. Blackstone, D.T. Phillips, G.L. Hogg., “A State of the art survey of dispatching rules for manufacturing job shop operations”, International Journal Production Research, vol 20, pp 27-45, 1982.

[10] J. Blazewicz, J.K. Lenstra and A.R. Kan., “Scheduling subject to resource constraints classification and complexity”, Discrete Applied Mathematics, vol 5, no1, pp 11-24, 1983.

[11] T. Blickle, L. Thiele., “A comparison of selection schemes used in genetic algorithms”, 2nd Edition TIK Report, no 11, Computer Engineering and Communication Networks Lab (TIK), Swiss Federal Institute of Technology (ETH) Zürich, Switzerland, (1995).

[12] J. Breit, G. Schmidt, V.A. Strusevich., “Non-preemptive two-machine open shop scheduling with non-availability constraints”, Math Method, Operation Research, vol 57, pp 217–234 (2003).

[13] P. Brucker., “Scheduling algorithms”, Berlin Heidelberg, Springer, 1995.

[14] D. M. Byrne, S. Taguchi., “The taguchi approach to parameter design”, In Quality Congress Transaction – Anaheim, ASQC, pp 168–177, May 1986.

[15] P. Brucker, J. Hurink, B. Jurish, B. Wostmann., “A branch and bound algorithm for the open-shop problem”, Discrete Applied Mathematics, vol 76, pp 43-59, 1997.

[16] R. Cheng, M. Gen, Y. Tsujimura., “A tutorial survey of job-shop scheduling problems using genetic algorithms”, Part II. Hybrid Genetic Search Strategies, Computers & Industrial Engineering, vol 37, pp 51-55, 1999.

[17] C.S. Chong, M.Y. Low, A.I. Sivakumar, K.L. Gay., “A bee colony optimisation algorithm to job shop scheduling”, Proceedings of the Winter Simulation Conference, 2006.

[18] J. Chuanwen, E. Bompard., “Adaptive chaotic particle swarm algorithm for short term hydroelectric system scheduling”, In Deregulated Environment Energy Conversion and Management, vol 46, pp 2689–2696, 2005.

[19] E.G. Colemman., “Computer and job shop scheduling theory”, Wiley, New York, 1976.

[20] R.W. Conway., “Priority dispatching and job lateness in a job shop”, Journal of Industrial Engineering, vol 16, pp 228-237, 1965.

[21] R.W. Conway, W.L. Maxweel, L.W. Miller. “Theory of scheduling”, Addison-Wesley, Readingg, Mass, (1968).

[22] F.D. Croce, R. Tadel, G. Volta., “A genetic algorithm for the job shop problem”, Computers and Operations Research, vol 22, no 1, pp 15-24, 1995.

[23] S.T. Davidovi, T.G. Crainic., “Benchmark-problem instances for static scheduling of task graphs with communication delays on homogeneous multiprocessor”, Computers & Operations Research, vol 33, pp 2155–2177, 2006.

[24] R. Dawkins., “The selfish gene”, Oxford University Press, 1976.

[25] E. Demirkol, S. Mehta, R. Uzsoy., “Benchmarks for shop scheduling problems”, European Journal of Operational Research, vol 109, pp 137-141, 1998.

[26] Y.V. Deshmukh., “Industrial heating”, Taylor and Frances Group, CRC, 2005.

[27] M. Dorigo. V. Maniezzo, A. Colorni., “The ant system: optimization by a colony of cooperating agents”, IEEE Transactions on Systems, Man & Cybernetics B, vol 26, no 2, pp 29-41, 1996.

[28] M. Dorigo, “Optimisation, learning and natural algorithms”, PhD thesis, Department of Electronics, Politecnico di Milano, Italy, 1992.

[29] J. Dorn, K. Froeschl., “Scheduling of production processes”, Ellis Horwood Limited, 1993.

[30] J. Du, J.Y.T. Leung., “Minimize mean flow time in two-machine open shops and flow shops”, Journal of Algorithms, vol 14, pp 24 – 44, 1990.

[31] A. Duenas, D. Petrovic., “Multi-objective genetic algorithm for single machine scheduling problem under fuzziness”, Fuzzy Optimization and Decision Making, vol 7, pp 87–104, 2008.

[32] R.C. Eberhart, Y. Shi., “Tracking and optimizing dynamic systems with particle swarms”, in: Proceedings of the IEEE Congress on Evolutionary Computation (CEC 2001), IEEE, Seoul, Korea, pp 94–97 (2001).

[33] Editorial., “An introduction to artificial intelligence applications in petroleum exploration and production”, J Petroleum Science and Engineering, vol 49, pp 93– 96, 2005.

[34] S. Eilon, J.R. King., “Industrial scheduling abstracts”, pp 1950-1966, Published by Oliver and Boyd, London in 1967.

[35] S. Eilon., “Production scheduling”, in K.B. Haley, Editor, Operation Research 78, North-Holland, Amsterdam, pp 237-266, 1978.

[36] A. El-Galland, M. El-Hawary, A. Sallam., “Swarming of intelligent particles for solving the nonlinear constrained optimisation problem”, Engineering Intelligent Systems for Electrical Engineering and Communications, vol 9, pp 155–163, 2001.

[37] H. Emmons., “One-machine sequencing to minimize certain functions of job tardiness”, Operations Research, vol 17, pp 701-705, 1969.

[38] H. Fang, P. Ross, D. Corne., "A promising hybrid GA/heuristic approach for open shop scheduling problems", DAI Research Paper, no 699, Proceedings of the 11th European Conference on Artificial Intelligence, John Wiley and Sons, pp 590-594, 1994.

[39] E. Falkenauer and S.Bouffouix. "A genetic algorithm for job shop", proceeding of the IEEE International Conference on Robotics and Automation Sacramento, California, April 1991.

[40] R. Feldman, M.C. Golumbic., "Constraint satisfiability algorithms for interactive student scheduling", Proceedings of the 11th International Joint Conference on Artificial Intelligence, vol 2, pp 1010-1016 (1989).

[41] M. Fischetti, S. Martello, P. Toth., "The fixed job schedule problem with working-time constraints", Operations Research, vol 37, pp 395-403, 1989.

[42] H. Fisher, G. Thompson., "Probabilistic learning combinations of local job-shop scheduling rules", In Industrial Scheduling, J. Muth and G. Thompsoneds, Prentice-Hall, pp 1225-1251, 1963.

[43] H. Fisher and G.L. Thompson., "Probabilistic learning combinations of local job-shop scheduling rules", In Factory Scheduling Conference, Carnegie Institute of Technology, May 1961.

[44] M.R. Garey, D.S Johnson, R. Sethi., "The complexity of flowshop and job shop scheduling", Mathematics Of Operation Research, vol 1, no 2, pp 117-128, 1974.

[45] M.R. Garey, D.S. Johnson., "Computers and intractability a guide to the theory of NP-completeness", Freeman, San Francisco, 1979.

[46] A. Ghosh., S. Tsutsui, H. Tanaka., "Individual aging in genetic algorithms", In: Australian and New Zealand Conference on Intelligent Information Systems, 1996.

