H24/3562

MONASH UNIVERSITY
THESIS ACCEPTED IN SATISFACTION OF THE
REQUIREMENTS FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY
ON................ 6 April 2004...............

.................................[redacted]....................

Sec. Research Graduate School Committee

**4. Declaration by candidate**

Candidate's signature: [redacted] Date: 17/12/03

**5. Ratification by academic unit**

This is to ascertain that the Department / School / Centre / Institute has no objection to the candidate's options regarding access to the Library thesis copy. If so, please sign below and return the completed form to: **Monash Graduate School, Building 3D, Clayton Campus.**

Supervisor's signature: [redacted] Date: 17/12/03
(please print name) Prof. Chung-Hsing Yeh

MONASH University

AUSTRALIA

# Fuzzy Multi-Mode Resource-Constrained Project Scheduling

BY

Hongqi Pan

B. Eng, GradDip. BusSys, M. BusSys

School of Business Systems

Faculty of Information Technology

Monash University

December 2003

*Submitted in total fulfilment of the requirements of the degree of Doctor of Philosophy*

# Declaration

I, Hongqi Pan, hereby declare that this thesis contains no material that has been accepted for the award of any other degree or diploma in any university or equivalent institution. To my best knowledge and belief, this thesis contains no material previously published or written by other authors except where due reference is made in the text of the thesis.

Signed [redacted]

Date 17/12/03

School of Business Systems
Faculty of Information Technology
Monash University

2003

# Acknowledgements

I am deeply indebted to my supervisor, Professor Chung-Hsing Yeh for his advice and encouragement throughout the period of my PhD research, particularly in the final stage of my thesis writing. He has spent an enormous amount of time giving me constructive suggestions and valuable comments in thesis writing, and proofreading my draft patiently. I wish also to express my sincere thanks to Professor Robert J. Willis for his beneficial advice and encouragement while he supervised my PhD studies during the early stage while he was head of our school. Also I would like to acknowledge Professor Zhongkai Xiong who gave up much of his time to help me understand complex mathematics, which laid the foundation for me to develop effective approaches to fuzzy project scheduling while I was studying away at Shantou University in China.

I am also grateful to the School of Business Systems for providing research facilities, and particularly thanks to Mrs. Diane West, Mrs. Lesley Eamsophana, and Ms. Grace Bol who were so very kind to me and gave me a great help in so many ways throughout my years of studies. I also wish to express my sincere thanks to Professor Andrew Flitman, Professor Graeme Shanks, Professor Kate Smith, Dr. Vincent C S Lee, and Dr. John Betts for their solicitude during my final stage of PhD research. I also thank my fellow research students, including Dr. Rowena Chau, Mr. Yu-Liang Kuo, Mr. Pramesh Chand, Mr. Jisong Chen, Mr. Peter Phat Khai Huynh, Mr. Lin Yi Shao, Mr. Tze Hiang Alex Sim and Mr. Bob Li for their support and encouragement during my studies.

I would like specially to thank my good friend, Rev. Peter Grasby from St. Mary Magdalen's Parish in Jordanville for his time and patience in reading my final draft carefully, and for picking up spelling and small grammar mistakes which I missed during my writing in MS word.

I would like to dedicate this thesis to my parents who flew from China to care for our family, take on babysitting duties, and look after our family's daily life during the crucial stage of my thesis writing. Without their help and support, it would have been impossible for me to finish my PhD studies. I also warmly express my appreciation to my God mother, Xianmei Dai and my mother-in-law, Peijun Zhang.

In conclusion, my eternal thanks are due to my wife. Although her health has not always been good, she has supported me unfailingly, and I am so grateful for the sacrifices she has made, and for her good humour in the face of what must have appeared to be neglect on my part. To my daughter, Jiani Pan and my son, Louie Ziyang Pan, I must express my gratitude for their support, understanding, and sacrifice during the past few years when the scope of my research seemed to stretch far beyond what I had initially envisaged.

# Abstract

This thesis develops six optimisation approaches for efficiently and effectively solving fuzzy multi-mode resource-constrained project scheduling (FMMRCPS) of any realistic sizes with single or multiple objectives, where activities have fuzzy duration times and have several performance modes under different workable resource requirements. FMMRCPS is complex because it needs to solve two subproblems simultaneously: (a) mode assignment, and (b) the sequence of activities in a schedule.

A fuzzy hybrid goal programming approach is developed to deal with multiple objectives in FMMRCPS. The approach minimises both the project completion time and the project cost, as these two objectives are major concerns in project scheduling. This approach integrates both fuzzy set theory for handling fuzzy activity duration times and a rule knowledge base for mode assignment. Thus, the complexity of FMMRCPS is simplified to single mode project scheduling.

Five heuristic and metaheuristic approaches are developed to deal with the single objective of minimising the project completion time, including: (a) a fuzzy heuristic approach, (b) a fuzzy genetic algorithm (GA), (c) a fuzzy GA with tabu, (d) a fuzzy simulated annealing (SA) approach, and (e) a fuzzy SA approach with tabu. These approaches provide a methodological framework for solving complex FMMRCPS of practical sizes efficiently and effectively, whereas exact approaches are not viable.

The fuzzy heuristic approach applies a set of priority rules and mode assignment policies to fuzzy forward and backward scheduling simultaneously in order to minimise the fuzzy project completion time. Its application to a real instance of dredge overhaul scheduling demonstrates its simplicity and effectiveness in solving FMMRCPS involving fuzzy activity duration times.

Both fuzzy GA-based approaches use both fuzzy forward and backward scheduling to permit the survival of good solutions and the elimination of bad ones in generations, after operations on activity priorities and modes are manipulated. To perform fuzzy GA effectively, a chromosome, representing a schedule, is specifically designed which has the following advantages: (a) reflecting the fuzzy nature of FMMRCPS, (b) facilitating the application of genetic operations on both activity priorities and modes, (c) always generating feasible schedules, and (d) avoiding an additional procedure of decoding a chromosome that is commonly required in GA. In addition to the functions of fuzzy GA, fuzzy GA with tabu has an extra function for avoiding newly generated chromosomes from being the same as those previously generated. The performance evaluation study shows that fuzzy GA with tabu produces more diverse schedules than the fuzzy GA alone for finding a good approximate globally optimal schedule effectively.

Two versions of fuzzy SA approaches: (a) fuzzy SA and (b) fuzzy SA with tabu are developed for obtaining a good approximate globally optimal schedule through searching various neighbourhoods by perturbing both activity priorities and modes at different scheduled times from individual current schedules. A solution representation is carefully designed in order to generate neighbourhoods easily and efficiently through the perturbation. In addition, a novel technique, using the time pointer, is proposed for evading a full scheduling in neighbourhood generations, thus reducing the computational time enormously. The purpose of developing fuzzy SA with tabu is to avoid cycling the same search space that may occur in applying fuzzy SA alone. The integrated tabu mechanism keeps track of solution attributes of recently visited neighborhoods in order to forbid the search of neighborhoods already visited. Thus, the search explores more areas to identify better solutions. An experiment conducted suggests that fuzzy SA with tabu outperforms fuzzy SA alone in terms of minimising the fuzzy project completion time.

The major contributions of this research are in the development of a number of approaches that have their own distinctive merits in solving complex FMMSCPS of practical sizes, providing flexibility in effectively tackling project scheduling as encountered in practical applications. First, the research develops a novel technique to decompose a complex multi-mode scheduling problem into a simplistic single mode scheduling problem that can be solved effectively by simple ways such as fuzzy hybrid goal programming. Second, in the evolutionary based approaches, a problem-specific chromosome is effectively designed to gain insights into the nature of complex scheduling for the approaches to be applied efficiently in order to obtain a globally optimal solution. Third, the specifically designed solution representation increases the possibility of digging for the best solution. Fourth, the result of this research provides useful insights for combining different approaches into a system framework by utilising the advantages of each approach. It gives an efficient and effective way of procuring an approximate globally optimal solution for solving practically sized FMMRCPS under fuzzy activity duration times.

# Table of Contents

# List of Publications

## REFEREED JOURNAL PAPERS

Yeh, C.-H., H. Deng and H. Pan (1999). Multi-criteria analysis for dredger dispatching under uncertainty, *Journal of the Operational Research Society*, 50(1), 35-43.

Yeh, C.-H., R. J. Willis, H. Deng and H. Pan (1999). Task oriented weighting in multi-criteria analysis, *European Journal of Operational Research*, 119(1), 130-146.

Pan, H. and C.-H. Yeh (2003). Simulated annealing for multi-mode project scheduling, *WSEAS Transactions on Systems*, 3(2), 681-689.

Pan, H. and C.-H. Yeh. (2003) GA for Fuzzy multi-mode resource-constrained project scheduling, *WSEAS Transactions on Systems*, 4(2), 983-990.

Pan, H. and C.-H. Yeh Genetic algorithm for resource-constrained project scheduling, submitted to *IEEE Transactions on Engineering Management*.

Pan, H. and C.-H. Yeh Genetic algorithm incorporating the tabu mechanism for resource-constrained project scheduling, submitted to *Computers & Operations Research*.

Pan, H. and C.-H. Yeh. Hybrid genetic algorithm for fuzzy resource-constrained project scheduling, submitted to *Journal of the Operational Research Society*.

## REFEREED PAPERS IN EDITED BOOKS AND CONFERENCE PROCCEDINGS

Pan, H., C.-H. Yeh and R. J. Willis (1998). A Fuzzy goal programming model for purchasing dredgers under uncertainty, *Proceedings of the 3rd International Conference on Management*, Shanghai, China, R115, pp.1-12.

Yeh, C.-H., H. Pan and R. J Willis (1999). A heuristic approach to fuzzy resource-constrained project scheduling, In M. Mohammadian (Ed.), *Computational Intelligence for Modelling, Control and Automation, International Conference on Computational Intelligence for Modelling, Control and Automation* (CIMCA'99), Vienna, Austria, IOS Press, pp.423-428.

Pan, H., R. J. Willis and C.-H. Yeh (1999). Resource-constrained project scheduling under uncertain activity duration, In M. Mohammadian (Ed.), *Computational Intelligence for Modelling, Control and Automation, International Conference on*

*Computational Intelligence for Modelling, Control and Automation* (CIMCA'99), Vienna, Austria, IOS, pp.429-434.

Deng, H., C.-H. Yeh, R.J. Willis and H. Pan (1999). A knowledge-based approach for criteria weighting in multi-criteria analysis, In M. Mohammadian (Ed.), *Computational Intelligence for Modelling, Control and Automation, International Conference on Computational Intelligence for Modelling, Control and Automation* (CIMCA'99), Vienna, Austria, IOS, pp.435-440.

Pan, H., C.-H. Yeh, and R. J. Willis (2001). Computer-aided system to solve uncertainty in project management, *Proceedings of the 10th IEEE International Conference on Fuzzy Systems* (FUZZ-IEEE 2001), Melbourne, Vol. 3, pp.1376-1379.

Pan, H, C.-H. Yeh and R. J. Willis (2001). A hybrid genetic algorithm for project scheduling, In C.E. D'Attellis, V.V. Kluev and N.E. Mastorakis (Eds.), *Mathematics and Simulation with Biological, Economical and Musicoacoustical Applications*, pp. 40-44.

Pan, H, R. J. Willis and C.-H. Yeh (2001). Resource-constrained project scheduling with fuzziness, In N. Mastorakis (Ed.), *Advances in Fuzzy Systems and Evolutionary Computation*, World Scientific & Engineering Society Press, pp.173-179.

Pan, H., C.-H. Yeh and R. J. Willis (2002). Solving uncertainty in project scheduling, *Proceedings of the 19th Annual Conference of Management and International Association of Management (AoM/IAoM)*, Quebec, Canada, pp. 219-225.

Pan, H and C.-H. Yeh (2003). Fuzzy project scheduling, *Proceedings of the 2003 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE 2003)*, St. Louis, USA, pp.755-760.

Pan, H. and C.-H. Yeh (2003). A metaheuristic approach to fuzzy project scheduling, In V. Palade, R.J. Howlett and L.C. Jain (Eds.), *Knowledge-Based Intelligent Information and Engineering Systems, Lecture Notes in Computer Science (Artificial Intelligence)*, Vol. 2773, Springer-Verlag, Berlin, Heidelberg, pp. 1081-1087.

## TECHNICAL REPORTS

Pan, H., C.-H. Yeh, R. J. Willis and H. Deng (1997). Software development for fuzzy multiple criteria decision making models, *Business Systems Research 1997*, School of Business Systems, Monash University, pp.539-547.

Pan, H., C.-H. Yeh, R. J. Willis and H. Deng (1997). Task oriented weighting in multi-criteria analysis under uncertainty, *Business Systems Research 1997*, School of Business Systems, Monash University, pp.201-220.

Pan, H., R. J. Willis and C.-H. Yeh (1997). The further perspective in multiple criteria decision making, *Business Systems Research 1997*, School of Business Systems, Monash University, pp.479-489.

Yeh, C-H., H. Deng, R. J. Willis and H. Pan (1997). A multi-criteria decision making approach to dredger dispatching, *Business Systems Research 1997*, School of Business Systems, Monash University, pp.447-460.

Pan, H., C-H. Yeh, R. J. Willis and H. Deng (1997). A fuzzy rule-based approach to determination of criteria weights in multiple criteria decision making, *Business Systems Research 1997*, School of Business Systems, Monash University, pp.111-131.

Willis, R. J., H. Pan C.-H. Yeh (1998). Resource-constrained project scheduling under uncertain activity duration, *Business Systems Research 1998*, School of Business Systems, Monash University, pp.21-27.

Yeh, C.-H., H. Pan and R. J. Willis (1998). A heuristic approach to fuzzy resource-constrained project scheduling, *Business Systems Research 1998*, School of Business Systems, Monash University, pp.331-339.

Pan, H., C.-H. Yeh and R. J. Willis (1998). A fuzzy goal programming model for purchasing dredgers under uncertainty, *Business Systems Research 1998*, School of Business Systems, Monash University, pp.126-133.

Willis, R. J., H. Pan and C.-H. Yeh (1998). Resource-constrained project scheduling under uncertain activity duration, *Business Systems Research 1998*, School of Business Systems, Monash University, pp. 21-30.

# List of Tables

# List of Figures

# Chapter 1

# Introduction

## 1.1 Motivation of the Research

Resource-constrained project scheduling (RCPS) is a research field of scheduling activities under the constraints of both precedence relationships and limited resources in order to satisfy a specified single objective or multiple objectives. RCPS has been widely used in such areas as industry, engineering, and defence. However, current project scheduling problem models have difficulty in meeting many requirements in the real-world. This motivates me to develop novel approaches to a new realistic scheduling problem model that is capable of handling most possible situations occurring in project scheduling.

RCPS was introduced by Kelley (1963), Lambourn (1963) and Wiest (1964) in the 1960s. In this field, each activity can only have one performance option, referred to as an executive mode. This classical RCPS of considering one single mode, has been the focus of much research over the past four decades (Özdamar and Ulusoy 1995, Brucker *et al.* 1999, and Herroelen *et al.* 2001).

To meet practical needs, multi-mode resource-constrained project scheduling (MMRCPS) was brought to attention by Elmaghraby (1977) in the late 1970s. MMRCPS was more flexible and realistic, allowing several executive options under different resource settings for an activity, as compared with classical RCPS. Each mode of an activity has its unique duration time and its unique resource requirements. Researchers have attempted to develop many different exact approaches using enumeration schemes and bounding rules. Even though some of these exact approaches are claimed to be "a powerful approach to MMRCPS", they can only solve small sized problems of up to a maximum of 30 activities (Bouleimen and Lecocq 2003). This is because MMRCPS is a complex problem, and requires tackling two subproblems simultaneously in a schedule: mode assignment and activity sequence. It is NP-hard in the strong sense in terms of combinatorial optimisation. Thus, these approaches are often unable to solve MMRCPS problems of practical sizes.

In MMRCPS, project sizes and the number of modes for an activity can greatly influence the mathematical complexity of scheduling, especially when many constraints are imposed. Developing heuristic, particularly metaheuristic approaches seems to be a more promising means of solving MMRCPS with NP-hardness (Kolisch and Padman 2001). Boctor (1993) proposed a rule-based heuristic approach employing priority rules and a resource-feasible mode for solving MMRCPS. Drexl and Gruenewald (1993) applied regret-based biased random sampling with serial scheduling. López *et al.* (1996), Özdamar (1999), and Hartmann (2001) developed genetic algorithms (GA) for application to MMRCPS. Józefowska (2001) proposed a simulated annealing (SA) approach to MMRCPS. All the above approaches assume to be deterministic, and these approaches developed to MMRCPS are few in the literature.

In real situations, MMRCPS is often uncertain, and realistic approaches need to be developed to meet these situations. In this research, much attention has been paid to the development of novel and practical approaches for solving MMRCPS of any realistic sizes with single or multiple objectives under uncertainty.

## 1.2 Uncertainty in Project Scheduling

Project scheduling is often ridden with uncertainty. The uncertainty may arise from either a form of randomness or fuzziness. Randomness arises due to the uncertain future performance of an activity. The future performance can only be predicted using probability theory even if the information on past performances of the activity is sufficient. Fuzziness arises due to lack of precise information on the performance of the activity, and the performance has to be evaluated by vagueness and often subjectively. Such a phenomenon of the uncertainty can only be modelled by possibility theory, referred to as fuzzy set theory (Zadeh 1965).

Activity duration times under uncertainty in project scheduling were discussed by Britney (1976), who applied a probabilistic approach. That is, the activity duration time is estimated by beta distribution on the assumption of sufficient prior information. However, probabilistic approaches are theoretically valid only when the experimental or observational data about the activity performance are adequate. Such approaches have been widely applied in project scheduling for the last two decades.

In the real-world, project scheduling often enters the fuzzy environment because of lack of precise information on activity duration times or because of very limited availability of past information on similar types of activities. It is very difficult to forecast precise estimates of activity duration times due to their vagueness and imprecision. The application of fuzzy set theory in project scheduling provides an effective way of handling the uncertainty of this kind. Figure 1.1 indicates two major disciplines dealing with different kinds of uncertainty.

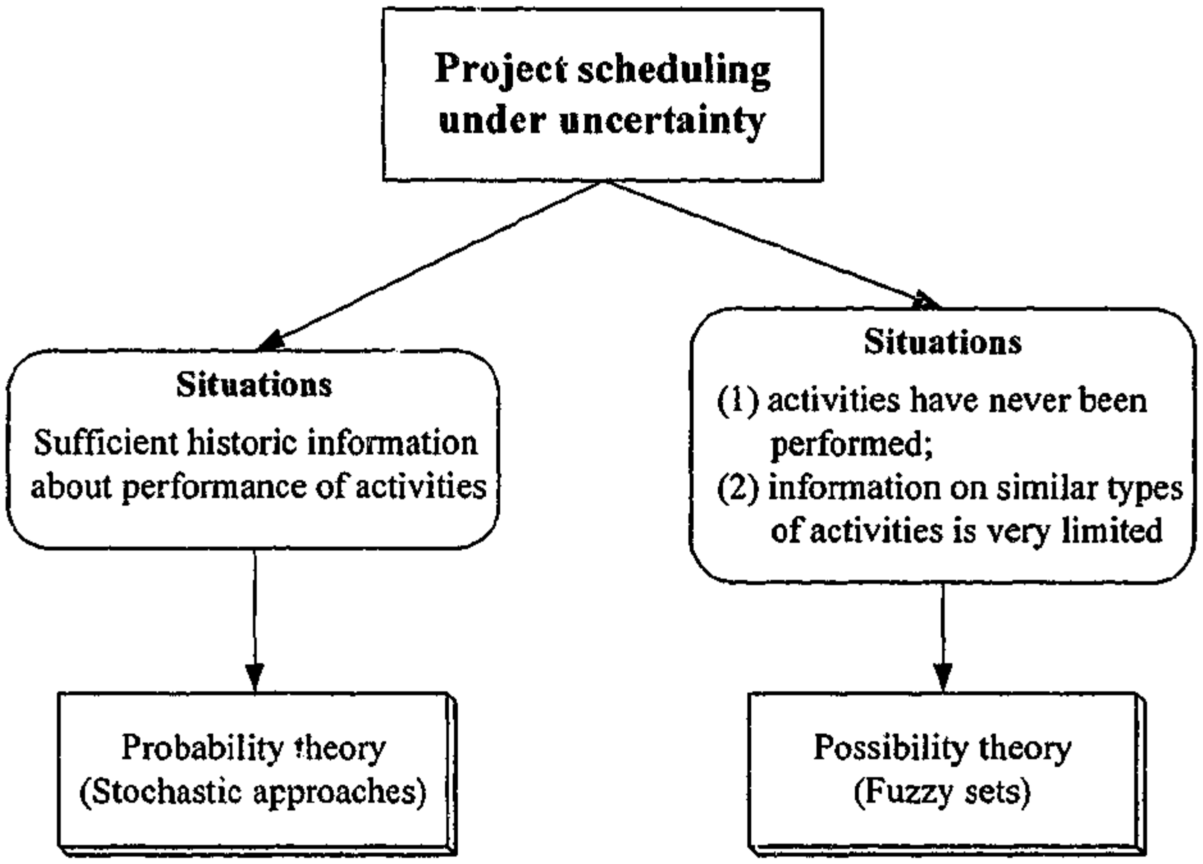

Project scheduling under uncertainty
Situations
Sufficient historic information about performance of activities
Situations
(1) activities have never been performed;
(2) information on similar types of activities is very limited
Probability theory (Stochastic approaches)
Possibility theory (Fuzzy sets)

Figure 1.1 Two types of uncertainty in project scheduling

To estimate activity duration times intuitively, linguistic terms represent an easy and effective way of expressing the decision maker (DM)'s assessment, using their knowledge and experience about limited and vague information. Linguistic terms such as "approximately", "impossible", "unlikely", "about", and "from ... to..." are often used to estimate fuzzy duration times of activities in project scheduling. For example, the duration time of an activity is approximately from 18 to 20 days and, impossible to finish within 15 days and unlikely to exceed 25 days. The duration time of another activity may be estimated to be completed in about 10 days, and unlikely to take less than 8 days or more than 15 days. These terms express vagueness and imprecision in nature, rather than clearly defined boundaries. Therefore, a fuzzy membership function is an effective way of expressing such imprecise information involving subjective judgement.