[47] F. Glover., "Future Paths for Integer Programming and Links", Artificial Intelligence Computers and Operations Research, vol 13, pp 533-549, 1986.

[48] D.E Goldberg., "Genetic algorithms in search, optimization", and machine learning, Addison-Wesley, Reading, 1989.

[49] D.E. Goldberg, K. Deb., "A comparative analysis of selection schemes used in genetic algorithms", In FGA1, pp 69-93, 1991.

[50] C. Gomes, and B. Selman., “Algorithm portfolios, Artificial Intelligence”, vol 126, no1-2, pp 43-62, 2001.

[51] J. Grabowski, J. Pempera., “Some local search algorithms for no-wait flow-shop problem with makespan criterion”, Computers and Operations Research, vol 32, pp 2197–2212, 2005.

[52] R.E. Graham, E.L. Lawler, J.K. Lenstra, R. Kan., “Optimisation and approximation in deterministic sequencing and scheduling theory a survey”, Annals of Discrete, Mathematics, vol 5, pp 287-326, 1979.

[53] M. Gravel, W.L. Price, C Gagne., “Scheduling continuous casting of aluminium using a multiple objective ant colony optimisation metaheuristic”, European Journal of Operational Research, vol 143, pp 218–229, 2002.

[54] N. Hall, W. Kubiak, S. Sethi., “Earliness-tardiness scheduling problem II: deviation of completion times about a restrictive common due date”, Operation, Research, vol 39, pp 847–856, 1991.

[55] R. Haupt., “A survey of priority rule-based scheduling”, OR Spektrum, vol 11, pp 3-16, 1989.

[56] J. Heinonen, F. Pettersson., “Hybrid ant colony optimisation and visibility studies applied to a job-shop scheduling problem”, Applied Mathematics and Computation, vol 187, pp 989–998, 2007.

[57] A. Hertz., “Tabu search for large scale timetabling problems”, European Journal of Operational Research, vol 54, pp 39–47, 1991.

[58] J.H. Holland., “Adaption in natural and artificial Systems”, The University of Michigan Press, Ann Arbor, 1975.

[59] X. Hu, Y. Shi, R.C. Eberhart., “Recent advances in particle swarm”, Proceedings of the IEEE congress on evolutionary computation, vol 1, Oregon, Portland, pp 90–97, 2004.

[60] Y. Ikura, M. Gimple., “Efficient scheduling algorithms for a single batch processing machine”, Operations Reseach. Letters, vol 5, no 2, pp 61–65, 1986.

[61] A. S. Jain, S. Meeran., “Deterministic job-shop scheduling: Past, present and future”, European Journal of Operational Research, vol 113, pp 390-434, 1999.

[62] J. Jerald, P. Asokan, G. Prabaharan, R. Saravanan., “Scheduling optimisation of flexible manufacturing systems using particle swarm optimisation algorithm”, The International Journal of Advanced Manufacturing Technology, vol 25, pp 964–971, 2005.

[63] E.L. Johnson., “Integer programming, Society for industrial and applied mathematics”, USA, 1980.

[64] S.M. Johnson., “Optimal two- and three-stage production schedules with setup times included”, Naval Research Logistics Quarterly, vol 1, pp 61–68, 1954.

[65] F. Jolai., “Minimizing number of tardy jobs on a batch processing machine with incompatible job Families”, European Journal of Operational Research, vol 162, pp 184–190, 2005.

[66] K.D. Jong., “An analysis of the behaviour of a class of genetic adaptive system”, the University of Michigan Press, Ann Arbor, 1975.

[67] A.R. Kan., “Machine Scheduling problems classification”, complexity and computations, Martinus Nijhoff, The Hague, 1976.

[68] J. Kennedy, R.C. Eberhart., “Particle swarm optimisation”, In Proceedings of the IEEE international conference on neural networks, New Jersey IEEE Press, pp 1942–1948, 1995.

[69] J. Kennedy, R.C. Eberhart., “A discrete binary version of the particle swarm algorithm”, Proceedings of the world multi conference on systemic, cybernetics and informatics, NJ Piscatawary , pp 4104–4109, 1997.

[70] S. Khuri, S.R. Miryala., “Genetic Algorithms for solving open shop scheduling problems”, Lecture Notes in Computer Science, vol 1695, pp 357-368, 1999.

[71] R. Kolisch, A. Sprecher., “PSPLIB – A project scheduling problem library”, European Journal of Operational Research, vol 96, no 1, pp 205–216, 1997.

[72] S.V. Kolliopoulos, G. Steiner., “Approximation algorithms for minimizing the total weighted tardiness on a single machine”, Theoretical Computer Science, vol 355, pp 261–273, 2006.

[73] C. Koulamas., “Theory and methodology single-machine scheduling with time windows and earliness/tardiness penalties”, European Journal of Operational Research, vol 91, pp 190-202, 1996.

[74] W. Kubiak, C. Sriskandarajah, K. Zaras., “A note on the complexity of open shop scheduling problems”, Information Systems and Operational Research, vol 29 , pp 284–294, 1991.

[75] A.H. Land, A.G. Doig., “An automatic method of solving discrete programming problems”, Econometricca, vol 28, no 3, pp 497–520, 1960.

[76] E.L. Lawler., “A pseudopolynomial algorithm for sequencing jobs to minimise total tardiness”, Annals of Discrete Mathematics, vol 1, pp 331–342, 1977.

[77] J.K. Lenstra, A.H.G. Rinnooy Kan, P. Brucker., “Complexity of machine scheduling problems”, Annals of Discrete Mathematics, vol 1, pp 343–362, 1977.

[78] C. Liaoa, C. Tsengb, P. Luarnb., “A discrete version of particle swarm optimisation for flow shop scheduling problems”, Computers & Operations Research ,vol 34, pp 3099 – 3111, 2007.

[79] C. Liaw., “A hybrid genetic algorithm for the open shop scheduling problem”, European Journal of Operational Research, vol 124, pp 28-42, 2000.

[80] J. Lis, A. Eiben., “A Multi-sexual genetic algorithm for multi-objective optimization”, In Proceedings of IEEE International Conference on Evolutionary Computaiton, Nagoyo, Japan, pp 59–64, IEEE, Los Alamitos, 1996.

[81] C.Y. Liu, R.L. Bulfin., “Scheduling ordered open shops”, Computer Operation Research, vol 14, pp 257–264, 1987.

[82] J. Liu and L. Tang., “A Modified Genetic Algorithm for Single Machine Scheduling”, Computers & Industrial Engineering, vol 37, pp 43-46, 1999.

[83] T. Liu, J. Tsai, J. Chou., “Improved genetic algorithm for the job-shop scheduling problem”, International Journal of Advance Manufacturing Technology, vol 27, pp 1021–1029, 2006.

[84] H.R.D. Lourenco, M. Zwijnenburg., “Combining the large-step optimization with tabu-search application to the job-shop scheduling problem”, Meta-heuristics: Theory and Applications, Kluwer Academic Publishers, Boston, MA, pp 219-236, 1996.

[85] C. Lowa, Y. Yeha., “Genetic algorithm-based heuristics for an open shop scheduling problem with setup processing and removal times separated”, Robotics and Computer-Integrated Manufacturing, vol 25, pp 314– 322, 2009.

[86] M. Mathirajan, V. Chandru, A. Sivakumar., “Heuristic algorithms for scheduling heat treatment furnaces of steel casting industries”, Sadahana, vol 32, part 5, pp 479–500, 2007.

[87] D. Merkle, M. Middendorf, H. Schmeck., “Ant colony optimisation for resource-constrained project scheduling”, J. Weglarz, Ed. Amsterdam, project scheduling problems, in Handbook on Recent Advances in Project Scheduling, The Netherlands, Kluwer, pp 147–178, 1999.

[88] P. Merz, B. Freisleben., “A genetic local search approach to the quadratic assignment problem”, In: C.T. Back (editor), Proceedings of the 7th international conference on genetic algorithms. San Diego, CA: Morgan Kaufmann, pp 465-472, 1997.

[89] D. Miller, H. Chen, J. Matson, Q. Liu., “A hybrid genetic algorithm for the single machine scheduling problem”, Journal of Heuristics, vol 5, pp 437–454, 1999.

[90] L. Monch, H. Balasubramanian, J. Fowler, M. Pfund., “Heuristic scheduling of jobs on parallel batch machines with incompatible job families and unequal ready times”, Computers and Operations Research, vol 32, pp 2731–2750, 2005.

[91] J.F. Muth, G.L. Thompson., “Industrial scheduling”, Prentice Hall, Englewood Cliffs (1963).

[92] S. Naka, T. Genji, T. Yura, Y. Fukuyama., “A hybrid particle swarm optimisation for distribution state estimation”, IEEE Transactions on Power Systems, Chuanwen, E. Bompard , Energy Conversion and Management, vol 46 , pp 2689–2696, 2005.

[93] R. Nakano, T. Yamada., “Conventional genetic algorithm for job shop scheduling”, Proceedings of the Fourth International Conference on Genetic Algorithms and Their Application, 1991.

[94] H. Nazif, L.S. Lee., “A genetic algorithm on single machine scheduling problem to minimise total weighted completion time”, European Journal of Scientific Research, vol 35, pp 444-452, 2009.

[95] H. Oliver and R. Chandrasekharan., “Efficient dispatching rules for scheduling in a job shop”, International Journal of Production Economics, vol 48, pp 87-105, 1997.

[96] Q. Pana, M.F. Tasgetirenc, Y. Liangd., “A discrete particle swarm optimisation algorithm for the no-wait flowshop scheduling problem”, Computers & Operations Research, vol 35, pp 2807-2839, 2008.

[97] K.M. Passino., “Biomimicry of bacterial foraging for distributed optimization and control”, Control Systems Magazine, IEEE, vol 22, no 3, pp 52–67, 2002.

[98] M. Pinedo., “Scheduling theory, algorithms, and systems”, Prentice Hall, New York, 1995.

[99] S.G. Ponnambalam, H. Jagannathan, M. Kataria, A. Gadicherla., “A TSP-GA multi-objective algorithm for flow-shop scheduling”, The International Journal of Advanced Manufacturing Technology, vol 23, pp 909–915 (2004).

[100] C.N Potts, L.N. Van Wassenhove., “Single machine tardiness sequencing heuristics”, IIE Transactions, vol 23, pp 346–354 (1991).

[101] C. Prins., “An overview of scheduling problems arising in satellite communications”, Operation Research Society, vol 40, pp 611–623, 1994.

[102] R. Ramasesh., “Dynamic job shop scheduling a survey of simulation research”, Elsevier, Omega, vol 18, pp 43-57, 1990.

[103] R. Reeves., “Genetic algorithms and neighbourhood search, In Evolutionary Computing”, AISB Workshop, Leeds, U.K, pp 115-130 1994.

[104] C. Roberts, M. Dessouky, Y. Dessouky., “A virtual plant modeller for batch-chemical processes”, Intelligent Manufacturing Journal, vol 10, pp 211–223, 1999.

[105] J.S. Roger Jang., "ANFIS adaptive-network-based fuzzy inference systems", IEEE Transactions on Systems, Man, and Cybernetics, vol 23, no 3, pp 665-685, 1993.

[106] B. Roy, B. Sussmann., "Les probl'emes d'ordonnancement avec contraintes disjunctives", Note D.S, no 9, SEMA, Paris, France, 1964.

[107] R. Saravanan., "Manufacturing optimisation through intelligent techniques", by Taylor & Francis Group, LLC, 2006.

[108] S. Hartmnn., "Self-adapting genetic algorithm with an application to project scheduling", University of Kiel, Kiel, Germany, Tech. Rep. 506, (1999).

[109] D.Y. Sha , H. Lin., "A multi-objective PSO for job-shop scheduling problems", Expert Systems with Applications, vol 37, pp 1065–1070, 2010.

[110] M. Shaw, A. Whinston., "An artificial intelligence approach to the scheduling of flexible manufacturing systems", IIE Transactions, vol 21, pp 170–182, 1989.

[111] E. Taillard., "Benchmarks for basic scheduling problems", European Journal of Operational Research, vol 64, pp 278–285, 1993.

[112] E. Taillard., "Parallel taboo search techniques for the job shop scheduling problem", ORSA Journal on Computing, vol 16, no 2, pp 108-117, 1994.

[113] L. Tang, G. Liu., "A mathematical programming model and solution for scheduling production orders in Shanghai Baoshan Iron & Steel Complex", European Journal of Operational Research, vol 182, pp 1453–1468, 2007.

[114] M.F. Tasgetiren, Y.C. Liang, M. Sevkli, G. Gencyilmaz., "Particle swarm optimisation algorithm for makespan and maximum lateness minimization in permutation flowshop sequencing problem", Proceedings of the fourth international symposium on intelligent manufacturing systems, Turkey, Sakarya, pp 431–441, 2004.

[115] G. Teofilo, S. Sahni. Open Shop Scheduling to Minimize Finish Time. Journal of the Association for Computing Machinery, vol 23, no 4, pp 65-679 (1976).

[116] The Engineering Tool Box. http://www.engineeringtoolbox.com. Accessed on 22-11-2007.

[117] P. Tian, Z. Yang, S. Zhang., "Flowshop scheduling by simulated annealing", Journal Information and Conrro1, vol. 23, no 3, pp 133-139, 1994.

[118] V. T'kindt, J. Billaut., "Multicriteria Scheduling", Spring-Verlag, Berlin Heidelberg, (2006).

[119] G. Totten., "Steel heat treatment handbook", Second Edition, Taylor & Francis Group, LLC, 2006.

[120] I.C. Trelea., "The particle swarm optimisation algorithm convergence analysis and parameter selection", Information Processing Letter, vol 85, pp 317–325, 2003.

[121] Y. Tsujimura, Y. Mafune, M. Gen., "Effects of symbiotic evolution in genetic algorithms for job-shop scheduling", In Proceedings of the IEEE 34th International Conference on System Sciences , Hawaii, 2001.