## 1.3 Research Problem Description

Project scheduling has permeated through almost every field in order to manage projects or large tasks. It is essential to define the problem domain that meets most possible practical requirements in project scheduling. In this research, a number of approaches are developed based on this scheduling problem model.

A number of project scheduling models have been proposed for tackling specific problems encountered in the practical world, since RCPS was introduced in the 1960s,. Weglarz (1979) introduced the preemptive RCPS model, in which some activities can be interrupted during processing. The time-constrained project scheduling model is concerned with completing a project in the given deadline (Deckro and Hebert 1989). The resource levelling model attempts to consume various resources as level as possible so that the steady usage rates may lead to the lower cost (Woodworth and Willie 1975). The net present value model is concerned with payment and cash flow from the financial point of view (Grinold 1972). The stochastic scheduling model involves probabilistic activity duration times according to estimates from prior available information (Devroye 1979). The multiple mode scheduling model makes flexible by allowing activities to have several executive manners (Elmaghraby 1977). These problem models presented above have aimed to meet specific needs in project scheduling.

The problem of interest in this research is to develop a number of practical approaches to solve realistic project scheduling problems. Therefore, the initial step is to define the research problem that responds to all possible situations occurring in real life project scheduling. Figure 1.2 shows the problem model representing the generalised case of project scheduling to be solved in this research.

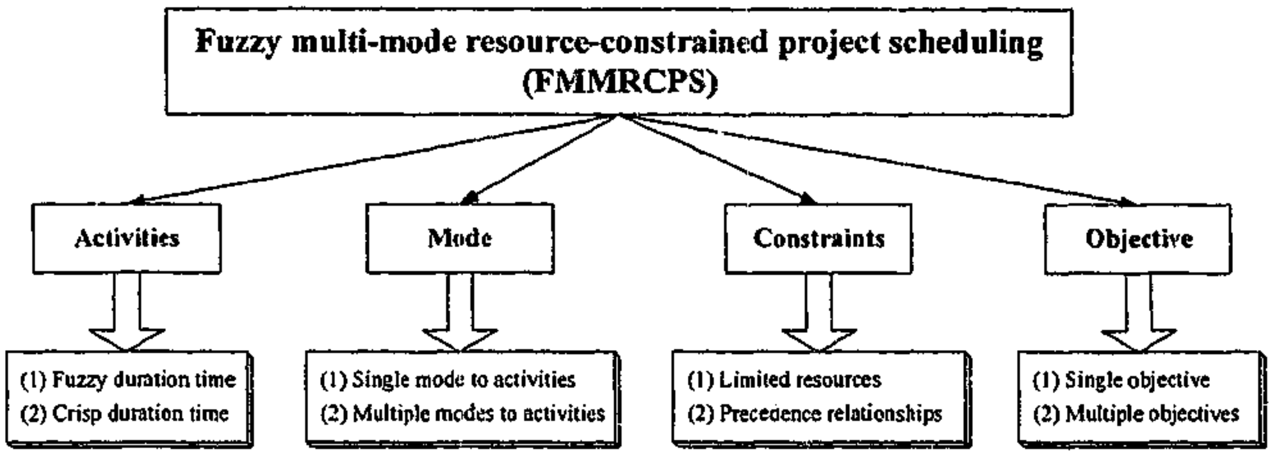

Fuzzy multi-mode resource-constrained project scheduling (FMMRCPS)
Activities
Mode
Constraints
Objective
(1) Fuzzy duration time
(2) Crisp duration time
(1) Single mode to activities
(2) Multiple modes to activities
(1) Limited resources
(2) Precedence relationships
(1) Single objective
(2) Multiple objectives

Figure 1.2 Scheduling problem description

A number of activities in a project may have never been performed previously or some activities may have been carried out for only a limited number of times. Precise information on these activities is often unavailable or not sufficient for use in practical settings. The uncertainty relating to fuzziness should be taken into consideration in real life projects. Of course, the duration times of some activities in the project may be quite certain and therefore deterministic. An instance of an activity is to test the quality of a machine required to be run for a specific time. The project scheduling problem model therefore should cover both fuzzy and crisp activity duration times, as shown in Figure 1.2.

In a large number of realistic instances, activities of a project may be possible to be performed in one of several executive options under different workable resources settings. Each executive option of an activity, referred to as an executive mode, has its own duration time with its corresponding resource requirements. However, in some cases, a few activities of the project may only be allowed to have one executive mode. An activity of opening up an engine may permit only one executive mode under the resource requirement of a specific number of labourers and engineers. Neither extra human resources will improve the efficiency of this activity, nor will less staff be able to perform the activity. As pointed out in Figure 1.2, the generalised scheduling problem should have both single and multiple executive modes.

Constraints in project scheduling are another concern of project as shown in Figure 1.2. Precedence relationships are a logical connection imposed on related activities that often appears in practice. That is, one activity cannot start until a number of preceding activities have been completed. For example, in laying a gas pipe, digging the ditch must precede setting the supporting frame for the pipe which, in turn, must precede laying the gas pipe underground. Precedence relationships among activities can be visually represented in the form of a network diagram. The resource constraint is usually imposed on projects because the amount of resources available to a project is often fixed for the life of the project. This constraint can be technological restrictions or budget limitations. How resources are allocated to eligible activities is crucially important in managing limited total resources, and the efficiency of resource allocation greatly affects the quality of project scheduling.

The goal, referred to as the objective of project, often needs to be considered in order to satisfy specified requirements in scheduling, as shown in Figure 1.2 of single and multiple objectives. In terms of the single objective, minimising the project completion time is one of the most common interests for industry and organisations because the project is often required to complete as soon as possible under existing resources. However, sometimes project scheduling requires considering several performance measures. In the research, both the project completion time and the project cost have been taken into consideration simultaneously as these two objectives are often the major concerns of industry and organisations in terms of budget and timing.

As addressed above, the research problem must cover most practical situations encountered in project scheduling. The problem model defined in the research is easily modified and extended when a different single objective (other than project completion time) is required or different multiple objectives are imposed in project scheduling.

## 1.4 Objectives of the Research

The aim of this research is to develop a number of novel approaches that can efficiently and effectively solve both single and multiple objectives in fuzzy multi-mode resource-constrained project scheduling (FMMRCPS) of practical sizes under different requirements where activity duration times are fuzzy. Figure 1.3 shows a number of realistic approaches developed for solving the real-life sized FMMRCPS under a fuzzy environment.

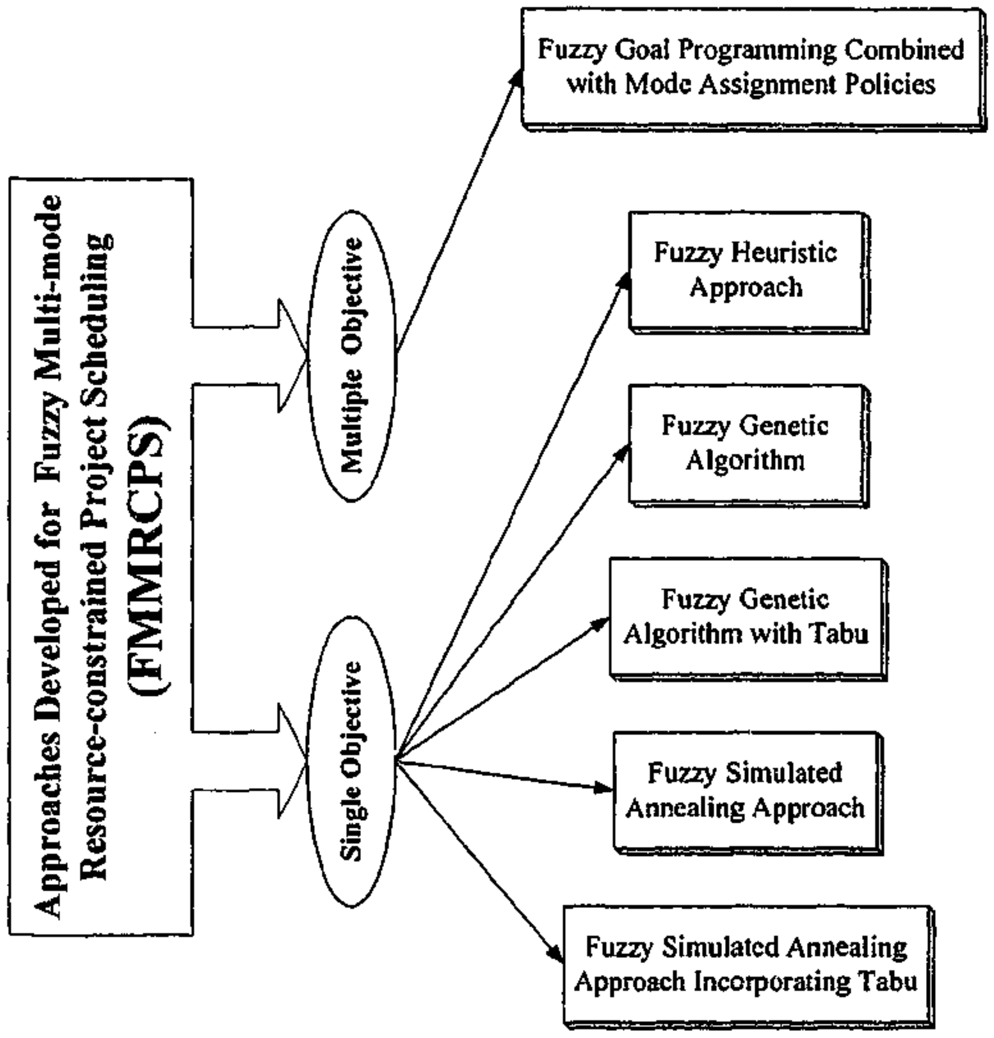

Approaches Developed for Fuzzy Multi-mode Resource-constrained Project Scheduling (FMMRCPS)
Multiple Objective
Single Objective
Fuzzy Goal Programming Combined with Mode Assignment Policies
Fuzzy Heuristic Approach
Fuzzy Genetic Algorithm
Fuzzy Genetic Algorithm with Tabu
Fuzzy Simulated Annealing Approach
Fuzzy Simulated Annealing Approach Incorporating Tabu

Figure 1.3 Approaches developed to FMMRCPS

To deal with multiple objectives in FMMRCPS as shown in Figure 1.3, a fuzzy hybrid goal programming approach, incorporating mode assignment policies, is developed. Activities of a project are often ridden with uncertainty because of lack of precise information on performance of activities. Fuzzy set theory is integrated into goal programming to handle fuzzy activity duration times caused by vagueness and imprecision. In addition, FMMRCPS is a complex combinatorial optimisation problem. To reduce the complex nature of FMMRCPS, mode assignment policies are proposed to facilitate the assignment of modes to activities. This approach effectively reduces FMMRCPS into single mode RCPS. Thus, the scheduling problem can be easily solved within this straightforward framework.

For tackling a single objective, five FMMRCPS approaches are developed in this research, which are shown in Figure 1.3 and briefly described as follows:

(a) Fuzzy heuristic approach

This approach applies both a set of priority rules and a mode assignment policy to the fuzzy forward and backward scheduling. Priority rules determine scheduling priorities of candidate activities at a given scheduling stage, and the mode policy decides the mode of selected candidate activities. The mode policy is specially designed to complete each path at the earliest possible time, subject to the condition of making all paths as even as possible.