[122] H.M. Voigt, J. Born, I. Santibanez-Koref., "Modelling and simulation of distributed evolutionary search processes for function optimization", In Parallel Problem Solving from Nature, vol 496, pp. 373-380, 1991.

[123] L. Wang, D.Z. Zheng., "An effective hybrid optimization strategy for job-shop scheduling problems". Computers and Operations Research, vol 28, pp 585-596, 2001.

[124] D. Weld., "An introduction to least-commitment planning", Artificial Intelligent Magazine, Winter, 1994.

[125] D. de Werrat, J. Erschler., "Open shop scheduling with some additional constraints", Graphs and Combinatorics, vol 12, pp 81-93, 1996.

[126] A.H. Wright., "Genetic algorithm for real parameter optimisation", In Rawlins, G.J.E. (Ed.), Foundations of Genetic Algorithms, Morgan-Kaufmann. pp 205-218, 1991.

[127] T. Yamada, R. Nakano., "A fusion of crossover and local search", In IEEE International Conference on Industrial Technology (ICIT'96), Shanghai, China, pp 426-430, 1996.

[128] K. Ying, C. Lia., "An ant colony system approach for scheduling problems", Production Planning & Control, vol 14, no 1, pp 68–75, 2003.

[129] R. McNaughton., “Scheduling with deadlines and loss functions”, Management Science, vol 6, pp 1–12 (1959).

[130] Y. Yin, J. Yu, Z. Cheng., “A genetic algorithm based approach to flowshop scheduling”, Proceedings of the 5th World Congress on Intelligent Control and Automation, Hangzhou, P.R. China, June 2004.

[131] N. Yoshitani, A. Hasegawa., “Model-based control of strip temperature for the heating furnace in continuous annealing”, IEEE Transactions on Control Systems Technology, vol 6, no 2, pp 146-156, 1998.

[132] Y. Yoshitomi., “A genetic algorithm approach to solving stochastic job-shop scheduling problems”, International Transaction in operational research, vol 9, pp 479-495, 2002.

[133] J. Yu, Y. Yin, Z. Chen., “Scheduling of an assembly line with a multi objective genetic algorithm”, The International Journal of Advanced Manufacturing Technology, vol 28, pp 551–555, 2006.

[134] M. Zdansky, J. Pozivil., “Combination genetic/tabu search algorithm for hybrid flowship optimization”, Proceedings of Algoritmy Conference on Scientific Computing, Vysoke Tatry –Podbanske, Sloakia, pp 230-236, September 2002.

[135] D. Zheng, S. Thomas, M. Kumaraswamy., “Applying pareto ranking and niche formation to genetic algorithm-based multi objective time–cost optimization”, Journal of Construction Engineering and Management, vol 131, no 1, pp 81-91, 2005.

Appendix A

Rule Bases

Appendix A shows the rules of each NF model that have been developed in this work.

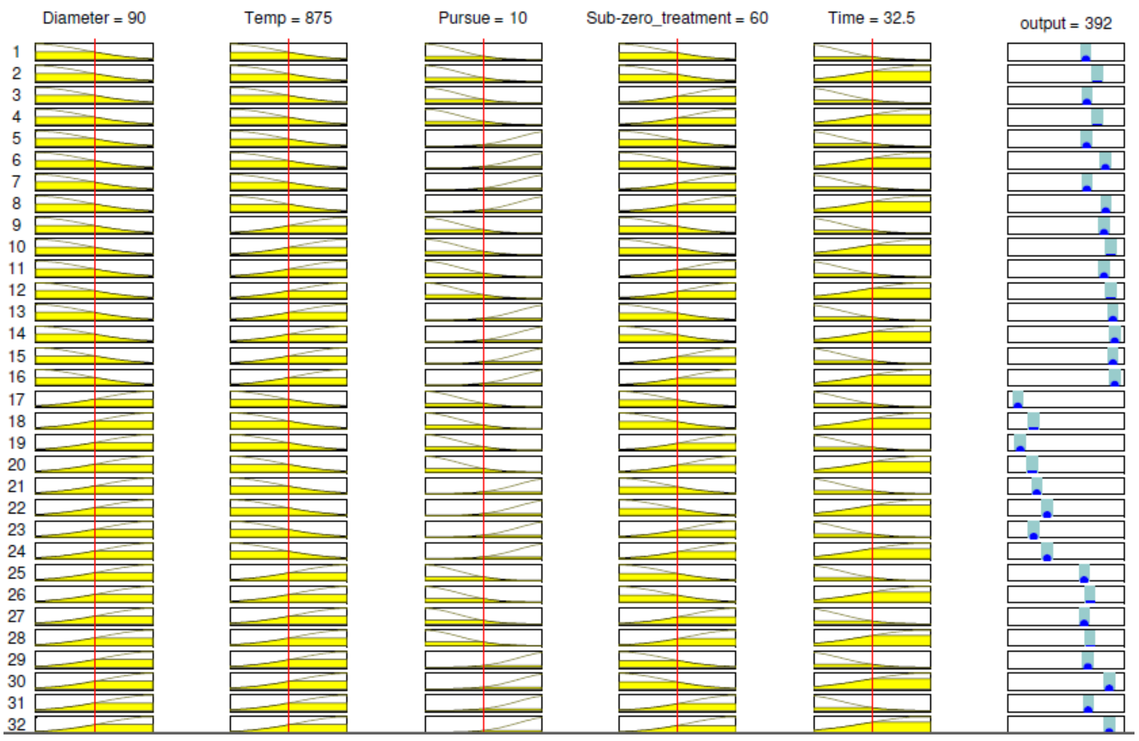


Figure A.1 NF model 2 rules for predicting steel's hardness.

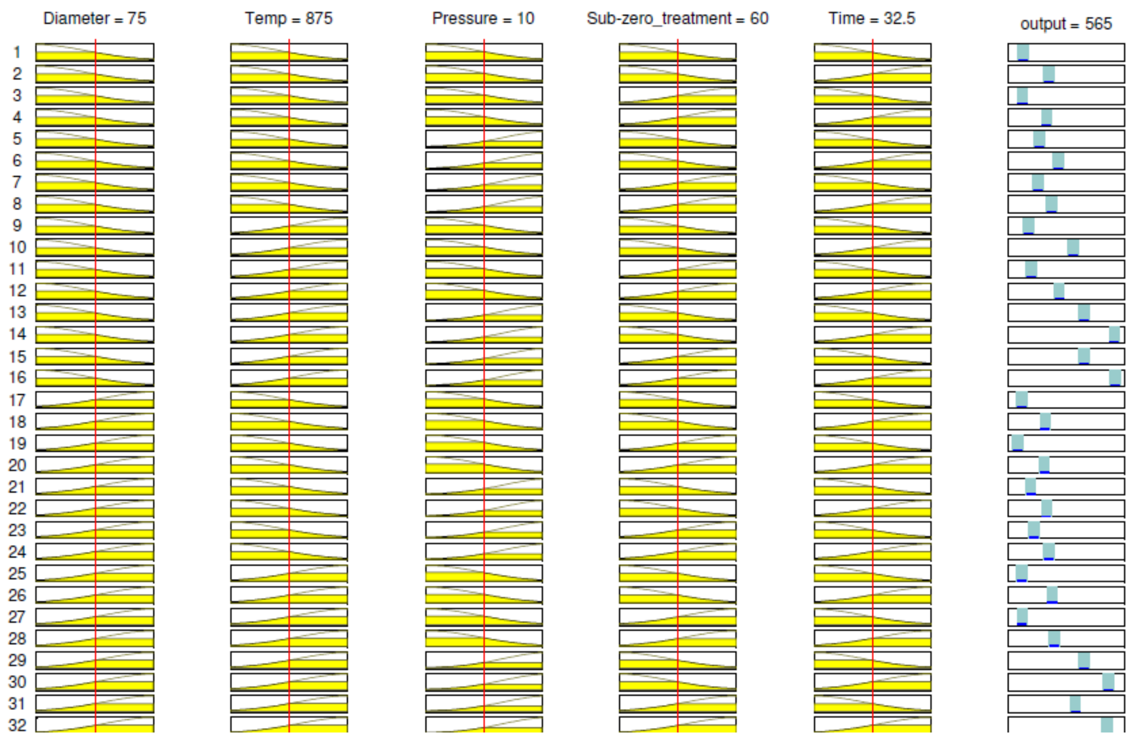


Figure A.2 NF model 3 rules for predicting steel's hardness.

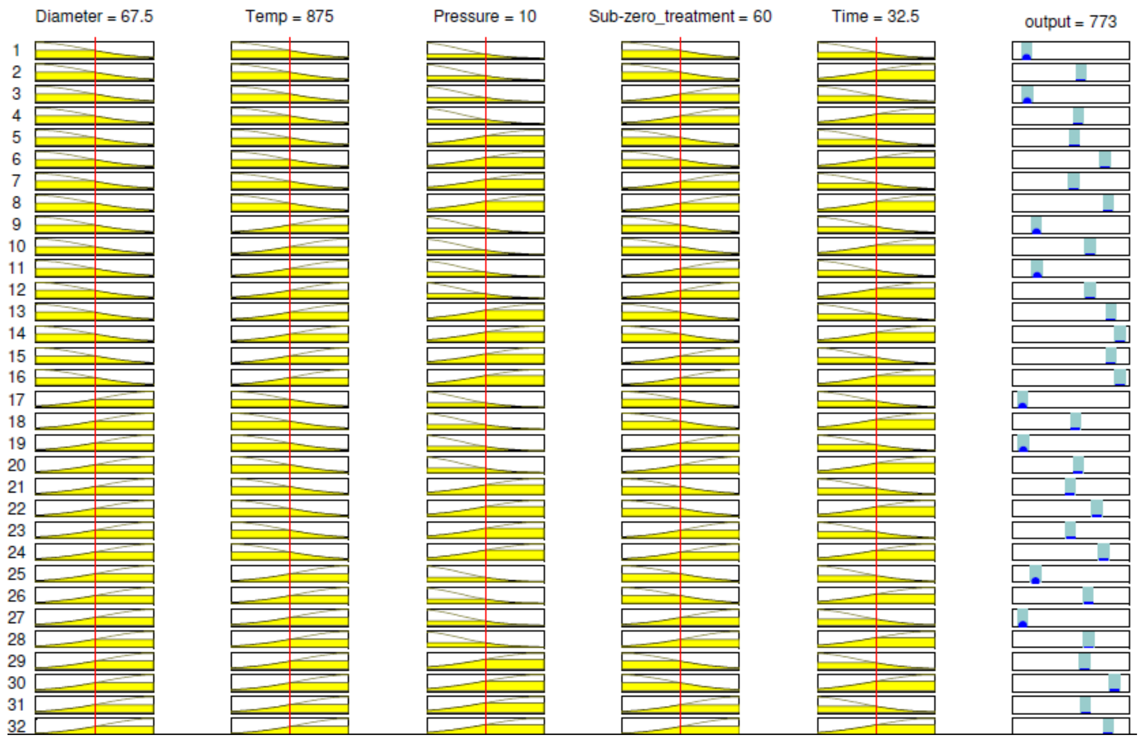


Figure A.3 NF model 4 rules for predicting steel's hardness.

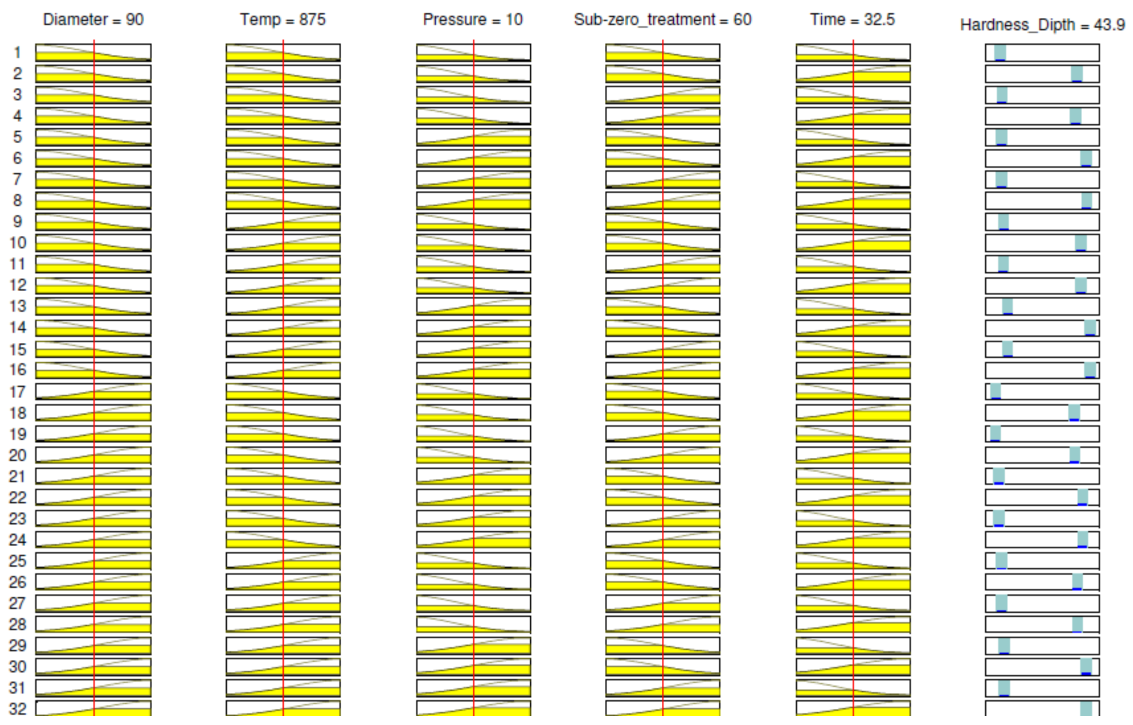


Figure A.4 NF model 2 rules for predicting steel’s depth of hardness.