(b) Fuzzy GA

This approach is based on the principle of the evolutionary process by surviving fit chromosomes and eliminating weak ones for gaining optimal solutions at the end of the process. To solve FMMRCPS effectively, a chromosome is carefully designed in order that it can be easily operated in the genetic means. In addition, generated chromosomes always guarantee feasibility.

(c) Fuzzy GA with tabu mechanism

The tabu mechanism is an intelligent mechanism for memorising the status of chromosomes that have been previously generated. This mechanism is specifically integrated into the fuzzy GA in order to stop the production of the same chromosomes that have already been generated in recent searches. Thus, it gives more diverse chromosome generations.

(d) Fuzzy SA approach

This approach is an iterative search algorithm, that is a metaphor of annealing. This approach is based on changing a temperature from a higher level to a lower level at a demand cooling ratio, in order to gain a minimum energy level, considered as the optimal solution of an FMMRCPS problem. To compute effectively, a solution representation is specially designed, in which only partial scheduling is required each time for neighbourhood generations, thus, reducing the computational time greatly.

(e) Fuzzy SA with tabu approach

This approach developed incorporates a short-term memory function of tabu into fuzzy SA. The function stores information about the solution attributes of previous searches for specific short period of time. This function, in combination with fuzzy SA, can prevent the revisiting of recently visited search space in a certain period. Thus, more areas can be searched for effectively obtaining an approximate globally optimal schedule.

The details of these five heuristic and metaheuristic approaches to FMMRCPS will be presented in Chapter 7, 8, and 9 respectively.

## 1.5 Outline of the Thesis

The thesis is divided into twelve chapters. The first two chapters are the foundation of fuzzy set theory and review of project scheduling. The remaining ten chapters are associated with my research work. Figure 1.4 illustrates the overall structure of the thesis. It is the framework of the research conducted during my PhD studies in the methodology developments on the basis of the research problem domain I have defined in this thesis.

Prior to addressing a new project scheduling problem model involving fuzziness, Chapter 2 presents the basic concepts of fuzzy set theory related to this research effort, facilitating the understanding of these fundamental concepts and their general knowledge of the applications in approximating human subjective assessments and perceptions.

To pave the way for developing a number of novel approaches to fuzzy project scheduling, Chapter 3 reviews the history of project scheduling, and presents the existing methodologies developed for different scheduling problem models. It also addresses the weakness of existing approaches developed, and the underlying motivation for this research is also discussed.

Chapter 4 addresses the practical significance in the definition of a new scheduling problem model. The new problem model presented here has the capacity of covering most possible practical situations and different requirements. Thus, the new defined problem model gives the foundation for developing a number of novel and practical approaches.

Chapter 5 presents the fuzzy scheduling mechanism, including fuzzy forward and backward scheduling. This is one of components of the fuzzy heuristic and metaheuristic approaches developed. Examples of fuzzy arithmetic are also given to demonstrate how fuzzy set theory is applied to the fuzzy scheduling mechanism.

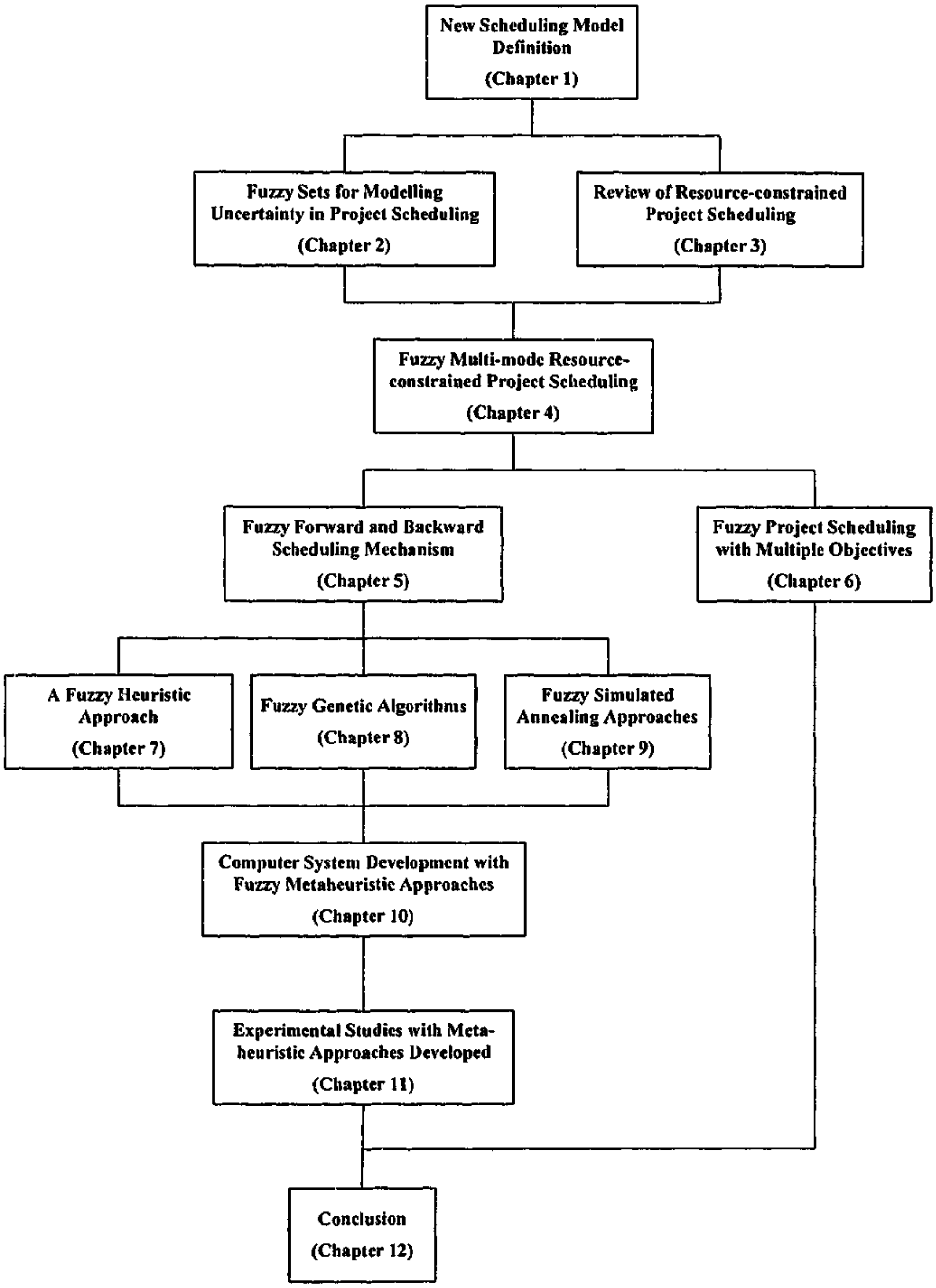

New Scheduling Model Definition (Chapter 1)
Fuzzy Sets for Modelling Uncertainty in Project Scheduling (Chapter 2)
Review of Resource-constrained Project Scheduling (Chapter 3)
Fuzzy Multi-mode Resource-constrained Project Scheduling (Chapter 4)
Fuzzy Forward and Backward Scheduling Mechanism (Chapter 5)
Fuzzy Project Scheduling with Multiple Objectives (Chapter 6)
A Fuzzy Heuristic Approach (Chapter 7)
Fuzzy Genetic Algorithms (Chapter 8)
Fuzzy Simulated Annealing Approaches (Chapter 9)
Computer System Development with Fuzzy Metaheuristic Approaches (Chapter 10)
Experimental Studies with Meta-heuristic Approaches Developed (Chapter 11)
Conclusion (Chapter 12)

Figure 1.4 The outline of research framework

Chapter 6 presents a fuzzy hybrid goal programming approach for integrating both fuzzy set theory and mode assignment policies for solving multiple objectives in project scheduling under the fuzziness. This approach facilitates a complex multiple mode scheduling problem into a single mode problem. This approach can be readily extended to any specified multiple objectives in fuzzy project scheduling.

Chapter 7 proposes a fuzzy heuristic approach, based on both activity priority rules and mode assignment policies. This is a simple and robust way of handing multi-mode scheduling problems under fuzziness for minimising the fuzzy project completion time.

Chapter 8 presents two versions of fuzzy GAs for multi-mode project scheduling under fuzzy activity duration times. One is fuzzy GA alone and the other is GA combined with tabu mechanism. These two approaches can solve any practical-sized project scheduling efficiently and effectively under both multiple executive modes and fuzziness.

Chapter 9 presents two fuzzy SA: (a) a fuzzy SA approach, and (b) a fuzzy SA incorporating tabu mechanism. These two approaches provide the other type of metaheuristics for solving fuzzy multi-mode scheduling problems of realistic sizes, thus, providing the other way of dealing with fuzzy complex project scheduling.

Chapter 10 illustrates the design of individual modules in the system written in VB.net, and summaries the function of each module. In the system, four metaheuristic approaches: (a) a fuzzy GA, (b) a fuzzy GA with tabu, (c) a fuzzy SA approach, and (d) a fuzzy SA with tabu, are integrated to solve single or multiple mode project scheduling whatever activity duration times are scrip, or fuzzy or both.

Chapter 11 summaries the experiments conducted with these four metaheuristic approaches, and provides statistical analysis for evaluating all the approaches developed in this research. Parameters of individual approaches are also tested to examine how parameter values affect the efficiency and effectiveness in obtaining an approximate optimal schedule. The comparison analysis of the four metaheuristic approaches is also conducted with a number of the same sample projects.

Finally, Chapter 12 summaries six approaches developed in this research. The characteristics of each approach are highlighted with the discussion of their individual advantages. The main contributions of this research are summarised. This chapter ends with the suggestions made for future research.

# Chapter 2

# Fuzzy Set Theory for Uncertainty Modelling in Project Scheduling

## 2.1 Introduction

There are two kinds of uncertainty we deal with in many areas such as engineering and management. One form of uncertainty is concerned with randomness, first proposed by Jakob Bernoulli (1654-1705). This phenomenon of uncertainty can be modelled by probability theory, in which the probabilistic distribution can be obtained using experimentation in statistics from precise observation, or from available historic information. Since the late 1920s, because of important discoveries in atomic physics and quantum mechanics, randomness has become a necessary theoretical base to provide the probability of results for its nature (Galambos 1995). In this research, we mainly deal with the other form of uncertainty that is related to imprecision and vagueness. Due to the characteristics of project scheduling, the duration of each activity in a project must be determined before the scheduling can be undertaken. However, some activities in a project may not have been performed or some activities may lack sufficient historic data. Also environmental changes in some activities can cause the vagueness in the assessment of these activity durations. This phenomenon of uncertainty has to be assessed by use of human knowledge or experience. Such uncertainty must be modelled by fuzzy set theory, also called possibility theory founded by Zadeh in 1965.

The term 'fuzzy sets' has become commonly used as an abbreviated way of describing vague and imprecise information. But the term has sometimes been misused and misunderstood (Dubois *et al.* 2000). To clarify the idea of fuzzy concepts, the doctrine of fuzzy sets was clearly redefined by Gupta in 1977 as "*a body of concepts and techniques aimed at providing a systematic framework for dealing with the vagueness and imprecision inherent in human thought process*". In his definition, three keywords: vagueness, imprecision and thinking are emphasised as the philosophical base forming the doctrine of fuzzy set theory (Negoita 2000, Dubois *et al.* 2000)

Vagueness exists in our natural language when the meaning of language does not sharply define boundaries. The modifiers in our language such as "**very**" and "**almost not**" are commonly used in our daily lives to describe the true level of membership in a certain class. For example, the statement of "he is **very** tall" means that he has a higher level toward the class of "tall people group". Similarly if this person is "**almost not**" tall, the truth level falls into the class of "tall people group" is extremely lower. Vagueness also often encounters in project scheduling, making it difficult or impossible to determine precise durations of some activities. In such cases, the project expert often has to intuitively assess durations in linguistic terms such as "**most possibly**" and "**impossibly**". To determine such activity durations, the project expert may describe that a particular activity "**most possibly**" finishes in 7 to 9 days, and "**impossibly**" finishes in 4 days or beyond 13 days by the optimistic and pessimistic views respectively. Such vague descriptions can be characterised by their degree of membership to depict a grade of fuzziness in the class.

Imprecision deals with the confidence in the accurate measurements, more particularly concerned with numerical imprecision. In many circumstances, due to lack of precise information, the measurement of accuracy in confidence has to be expressed only in the level of degree (Novák *et al.* 1999). In the situation of project scheduling, data related to some activities may sometimes have only limited sample information available so that applying any stochastic approaches to handle the uncertainty may be inappropriate. In such cases, fuzzy set theory is the only way to provide approximation measurements in the degree of confidence.

Thinking is the process of mental creation relying on our knowledge and expertise when problems need to be solved in the environment of vagueness. In project scheduling, the assessment of some activity durations may have to rely on this thinking process using the project expert's knowledge and experience. Naturally, linguistics influence our thought, providing a means of expressing our perception and understanding of reality, or describing an event in human experience. Fuzzy set theory reflects how people think. It can model our sense of words, common sense and decision making. The linguistic form of knowledge or the expert rules-base is capable of analysis and logical tests in many fuzzy systems because the process facilitates the quantification of human nature and intelligence (Negnevitsky 2002).

To better understand how fuzzy set theory can be applied to the vague and imprecise environment in mathematical form, in the following, fuzzy sets, its operations, and fuzzy number ranking will be illustrated.

## 2.2 Fuzzy Notation and Terminology

### 2.2.1 Fuzzy Set

The classical set has a "crisp" definition in which, the elements are either completely inside or completely outside of the set. So the classical set $A$ can be defined as a binary membership function $\mu_A(x)$ :

$$\mu_A(x)=\begin{cases}1 & \text{iff} \quad x\in A\\ 0 & \text{iff} \quad x\notin A\end{cases} \tag{2.1}$$

A fuzzy set is a set whose elements may only partially belong to that set in comparison with the classical set. In this case, an element in a fuzzy set can be a member of that set to some degree. Therefore, the membership function of a fuzzy set is a flexible way to describe vague and imprecise situations in some grades. Let a fuzzy set $\widetilde{A}$ be in the universal set $X$. The membership function $\mu_{\widetilde{A}}(x)$ of $x$ in $\widetilde{A}$ is the degree of membership in the closed unit interval $[0,1]$ on the real line $\Re$. It can be described as

$$\widetilde{A}=\left\{(\frac{\mu_{\widetilde{A}}(x)}{x}) \,\middle|\, x\in X, \mu_{\widetilde{A}}(x)\in[0,1]\right\} \tag{2.2}$$

For continuous $X$, $\widetilde{A}$ may be precisely written as

$$\widetilde{A}=\int_X \frac{\mu_{\widetilde{A}}(x)}{x} \tag{2.3}$$

For discrete $X$, $\widetilde{A}$ is usually represented as

$$\widetilde{A}=\sum \frac{\mu_{\widetilde{A}}(x)}{x} \tag{2.4}$$

The notations "∫", "∑" used here refer to set union rather than to arithmetic summation. Similarly "/" is used to connect an element and its membership value, and has nothing to do with arithmetic division.

From the description of the above classical and fuzzy sets, it is clear that fuzzy set can be considered to be a generalization of the classical set. When the degree of membership is restricted to be only 0 or 1, a fuzzy set becomes a classical set.

### 2.2.2 Support

Given a fuzzy set $\widetilde{A}$ of the universal set $X$, its support, commonly denoted as Supp($\widetilde{A}$), will be the set all of whose elements $x$ in $X$ have nonzero degree of the membership function. It is defined as

$$\text{Supp}(\widetilde{A})=\{x \in X \mid \mu_{\widetilde{A}}(x) > 0\} \tag{2.5}$$

Clearly, in Formula (2.5), the support of a fuzzy set contains elements having all positive membership grades.

### 2.2.3 Core

The core of a fuzzy set $\tilde{A}$ in the universal set $X$, symbolised as Core($\tilde{A}$), is the set of all elements, $x$ in $X$, whose membership function $\mu_{\tilde{A}}(x)$ must be 1. It can be described as

$$\text{Core}(\tilde{A}) = \{x \in X \mid \mu_{\tilde{A}}(x) = 1\} \tag{2.6}$$

Obviously, the core of a fuzzy set is a crisp set because the membership functions of all the elements in that set are equal to 1. That is, all elements are in that set.

### 2.2.4 α-cut or α-level Set

The α-cut, or α-level set of a fuzzy set $\tilde{A}$ in the universal set $X$, is a crisp set, denoted as $A_\alpha$ that consists of all elements $x$ in $X$, whose membership functions must be greater than or equal to $\alpha$. It can be mathematically defined as

$$A_\alpha = \{x \mid \mu_{\tilde{A}}(x) \geq \alpha, \alpha \in [0,1]\} \tag{2.7}$$

Similarly, the strong α-cut or strong α-level set of a fuzzy set $\tilde{A}$, signified as $A_{\alpha^+}$, is the set whose elements are greater than and exclusive of $\alpha$. It is shown as

$$A_{\alpha^+} = \{x \mid \mu_{\tilde{A}}(x) > \alpha, \alpha \in [0,1]\} \tag{2.8}$$

Using the notations of α-cut and strong α-cut, the support and core of a fuzzy set $\tilde{A}$ can be expressed as the α-cut form respectively

$$\text{Supp}(\tilde{A}) = A_{0^+} \tag{2.9}$$

$$\text{Core}(\tilde{A}) = A_1 \tag{2.10}$$

The α-cut concept in fuzzy set theory is applied to express the view of the project experts on the minimum degree of acceptance when assessing activity duration times for a project. The level of α-cut depends on the project experts' confidence in their assessments of each activity duration, using their knowledge and their perceptions of the project situation.

### 2.2.5 Convexity

A fuzzy set $\tilde{A} := \{(x, \mu_{\tilde{A}}(x)) | x \in \Re\}$ of a set of real numbers is convex, if, and only if its membership function $\mu_{\tilde{A}}(x)$ satisfies the following property

$$\mu_{\tilde{A}}(\lambda x_1 + (1-\lambda)x_2) \geq \min(\mu_{\tilde{A}}(x_1), \mu_{\tilde{A}}(x_2)), \forall x_1, x_2 \in X, \forall \lambda \in [0,1] \qquad (2.11)$$

Or, alternatively, fuzzy set $\tilde{A}$ is convex if all its α-cuts are convex. Figure 2.1 shows a convex and a nonconvex fuzzy set.

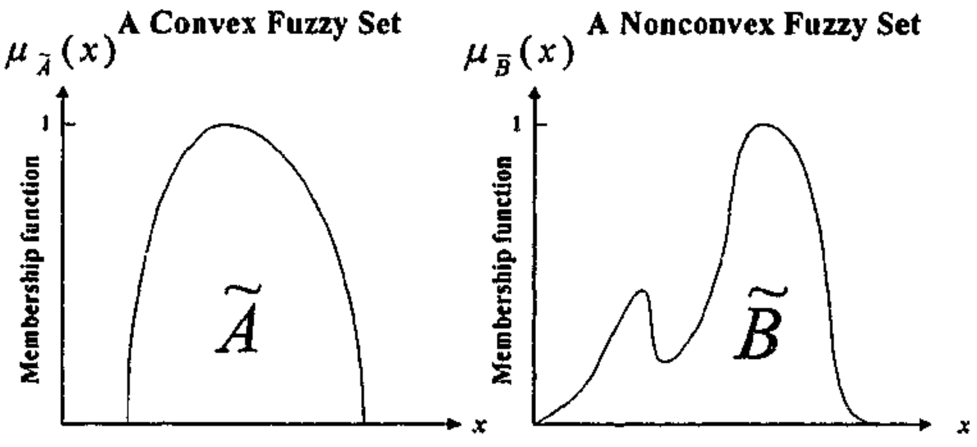


Figure 2.1 Convex and nonconvex fuzzy sets

Fuzzy sets applied to project scheduling in this research, must be convex to satisfy its membership function as required by the principle of fuzzy set theory.

### 2.2.6 Normality

A fuzzy set $\tilde{A}$ is normal if, and only if, its core is nonempty. In other words, we can always find at least one element whose membership function is equal to 1. The nonnality of a fuzzy set can be expressed as

$$\mu_{\tilde{A}}(x)=1, \exists x \in X \tag{2.12}$$

This property is important in our fuzzy application where all the fuzzy sets applied must satisfy the condition of Formula (2.12).

### 2.2.7 Fuzzy Numbers

A fuzzy number $\tilde{A}$ is a fuzzy set on real numbers that satisfy the condition of normality and convexity, with a piecewise continuous membership function (Zadeh 1965, Dubois *et al.* 1988). It has four points $a_1 \leq a_2 \leq a_3 \leq a_4$ with the following properties:

(1) $\mu_{\tilde{A}}(x)=0$, for every $x \in (-\infty, a_1) \cup (a_4, \infty)$;

(2) $\mu_{\tilde{A}}(x)$ is increasing on $[a_1, a_2]$ and decreasing on $[a_3, a_4]$;

(3) $\mu_{\tilde{A}}(a_1)=\mu_{\tilde{A}}(a_4)=0$ and $\mu_{\tilde{A}}(x)=1$, for every $x \in [a_2, a_3]$.

Clearly, through the above definition, fuzzy numbers are the most basic type of fuzzy sets. The most common fuzzy numbers used in many applications are trapezoidal, triangular, Gaussian and generalised Bell-shaped fuzzy numbers as shown in Figure 2.2.