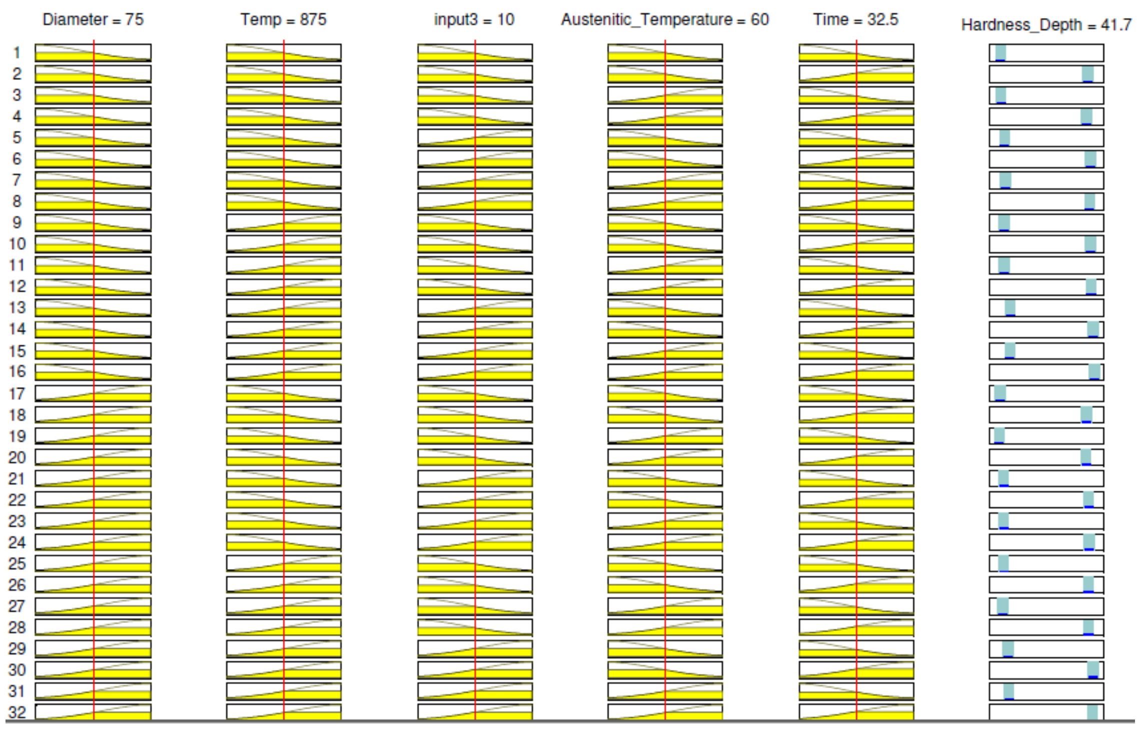


Figure A.5 NF model 3 rules for predicting steel’s depth of hardness.

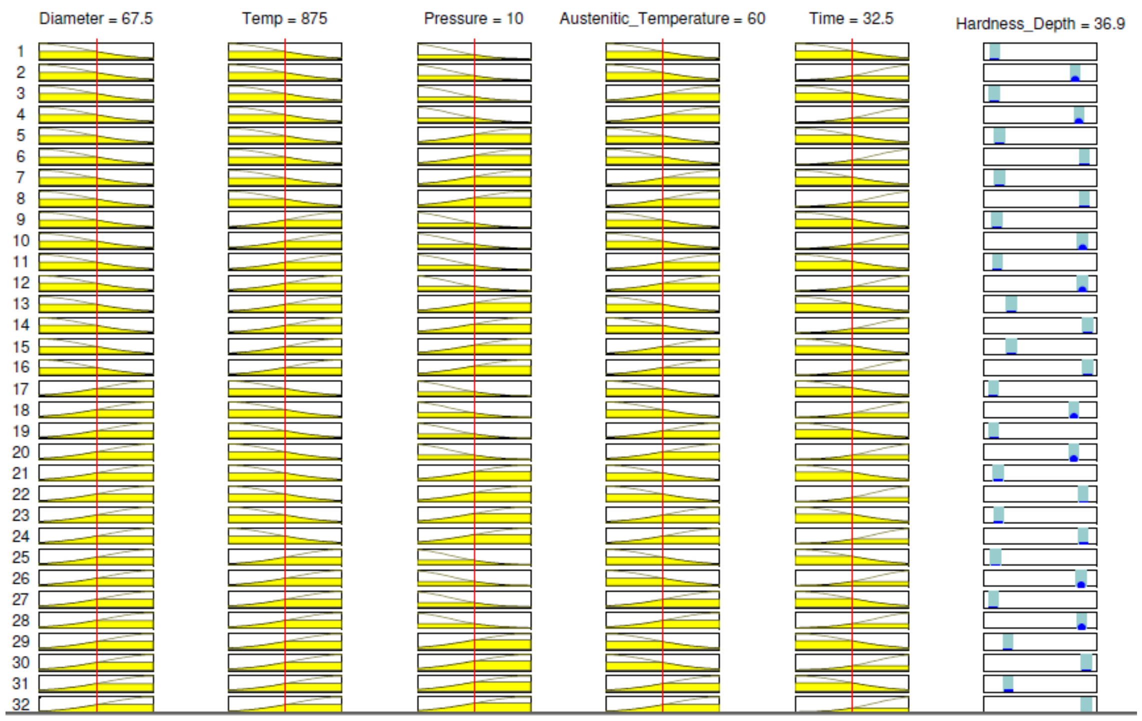


Figure A.6 NF model 4 rules for predicting steel's depth of hardness.

Appendix B

Experimental Data

Appendix B shows the experimental data that have been used to develop the NF models. Several models have been developed. The experimental data for different types of steel are given in Tables B.1 to B.4

Table B.1 Experimental data for steel (16Mn Cr5).