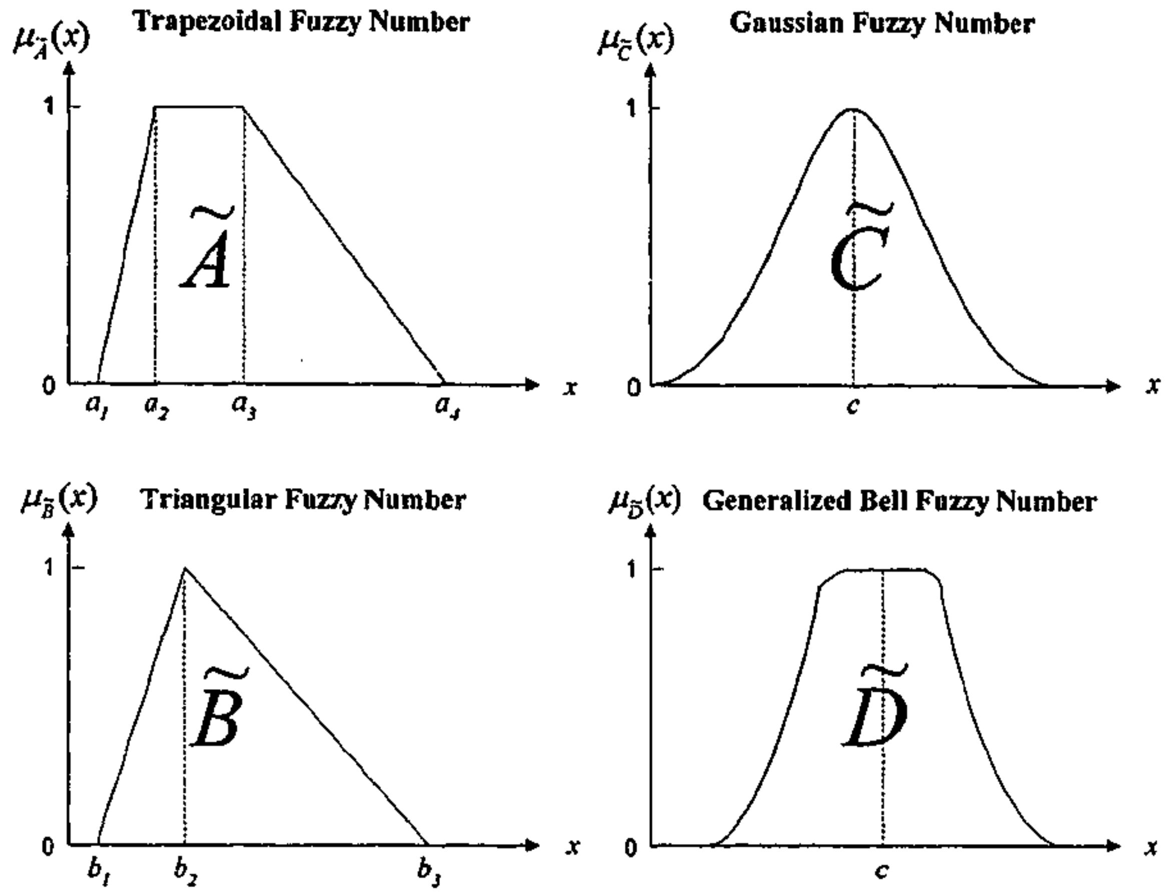


Figure 2.2 The common types of fuzzy numbers

The author worked in industry for many years and also consulted with some project experts during a recent tour of study and, on the basis of this experience, believes that trapezoidal and triangular fuzzy numbers can well represent the nature of fuzziness in terms of the assessment of activity durations. Thus two linear approximations for trapezoidal and triangular fuzzy numbers are applied to project scheduling in this research. Details and an explanation of the use of trapezoidal and triangular fuzzy numbers in project scheduling can be seen in Section 4.4.2 of Chapter 4. In the following, these two types of fuzzy numbers are discussed in the mathematical form.

A trapezoidal fuzzy number $\tilde{A}$ is a flat fuzzy number that can be represented by 4-tuples $(a_1, a_2, a_3, a_4)$ as shown in Figure 2.2, where $a_1$ and $a_4$ are the lower and upper bounds of the support of fuzzy number, $\tilde{A}$, representing optimistic and pessimistic values respectively, while $a_2$ and $a_3$ are the lower and upper modal values that means the interval from $a_2$ to $a_3$ is a most certain approximate assessment. The membership function of a trapezoidal fuzzy number $\widetilde{A}$ can be expressed as

$$\mu_{\widetilde{A}}(x)=\begin{cases}0, & x\le a_1,\\ \dfrac{x-a_1}{a_2-a_1}, & a_1\le x\le a_2,\\ 1, & a_2\le x\le a_3,\\ \dfrac{a_4-x}{a_4-a_3}, & a_3\le x\le a_4,\\ 0, & a_4\le x.\end{cases} \tag{2.13}$$

Or, alternatively given concisely as

$$\mu_{\widetilde{A}}(x)=\max(\min(\frac{x-a_1}{a_2-a_1},1,\frac{a_4-x}{a_4-a_3}),0)) \tag{2.14}$$

However, a triangular fuzzy number can be considered a special case of the trapezoidal fuzzy number when the lower and upper modal values are same $(a_2=a_3)$. As shown in Figure 2.2, a triangular fuzzy number, $\widetilde{B}$, has 3-tuples ($b_1\le b_2\le b_3$). $b_2$ is the most possible assessment value, and $b_1$ and $b_3$ are the lower and upper bounds expressing the fuzzy assessment of a fuzzy number. The membership function $\mu_{\widetilde{B}}(x)$ of a triangular fuzzy number, $\widetilde{B}$, can be described as

$$\mu_{\widetilde{B}}(x)=\begin{cases}0, & x\le b_1,\\ \dfrac{x-b_1}{b_2-b_1}, & b_1\le x\le b_2,\\ \dfrac{b_3-x}{b_3-b_2}, & b_2\le x\le b_3,\\ 0, & b_3\le x.\end{cases} \tag{2.15}$$

Or, alternatively, given concisely as

$$\mu_{\widetilde{B}}(x)=\max(\min(\frac{x-b_1}{b_2-b_1},\frac{b_3-x}{b_3-b_2}),0) \tag{2.16}$$

## 2.3 Operations on Fuzzy Sets

In this section, basic operations on fuzzy sets related to this research are introduced. The standard operations on fuzzy sets are performed when the membership grades of fuzzy sets must be restricted to the set $\{0,1\}$. To be meaningful in practice, the process of these operations should always satisfy the cutworthy or the strong cutworthy properties. The standard fuzzy set operations can be viewed as a generalisation of the corresponding classical set operations (Klir and Yuang 1995, Lin and Lee 1995).

Let the fuzzy sets $\widetilde{A}$ and $\widetilde{B}$ be in the same universal set, $X$, Their corresponding membership functions are $\mu_{\widetilde{A}}(x)$ and $\mu_{\widetilde{B}}(x)$ respectively. The standard operations on the fuzzy sets $\widetilde{A}$ and $\widetilde{B}$ can be defined as below:

(1) Equality

Fuzzy set $\widetilde{A}$ is considered equal to a fuzzy set $\widetilde{B}$, denoted as $\widetilde{A}=\widetilde{B}$, iff

$$\mu_{\widetilde{A}}(x)=\mu_{\widetilde{B}}(x), \forall x \in X \tag{2.17}$$

(2) Containment (Subset)

Fuzzy set $\widetilde{A}$ is contained in fuzzy set $\widetilde{B}$, denoted as $\widetilde{A}\subset\widetilde{B}$, iff

$$\mu_{\widetilde{A}}(x)\leq\mu_{\widetilde{B}}(x), \forall x \in X \tag{2.18}$$

(3) Complement (Negation)

The complement of fuzzy set $\widetilde{A}$, denoted as $\widetilde{A}^c$ or NOT $\widetilde{A}$, is defined as

$$\mu_{\widetilde{A}^c}(x)=1-\mu_{\widetilde{A}}(x), \forall x \in X \tag{2.19}$$

(4) Union (Disjunction)

The union of two fuzzy sets $\widetilde{A}$ and $\widetilde{B}$, denoted as $\widetilde{A} \cup \widetilde{B}$, is defined as

$$\mu_{\widetilde{A} \cup \widetilde{B}}(x) = \mu_{\widetilde{A}}(x) \vee \mu_{\widetilde{B}}(x) = \max(\mu_{\widetilde{A}}(x), \mu_{\widetilde{B}}(x)) \qquad (2.20)$$

(5) Intersection (Conjunction)

The intersection of two fuzzy sets $\widetilde{A}$ and $\widetilde{B}$, denoted as $\widetilde{A} \cap \widetilde{B}$, is defined as

$$\mu_{\widetilde{A} \cap \widetilde{B}}(x) = \mu_{\widetilde{A}}(x) \wedge \mu_{\widetilde{B}}(x) = \min(\mu_{\widetilde{A}}(x), \mu_{\widetilde{B}}(x)) \qquad (2.21)$$

## 2.4 Fuzzy Arithmetic

To model project scheduling under uncertainty, fuzzy numbers are often used to define the fuzziness and imprecision of the problem domain. However, fuzzy numbers are required for appropriate implementation. Therefore, fuzzy arithmetic is an important tool for operating on fuzzy numbers in problem solving. Here only arithmetic operations on triangular and rectangular fuzzy numbers are illustrated, since these two types of fuzzy numbers are applied to project scheduling.

Let $\widetilde{A}$ and $\widetilde{B}$ be fuzzy numbers, and $*$ denote any basic fuzzy arithmetic operations such as fuzzy addition (+), subtraction (-), multiplication (×), division (/) and others. Any operation $\widetilde{A} * \widetilde{B}$ can be defined as a fuzzy set on $\Re$ and expressed in the following form:

$$\eta_{\widetilde{A} * \widetilde{B}}(z) = \max_{x * y = z} \{\min[(\mu_{\widetilde{A}}(x), \mu_{\widetilde{B}}(y)]\} \qquad (2.22)$$

In triangular fuzzy numbers, there are 3-tuples as shown in Figure 2.2. Let $\widetilde{A}=(a_1,a_2,a_3)$ and $\widetilde{B}=(b_1,b_2,b_3)$ be two triangular fuzzy numbers, the basic fuzzy arithmetic operations can be expressed as

$$\widetilde{A}+\widetilde{B}=(a_1+b_1,a_2+b_2,a_3+b_3) \tag{2.23}$$

$$\widetilde{A}-\widetilde{B}=(a_1-b_3,a_2-b_2,a_3-b_1) \tag{2.24}$$

$$\widetilde{A}\times\widetilde{B}=(a_1\times b_1,a_2\times b_2,a_3\times b_3) \tag{2.25}$$

$$\widetilde{A}/\widetilde{B}=(\frac{a_1}{b_3},\frac{a_2}{b_2},\frac{a_3}{b_1}) \tag{2.26}$$

$$1/\widetilde{A}=(\frac{1}{a_3},\frac{1}{a_2},\frac{1}{a_1}) \tag{2.27}$$

$$\max(\widetilde{A},\widetilde{B})=(\vee(a_1,b_1),\vee(a_2,b_2),\vee(a_3,b_3)) \tag{2.28}$$

$$\min(\widetilde{A},\widetilde{B})=(\wedge(a_1,b_1),\wedge(a_2,b_2),\wedge(a_3,b_3)) \tag{2.29}$$

where $\vee$, $\wedge$ symbolise maximum, and minimum operations for fuzzy numbers.