C wt %	Si wt%	Mn wt%	Cr wt%	Diameter (mm)	Austenitic Temperature (°C)	Pressure (Pa)	Sub-zero treatment (°C)	Time (min)	Hardness (HV)	Depth of Hardness (mm)
16	20	115	95	60	850	8	70	20	393	24.5
16	20	115	95	60	850	8	70	30	397	37.5
16	20	115	95	60	850	10	70	30	400	39
16	20	115	95	60	850	10	70	45	407	56
16	20	115	95	60	850	12	70	45	411	57.5
16	20	115	95	60	900	8	50	20	411	26
16	20	115	95	60	900	8	50	30	415	39
16	20	115	95	60	900	10	50	30	417	40.5
16	20	115	95	60	900	10	50	45	419	57.5
16	20	115	95	60	900	12	50	45	423	59
16	20	115	95	120	850	8	70	20	334	23
16	20	115	95	120	850	8	70	30	340	36
16	20	115	95	120	850	10	70	30	345	37.5
16	20	115	95	120	850	10	70	45	350	55
16	20	115	95	120	850	12	70	45	360	56
16	20	115	95	120	900	8	50	20	390	25
16	20	115	95	120	900	8	50	30	394	38
16	20	115	95	120	900	10	50	30	399	39
16	20	115	95	120	900	10	50	45	402	56
16	20	115	95	120	900	12	50	45	415	57.5
16	20	115	95	60	850	8	50	20	393	24.5
16	20	115	95	60	850	8	50	30	397	37.5
16	20	115	95	60	850	10	50	30	400	39
16	20	115	95	60	850	10	50	45	407	56
16	20	115	95	60	850	12	50	45	411	57.5
16	20	115	95	60	900	8	70	20	411	26
16	20	115	95	60	900	8	70	30	415	39
16	20	115	95	60	900	10	70	30	417	40.5
16	20	115	95	60	900	10	70	45	419	57.5
16	20	115	95	60	900	12	70	45	423	59
16	20	115	95	120	850	8	50	20	334	23
16	20	115	95	120	850	8	50	30	340	36
16	20	115	95	120	850	10	50	30	345	37.5
16	20	115	95	120	850	10	50	45	350	55
16	20	115	95	120	850	12	50	45	360	56
16	20	115	95	120	900	8	70	20	390	25
16	20	115	95	120	900	8	70	30	394	38
16	20	115	95	120	900	10	70	30	399	39
16	20	115	95	120	900	10	70	45	402	56
16	20	115	95	120	900	12	70	45	415	57.5
16	20	115	95	120	900	12	50	20	397	26.5
16	20	115	95	120	900	12	70	20	397	26.5
16	20	115	95	60	850	12	50	20	395	25.5
16	20	115	95	60	850	12	70	20	395	25.5
16	20	115	95	120	850	8	50	20	330	23
16	20	115	95	60	850	8	70	20	394	25.5

Table B.2 Experimental data for steel (14 Ni Cr 14).

C wt %	Si wt%	Mn wt%	Cr wt%	Mo wt%	Ni wt%	Diameter (mm)	Austenitic Temperature (°C)	Pres sure (Pa)	Sub-zero treatment (°C)	Time (min)	Hardn ess (HV)	Depth of Hardness (mm)
30	20	45	200	40	200	90	850	8	70	20	542	23
30	20	45	200	40	200	90	850	8	70	30	545	35
30	20	45	200	40	200	90	850	10	70	30	552	36
30	20	45	200	40	200	60	850	10	70	45	560	54
30	20	45	200	40	200	90	850	12	70	45	565	55.5
30	20	45	200	40	200	60	900	8	50	20	552	24.5
30	20	45	200	40	200	60	900	8	50	30	558	36.5
30	20	45	200	40	200	90	900	10	50	30	575	38
30	20	45	200	40	200	60	900	10	50	45	593	56
30	20	45	200	40	200	60	900	12	70	45	610	57.5
30	20	45	200	40	200	60	900	10	50	20	564	26
30	20	45	200	40	200	90	900	10	50	20	560	25.5
30	20	45	200	40	200	60	850	10	50	20	550	24
30	20	45	200	40	200	90	850	10	50	20	548	24
30	20	45	200	40	200	90	850	8	50	20	542	23
30	20	45	200	40	200	90	900	8	50	20	543	24.25
30	20	45	200	40	200	60	850	8	50	20	544	23.5
30	20	45	200	40	200	60	900	12	50	20	590	26.5
30	20	45	200	40	200	60	900	12	70	20	590	26.5
30	20	45	200	40	200	90	900	12	50	20	586	26
30	20	45	200	40	200	90	900	12	70	20	586	26
30	20	45	200	40	200	60	850	12	50	20	556	25
30	20	45	200	40	200	90	850	12	70	20	553	24.5
30	20	45	200	40	200	60	850	12	70	20	556	25
30	20	45	200	40	200	90	850	12	50	20	553	24.5
30	20	45	200	40	200	90	850	12	50	30	554	36.5
30	20	45	200	40	200	60	850	12	50	30	564	37
30	20	45	200	40	200	60	900	12	50	30	594	39
30	20	45	200	40	200	90	900	12	50	45	605	57.5
30	20	45	200	40	200	60	900	12	70	30	594	39
30	20	45	200	40	200	90	900	12	70	30	590	38.5
30	20	45	200	40	200	90	900	10	50	45	580	56
30	20	45	200	40	200	60	900	8	70	20	552	25
30	20	45	200	40	200	60	900	8	50	20	552	25
30	20	45	200	40	200	60	850	8	70	20	544	23.5
30	20	45	200	40	200	90	900	8	70	20	543	24.25
30	20	45	200	40	200	60	900	8	70	30	558	36.5
30	20	45	200	40	200	90	900	8	70	30	555	36
30	20	45	200	40	200	60	850	8	70	30	549	35.5
30	20	45	200	40	200	60	900	10	70	20	564	26
30	20	45	200	40	200	60	850	10	70	20	550	24.5
30	20	45	200	40	200	90	900	10	70	20	560	25.5
30	20	45	200	40	200	60	850	12	70	45	569	56.5
30	20	45	200	40	200	60	850	10	50	45	565	55.5
30	20	45	200	40	200	90	850	8	50	45	560	54.5
30	20	45	200	40	200	90	900	8	70	45	566	55.5
30	20	45	200	40	200	90	850	12	50	45	565	55.5
30	20	45	200	40	200	90	850	8	70	45	560	54.5
30	20	45	200	40	200	60	850	8	70	45	564	55
30	20	45	200	40	200	60	850	8	50	45	564	55
30	20	45	200	40	200	60	850	8	70	45	564	55
30	20	45	200	40	200	90	900	8	50	45	566	55.5
30	20	45	200	40	200	60	900	8	50	45	570	56.5
30	20	45	200	40	200	60	900	8	70	45	570	56.5
30	20	45	200	40	200	60	900	12	70	45	610	57.5
30	20	45	200	40	200	90	900	12	50	45	605	56.75
30	20	45	200	40	200	90	900	12	70	45	605	56.75

Table B.3 Experimental data for steel (30 Cr Ni Mo 8).

C wt %	Si wt%	Mn wt%	Cr wt%	Ni wt%	Diameter (mm)	Austenitic Temperature (°C)	Press ure (Pa)	Sub-zero treatment (°C)	Time (min)	Hard ness (HV)	Depth of Hardness (mm)
14	25	55	75	325	100	850	8	70	20	407	23
14	25	55	75	325	100	850	8	70	30	409	35
14	25	55	75	325	100	850	10	70	30	411	36
14	25	55	75	325	100	850	10	70	45	413	54
14	25	55	75	325	100	850	12	70	45	416	56
14	25	55	75	325	100	900	8	50	20	415	24.4
14	25	55	75	325	100	900	8	50	30	425	37
14	25	55	75	325	100	900	10	50	30	428	38
14	25	55	75	325	100	900	10	50	45	435	58
14	25	55	75	325	100	900	12	50	45	440	60
14	25	55	75	325	80	850	8	50	20	411	24
14	25	55	75	325	80	850	8	50	30	414	36
14	25	55	75	325	80	850	10	50	30	417	37
14	25	55	75	325	80	850	10	50	45	420	55
14	25	55	75	325	80	850	12	50	45	422	57
14	25	55	75	325	80	900	8	70	20	416	24.5
14	25	55	75	325	80	900	8	70	30	426	38
14	25	55	75	325	80	900	10	70	30	429	39
14	25	55	75	325	80	900	10	70	45	436	59
14	25	55	75	325	80	900	12	70	45	441	61
14	25	55	75	325	100	850	8	50	20	407	23
14	25	55	75	325	100	850	8	50	30	409	35
14	25	55	75	325	100	850	10	50	30	411	36
14	25	55	75	325	100	850	10	50	45	413	54
14	25	55	75	325	100	850	12	50	45	416	56
14	25	55	75	325	100	900	8	70	20	415	24.4
14	25	55	75	325	100	900	8	70	30	425	37
14	25	55	75	325	100	900	10	70	30	428	38
14	25	55	75	325	100	900	10	70	45	435	58
14	25	55	75	325	100	900	12	70	45	440	60
14	25	55	75	325	80	850	8	70	20	411	24
14	25	55	75	325	80	850	8	70	30	414	36
14	25	55	75	325	80	850	10	70	30	417	37
14	25	55	75	325	80	850	10	70	45	420	55
14	25	55	75	325	80	850	12	70	45	422	57
14	25	55	75	325	80	900	8	50	20	416	24.5
14	25	55	75	325	80	900	8	50	30	426	38
14	25	55	75	325	80	900	10	50	30	429	39
14	25	55	75	325	80	900	10	50	45	436	59
14	25	55	75	325	80	900	12	50	45	441	61
14	25	55	75	325	80	900	8	70	45	421	54
14	25	55	75	325	80	900	8	50	45	421	54
14	25	55	75	325	100	900	8	50	45	420	54
14	25	55	75	325	80	900	12	50	20	422	25
14	25	55	75	325	80	900	12	70	20	422	25
14	25	55	75	325	100	900	12	50	20	421	25
14	25	55	75	325	100	900	12	70	20	421	25
14	25	55	75	325	100	850	12	70	20	414	24
14	25	55	75	325	100	900	8	70	45	420	54
14	25	55	75	325	100	850	8	50	45	410	53
14	25	55	75	325	80	850	12	70	20	415	24.5
14	25	55	75	325	80	850	12	50	20	415	24.5
14	25	55	75	325	100	850	8	70	45	410	53
14	25	55	75	325	80	850	8	70	45	410	53.5
14	25	55	75	325	100	850	8	50	45	410	53
14	25	55	75	325	80	850	8	50	45	410	53.5
14	25	55	75	325	100	850	10	50	20	409	23.5
14	25	55	75	325	100	850	10	70	20	409	23.5
14	25	55	75	325	100	900	10	50	20	421	25
14	25	55	75	325	100	900	10	70	20	421	25
14	25	55	75	325	80	900	10	50	20	423	25.5
14	25	55	75	325	80	900	10	70	20	423	25.5

Table B.4 Experimental data for steel (20 Mn 45).

C wt %	Si wt%	Mn wt%	Diameter (mm)	Austenitic Temperature (°C)	Pressure (Pa)	Sub-zero treatment (°C)	Time (min)	Hardness (HV)	Depth of Hardness (mm)
20	25	45	75	850	8	70	20	735	23
20	25	45	60	900	8	70	30	755	30.5
20	25	45	75	850	10	70	30	768	31
20	25	45	75	900	10	70	45	778	57
20	25	45	60	850	12	70	45	786	57
20	25	45	75	900	8	50	20	745	24.5
20	25	45	60	850	8	50	30	752	29.5
20	25	45	75	900	10	50	30	774	33
20	25	45	75	850	10	50	45	771	55.5
20	25	45	60	850	12	50	45	784	57
20	25	45	60	850	12	50	20	766	25
20	25	45	60	850	12	70	20	766	25
20	25	45	75	850	12	50	20	762	24.5
20	25	45	75	850	12	70	20	762	24.5
20	25	45	75	850	8	50	20	735	23
20	25	45	60	900	8	50	20	748	25
20	25	45	60	900	8	70	20	748	25
20	25	45	60	850	8	70	20	738	23.5
20	25	45	60	850	8	50	20	738	23.5
20	25	45	75	850	12	50	30	773	32
20	25	45	75	850	8	50	30	750	29
20	25	45	75	900	12	50	30	779	34.5
20	25	45	75	900	8	50	30	753	30
20	25	45	75	850	12	70	30	773	32
20	25	45	75	850	8	70	30	750	29
20	25	45	75	900	12	70	30	779	34.5
20	25	45	75	900	8	70	30	753	30
20	25	45	60	900	12	70	30	790	35.5
20	25	45	60	900	12	50	30	790	35.5
20	25	45	60	850	12	70	30	776	32.5
20	25	45	60	850	12	50	30	776	32.5
20	25	45	60	850	8	70	30	752	29.5
20	25	45	60	900	8	50	30	755	30.5
20	25	45	60	850	8	50	45	769	54
20	25	45	75	850	8	50	45	767	53.5
20	25	45	75	850	8	70	45	767	53.5
20	25	45	60	850	8	70	45	768	55
20	25	45	60	900	8	50	45	777	56.5
20	25	45	60	900	8	70	45	777	56.5
20	25	45	75	900	8	50	45	775	56
20	25	45	75	900	8	70	45	775	56
20	25	45	75	900	12	50	45	790	58
20	25	45	75	900	12	70	45	790	58
20	25	45	60	900	12	50	45	794	58.5
20	25	45	60	900	12	70	45	794	58.5
20	25	45	75	850	12	50	45	782	56.5
20	25	45	75	850	12	70	45	782	56.5

Appendix C

Single Benchmark Generation

Appendix C shows the Matlab code that has been used to generate single resource benchmarks problems [8].

```
function PG_for_DDP

clear all
global m;
global m1;
global b;
global seed;
global problem;

m=100000000;
m1=10000;
b=31415821;
seed=0;
start_seed=3794612;

n = input('PROBLEM SIZE: () ');
k = input('INSTANCE NUMBER: (1) ');
range_p = (input('RANGE PROCESSING TIME: () '));
range_a = (input('RANGE EARLINESS PENALTY: () '));
range_b = (input('RANGE LATENESS PENALTY: () '));
i =0;
seed =start_seed+n+k;

for i=1:n,
    problem(i,1) = Randomint(range_p);
    problem(i,2) = Randomint(range_a);
    problem(i,3) = Randomint(range_b);
end;
disp(problem);
A=problem;
function randnumber = Randomint(range)
global seed;
global m;
global m1;
global b;
seed =mod((Mult(seed,b)+1),m);
randnumber =floor((floor(seed/m1)*range)/m1)+1;

function mult = Mult(p,q)
global m;
global m1;
p1 =floor(p/m1);
p0=mod(p,m1);
q1 =floor(q/m1);
q0=mod(q,m1);
mult =mod((mod((p0*q1+p1*q0),m1)*m1+p0*q0),m);
```