Similarly, trapezoidal fuzzy numbers have 4-tuples as shown in Figure 2.2. Let $\widetilde{A}=(a_1,a_2,a_3,a_4)$ and $\widetilde{B}=(b_1,b_2,b_3,b_4)$ be two trapezoidal fuzzy numbers. The arithmetic operations on this type of fuzzy numbers are defined as

$$\widetilde{A}+\widetilde{B}=(a_1+b_1,a_2+b_2,a_3+b_3,a_4+b_4) \tag{2.30}$$

$$\widetilde{A}-\widetilde{B}=(a_1-b_4,a_2-b_3,a_3-b_2,a_4-b_1) \tag{2.31}$$

$$\widetilde{A}\times\widetilde{B}=(a_1\times b_1,a_2\times b_2,a_3\times b_3,a_4\times b_4) \tag{2.32}$$

$$\widetilde{A}/\widetilde{B}=(\frac{a_1}{b_4},\frac{a_2}{b_3},\frac{a_3}{b_2},\frac{a_4}{b_1}) \tag{2.33}$$

$$1/\widetilde{A}=(\frac{1}{a_4},\frac{1}{a_3},\frac{1}{a_2},\frac{1}{a_1}) \tag{2.34}$$

$$\max(\widetilde{A}, \widetilde{B}) = (\vee(a_1, b_1), \vee(a_2, b_2), \vee(a_3, b_3), \vee(a_4, b_4)) \tag{2.35}$$

$$\min(\widetilde{A}, \widetilde{B}) = (\wedge(a_1, b_1), \wedge(a_2, b_2), \wedge(a_3, b_3), \wedge(a_4, b_4)) \tag{2.36}$$

Further details in regard to fuzzy arithmetic can be found in Dubois and Prade (1980), Zimmermann (1996), Klir and Yuan (1995), and Negoita (2000).

## 2.5 Fuzzy Ranking

In project scheduling, fuzzy numbers are often required to be compared in each stage of scheduling in which a group of fuzzy numbers must be ordered in the scalar value. However, fuzzy numbers are often not straightforward in terms of scalar value compared to crisp numbers. To determine which fuzzy number is bigger or smaller, an appropriate ranking method is needed for ordering fuzzy numbers. Therefore, ranking of fuzzy numbers is an important issue in the project scheduling.

A number of ranking methods have been proposed so far, including Yager (1981), Li and Lee (1987), Campos and Munoz (1989), Liou and Wang (1992). However, there is no single approach that can produce a satisfactory result in every situation: some may generate counter-intuitive results and others are not discriminative enough (Chen and Hwang 1992, Deng and Yeh 1996).

Cheng (1998) developed a new distance approach for fuzzy number comparisons based on the calculation of the distance means from the original point to the centroid point to rank fuzzy numbers. He mathematically proves that his approach overcomes the above-mentioned shortcomings. Cheng's fuzzy ranking method is applied in my research in project scheduling for fuzzy number comparison.

Let $\mu_{\widetilde{A}}^{L}(x)$ and $\mu_{\widetilde{A}}^{R}(x)$ be strictly continuous left spread, and right spread of the membership function of a fuzzy number $\widetilde{A}$, and $g_{\widetilde{A}}^{L}(x)$ and $g_{\widetilde{A}}^{R}(x)$ be the inverse functions of $\mu_{\widetilde{A}}^{L}(x)$ and $\mu_{\widetilde{A}}^{R}(x)$ respectively. $\bar{x}_0$ and $\bar{y}_0$ are denoted as the centroid values both in the horizontal and vertical axes. The centroid point $(\bar{x}_0, \bar{y}_0)$ of a fuzzy number $\widetilde{A}$ can be defined as

$$\bar{x}_0(\widetilde{A}) = \frac{\int_{a_1}^{a_2} \left[x\mu_{\widetilde{A}}^{L}(x)\right]dx + \int_{a_2}^{a_3} x\,dx + \int_{a_3}^{a_4}\left[x\mu_{\widetilde{A}}^{R}(x)\right]dx}{\int_{a_1}^{a_2} \mu_{\widetilde{A}}^{L}(x)dx + \int_{a_2}^{a_3} dx + \int_{a_3}^{a_4}\mu_{\widetilde{A}}^{R}(x)dx}$$

$$\bar{y}_0(\widetilde{A}) = \frac{\int_0^1 \left[y g_{\widetilde{A}}^{L}(y)\right]dy + \int_0^1 \left[y g_{\widetilde{A}}^{R}(y)\right]dy}{\int_0^1 g_{\widetilde{A}}^{L}(y)dy + \int_0^1 g_{\widetilde{A}}^{R}(y)dy} \qquad (2.37)$$

The fuzzy ranking index, based on a distance between the centroid point and the original point of a fuzzy number $\widetilde{A}$, can be expressed as

$$R(\widetilde{A}) = \sqrt{(\bar{x}_0)^2 + (\bar{y}_0)^2} \qquad (2.38)$$

Because triangular and trapezoidal fuzzy numbers are used in this research, Formula (8) can be modified as

(1)

Triangular
Fuzzy Number
$\widetilde{A}=(a_1,a_2,a_3)$

$$\bar{x}_0 = \frac{a_1 \times (a_2^3 - a_3^3) - a_2 \times (a_1^3 - a_3^3) + a_3 \times (a_1^3 - a_2^3)}{3 \times \left[a_1 \times (a_2^2 - a_3^2) - a_2 \times (a_1^2 - a_3^2) + a_3 \times (a_1^2 - a_2^2)\right]}$$

$$\bar{y}_0 = \frac{a_1 + 4 \times a_2 + a_3}{3 \times (a_1 + 2 \times a_2 + a_3)} \qquad (2.39)$$

(2)

Trapezoidal
Fuzzy number
$\widetilde{A}=(a_1,a_2,a_3,a_4)$

$$\bar{x}_0 = \frac{a_4^2 + a_3^2 - a_2^2 - a_1^2 + a_3 \times a_4 - a_1 \times a_2}{3 \times (a_4 + a_3 - a_2 - a_1)}$$

$$\bar{y}_0 = \frac{a_1 + 2 \times a_2 + 2 \times a_3 + a_4}{3 \times (a_1 + a_2 + a_3 + a_4)} \qquad (2.40)$$

Let $\tilde{A}_i$, $\tilde{A}_j$ be any fuzzy numbers in set $\Re$, the comparison of fuzzy numbers has the following properties when obtaining ranking indices by formula (2.38).

$$\begin{aligned} &(1)\ \text{If}\ R(\tilde{A}_i) > R(\tilde{A}_j), \text{then}\ \tilde{A}_i > \tilde{A}_j, \\ &(2)\ \text{If}\ R(\tilde{A}_i) = R(\tilde{A}_j), \text{then}\ \tilde{A}_i = \tilde{A}_j, \\ &(3)\ \text{If}\ R(\tilde{A}_i) < R(\tilde{A}_j), \text{then}\ \tilde{A}_i < \tilde{A}_j. \end{aligned} \tag{2.41}$$

## 2.6 Concluding Remarks

Preliminary knowledge of fuzzy set theory has been introduced in this chapter. To clarify the concept of fuzzy sets, the phenomenon of imprecision and vagueness has been addressed. The contents of fuzzy sets and fuzzy numbers relevant to this research, are explained in detail. Fuzzy operations on fuzzy sets and fuzzy numbers are also presented in this chapter. Ranking of fuzzy numbers is important in project scheduling. To overcome the problems of counter-intuitiveness and insufficient discrimination in fuzzy number comparison, Cheng's new distance approach in regard to fuzzy numbers is introduced. By dealing only with triangular and rectangular fuzzy numbers, Cheng's new fuzzy raking approach has been modified so that it is suitable for the comparison of these two types of fuzzy numbers.

# Chapter 3

# A Review of Resource-Constrained Project Scheduling

## 3.1 Introduction

The basic idea of project scheduling dates back a few thousand years to the construction of the huge project of the Great Wall in China. This project started in the 7th century B.C. It required enormous amounts of different kinds of resources such as varied sorts of labour and materials. To be able to complete this huge project in time under the emperor's order, it was divided into immense pieces of work (called activities). Each under the charge of a specified official, for each activity durations were estimated and resources assigned. The timing of each activity, related to others was specified in a logical order. Such a sequence of planning is the essence of project scheduling.

Although the concept of project scheduling was already applied to some projects in ancient times, the concept became the subject of academic research only in recent times. Precedence relationships and limited resource constraints have been studied as an academic research field for only around 40 years, being dealt with, in the 1960's, by Kelley (1963), Lambourn (1963) and Wiest (1964). This kind of problem is formally called resource-constrained project scheduling (RCPS).

The type of problem raised by much project planning is a typical combinatorial problem that requires assigning discrete numerical values to some finite set of variables $x$ under the satisfaction of a set of constraints, in order to minimise and maximise some objective function $f(x)$. Therefore, any scheduling problem under constraints is computationally complex and hard, and the corresponding optimisation problem will be more complicated and harder. It becomes an NP-hard problem (Stockmeyer 1992, Shmoys and Tardos 1992).

This field has attracted much research because of the problem of NP-hardness, and as such many different approaches have been developed. But many issues in this field still require vigorous research since RCPS is now widely applied in modern, real life situations, such as building construction, engineering, and IT projects.

This chapter will review RCPS in both single and multiple modes, developed from past to present in both deterministic and uncertain situations. In section 3.2, the terms, definitions, and basic methodologies applied in project scheduling are explained. This section also describes the standard RCPS mode and briefs its extended modes with their mathematical formulations. Section 3.3 addresses the details of the multi-mode RCPS and its mathematical models on which my research is particularly focused. Section 3.4 reviews the current development in exact algorithms for both single and multiple modes. In section 3.5, scheduling schemes and common priority rules are described, followed by concisely introducing a number of priority rule-based heuristics used in both single and multiple modes. Section 3.6 presents the recent development of decision support systems applied in project scheduling. Section 3.7 highlights a number of metaheuristics developed for single and multiple modes in RCPS. Problem domains, methodologies, and approaches addressed in these sections are limited to the conventional way that has not taken the uncertainty into account caused by imprecise information related to a project. Uncertainty inherently exists in many real project scheduling problems. For this reason, from section 3.8, the nature of the uncertainty that exists in project scheduling will be addressed. In Section 3.9, the current development in fuzzy project scheduling will be intensively reviewed, followed by the contributions I made through my PhD studies. Section 3.10 will be a conclusion.

## 3.2 Description of Resource-Constrained Project Scheduling

To better understand RCPS, four important elements of project scheduling will be introduced. These elements are: resources, activities, precedence relations, and performance measures. This section will also provide a definition of project scheduling.

### 3.2.1 Resource Categories

The performance of a project requires different types of resources. It is therefore necessary to determine, before the project is executed, what resource categories are required. Slowinski (1980, 1981), Welglarz (1980) and Talbot (1982) proposed three basic types of resources for project scheduling: renewable, non-renewable and doubly-constrained. These resource categories will be detailed in the following subsection 3.2.1.1. However, some researchers have proposed additional resource categories in the literature, such as partially renewable, dedicated and cumulative resources. Although these additional categories are not commonly used in RCPS, they will be briefly explained.

#### 3.2.1.1 Basic Resource Types

**Renewable resources** are those types of resources where the amount of resources consumed by a job from the total availability in a project, can be renewed when the job is finished. For clarity, an example is given here: the availability of one renewable resource is 10 units in a project. Job 1 uses 3 units at the time period 0 until it finishes at the time period 5, and job 2 takes 4 units at the time period 2 until it is completed at the time period 7. In this case, the availability of the resource will be 7 units left from period 0 to period 2 because job 1 uses 3 units of this resource, and from the time period 2, the availability of the resource will be changed to 3 units since job 2 takes 4 units at that time period 2. From the time period 5, the resource will be renewed to the availability of 6 since job 1 releases 3 units after it completes at that time period. From the time period 7, the resource will be further renewed to the availability of 10 since job 2 finishes at the time period 7. Clearly, the availability of renewable resources varies from period to period based on how much this type of resources is consumed or released by jobs at different time periods. The time period can be hours, days or weeks depending on the requirement of a real project. Electricians, mechanics and general workers listed in Table 3.1 are typical renewable resources in the overhaul project. The examples of equipment, machines, trucks and tools may also be considered members of the renewable resource category. Further details on renewable resources can be found in Weiss (1988).

**Non-renewable resources** are limited over the entire project life-span. This means, the total amount of this type of resources is available in limited quantity throughout the whole project and cannot be renewed from period to period. However, there is no limitation on consumption in one period unless resource requirement exceeds the total available quantity. Raw materials and energy could be regarded as non-renewable resources. In an overhaul project, resource 4, water listed in Table 3.1 is a typical non-renewable resource because water consumption is limited through the whole overhaul project.

**Doubly-constrained resources** are constrained both for each period and for the entire project. In such a category, the resource availability is limited from time to time and at the same time, the consumption cannot surpass the resource availability throughout the entire project. Therefore, doubly-constrained resources have the characteristics of both renewable and non-renewable resources and can be viewed as the combination of these two types of resources. Cash flow is a good example of doubly-constrained resource because cash flow is always controlled in a required level each day. Of course, the total cash flow for a project is also constrained so that the project budget is not exceeded.

Table 3.1 Durations and resource requirements of activities in an overhaul project

| Activity number $j$ | Activity description | Activity duration (hrs) $d_j$ | Electricians Resource 1 $k_{j1}$ | Mechanics Resource 2 $k_{j2}$ | General workers Resource 3 $k_{j3}$ | Water ($m^3$) Resource 4 $k_{j4}$ |
|---|---|---|---|---|---|---|
| 1 | dismantle engines | 20 | 1 | 3 | 2 | 0 |
| 2 | dismantle pumps | 15 | 1 | 2 | 2 | 0 |
| 3 | clean up hopper | 10 | 0 | 3 | 4 | 15 |
| 4 | purify cylinders | 30 | 0 | 3 | 5 | 5 |
| 5 | clean out pumps | 16 | 0 | 2 | 4 | 10 |
| 6 | wash up pipes | 9 | 0 | 1 | 2 | 20 |
| 7 | inspect hydraulic system | 10 | 1 | 2 | 2 | 1 |
| 8 | assemble engines | 25 | 0 | 3 | 4 | 0 |
| 9 | assemble pumps | 20 | 1 | 2 | 3 | 1 |
| 10 | check doors of hopper | 8 | 1 | 2 | 2 | 1 |
| 11 | examine drags | 12 | 1 | 2 | 2 | 0 |
| 12 | check up general circuit | 5 | 1 | 0 | 2 | 0 |
| 13 | clean dragheads | 6 | 0 | 2 | 2 | 2 |
| 14 | test run | 10 | 2 | 5 | 5 | 0 |

All the resource types discussed above are discrete, and the smallest amount of resources that can be taken by a job or work is represented as one unit and not allowed to be further divided. That is, any amount of these resources that would be taken by jobs or works in a project must be an integer number such as 1, 2, and 3.

### 3.2.1.2 Other Resource Types

**Continuously Divisible Resources** were proposed by Weglarz (1981). This is the type of renewable resources that can be continuously divisible unlike those mentioned above that may be allocated only in a discrete unit. For continuously divisible resources, even one unit can be further divided into decimal quantities in contrast to discrete resources. Electric current and some liquid materials can be considered as members of the continuously divisible resource category.

**Recyclable Resources** were introduced by Shewchuk and Chang (1995). This type of resources can be worn out during operations or jobs taken on a project. Recyclable resources start with certain units of each of this type of resources, and will be recyclable as each of this type of resources is fully consumed or worn out while being used.

In the beginning, each of these resources has a given quantity. If a certain quantity is consumed by a job or activity, the total amount of this resource will be reduced by this quantity. When all available quantity of this resource is fully consumed or it is said that this resource is worn out, a specific time allocation may be required for its reprocessing. For example, if a recyclable resource, a mould, is worn out or fully consumed, a specific time is needed to remove the mould, to send it to workshop and to receive and install the refurbished mould. Once the mould arrives back on site, the amount of this type of resources can be set back to the normal quantity as determined at the beginning of a project. For example, cuttings and drills may have a predetermined safe operation limit. As their life spans are reached, they need to be taken to a tool crib for repair and the specific time needs to be allowed for the period of absence during repair.

**Dedicated Resources** is a term suggested by Bianco *et al.* (1998) when dealing with multiple mode scheduling problems in the special circumstance where this kind of resources can be available only in one unit per period. Dedicated resources are, however, are renewable. This type of resources can be assigned to only one task or one job at one time. Whenever, two tasks or jobs may compete for this type of resources, they cannot be executed in parallel. In order to share such resources, only one task or job is allowed to be performed at one time.

**Partially Renewable Resources** were proposed by Böttcher *et al.* (1999) and Drexl *et al.* (2000) and are considered to have different capacities (or availabilities) on different subsets of periods. The application might best be explained with an example. Let $P(\pi)$ denote the subset of periods and $k_{p(\pi)}$ be the capacity or availability of partial renewable resources in those periods. Three workers are working on a project for two consecutive weeks, and each worker is allowed to work 9 days of weekdays and one day on weekends. One subset of periods for working on weekdays can be expresses as $P(1) = \{1, 2, ..., 5, 8, 9, ..., 12\}$ and the resource capacity in this subset of periods $P(1)$ will be described as $k_{P(1)} = 3 \times 9 = 27$. The other subset of periods of weekend is signified as $P(2) = \{6, 7, 13, 14\}$ and its resource capacity is denoted as $k_{P(2)} = 3 \times 1 = 3$. It can be clearly seen that even the same resource can have different quantities available in different specific periods.

This type of resources introduced above has not attracted great interest in the research of RCPS because they are only used for special circumstances and conditions, and they cannot be applied to the basic features of RCPS (Klein 2000c).

### 3.2.2 Activities

Activity is a technical term used in the field of RCPS, representing a basic, unique task, job or operation in a project. Any projects can normally be broken into a number of activities, denoted as $j \in J$, where $J$ is the set of the total activities in a project. As shown in Table 3.1, the overhaul project can be divided into 14 basic activities.

In a project, each activity $j$ will be processed in a certain time, called the activity duration, denoted as $d_j$, as listed in Table 3.1. In addition, the performance of an activity $j$ over its duration requires or consumes a certain amount of resources, symbolised as $k_{jr}$, $r = 1, 2, \ldots R$, where $R$ represents the number of resources available in a project. In Table 3.1, there are 4 resources available for the overhaul. However, an upper limit on the amounts of resources taken by a current activity is that, its resource requirements cannot exceed the total resources available in scheduling.

### 3.2.3 Precedence Relations

In a project, due to technical requirements, some activities can only be started when certain other activities have been finished. The order in which activities are executed is called precedence relations. There are two ways to represent the precedence relationship for the sequence of activities in a network diagram: *activity-on-arc* and *activity-on-node* networks. Mathematically, the diagram of a project network is often denoted as

$$G = (V, E) \tag{3.1}$$

where G is expressed as a graph of the project network, and $V$ and $E$ denote the set of vertices (nodes) and the set of edges (arcs) respectively. They will be explained below.

To better describe project networks, the terms, "successor" and "predecessor" of an activity need to be introduced here. An activity $j$ can only be allowed to start if one or more certain activities have been completed, and these activities are called a set of immediate predecessor(s) of activity $j$, denoted as $P_j$. Similarly, if an activity $j$ must be completed before certain activities can be started, these activities awaiting the completion of an activity $j$ are referred to as a set of immediate successor(s) of activity $j$, denoted as $S_j$.

#### 3.2.3.1 Activity-on-Arc Network

The *activity-on-arc network* was first suggested by Dimsdale (1963). It is a tool for constructing the precedence relations among activities in a project from left to right, indicating the sequence of activities' executions. An arc represents an activity and a circle called a node signs an event. The activity number of any given task should be always greater than its predecessor(s) and nodes symbolising the events are labelled in ascending letter order.

The example of *activity-on-arc network* shown in Figure 3.1, is the overhaul project with 14 activities. The format, $j(d_j)$ is shown as near an arc, expressing the activity numbers and their durations. For instance, 1(20) just above the arc between the events A and B in Figure 3.1 indicates activity 1 with a duration of 20. Two events, referred to as "head" and "tail" events, define the start and end time points of an activity (arc), as shown in Figure 3.1. The head event, B and the tail event, E represent the start and finish times of activity 4 respectively. A dashed arc is a dummy activity that does not consume any time and resources, as it depicts only precedence relationships with other activities. For example, in Figure 3.1, two dashed lines between events K and N and between events M and N are dummy activities, indicating that activity 14 is a successor of activities 12 and 13.

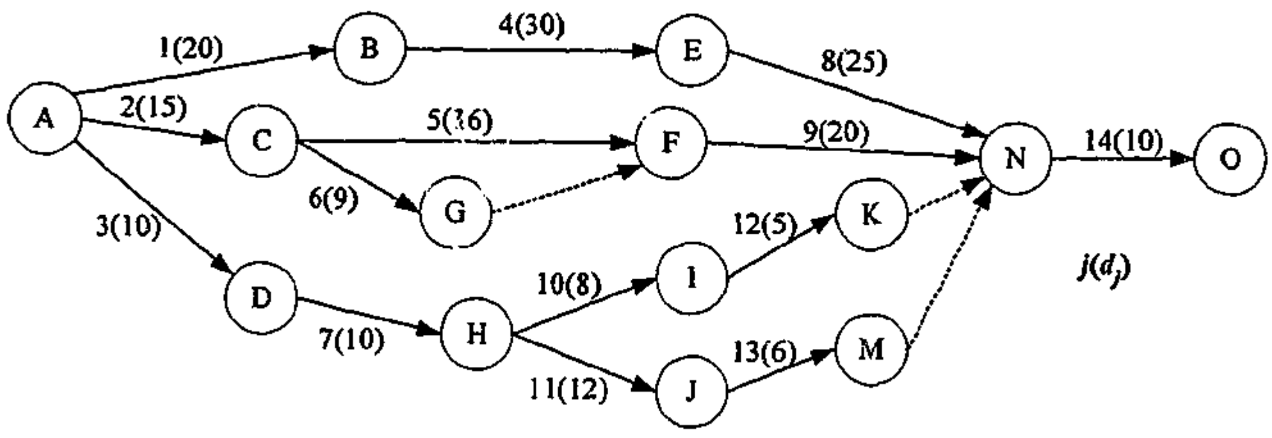


Figure 3.1 Activity-on-arc network of overhaul project

### 3.2.3.2 Activity-on-Node Network

The *activity-on-node network*, introduced by Crandall (1973) and Davis (1975), is a better way of representing precedence relations among activities in a project. In this non-cyclical network, each activity corresponds to a node and precedence relations are expressed by arcs between nodes. Of course, during the construction of the network, it is necessary to ensure that the activity number of a successor is bigger than its predecessor(s).

Figure 3.2 shows an *activity-on-node network* of the overhaul project, in which the data and precedence relations are the same as those given in Figure 3.1. The nodes SP and EP are *source* and *sink* of the network, representing the start and end of a project respectively. The node, $\boxed{j \mid d_j}$ depicts activity $j$ with its duration, $d_j$.

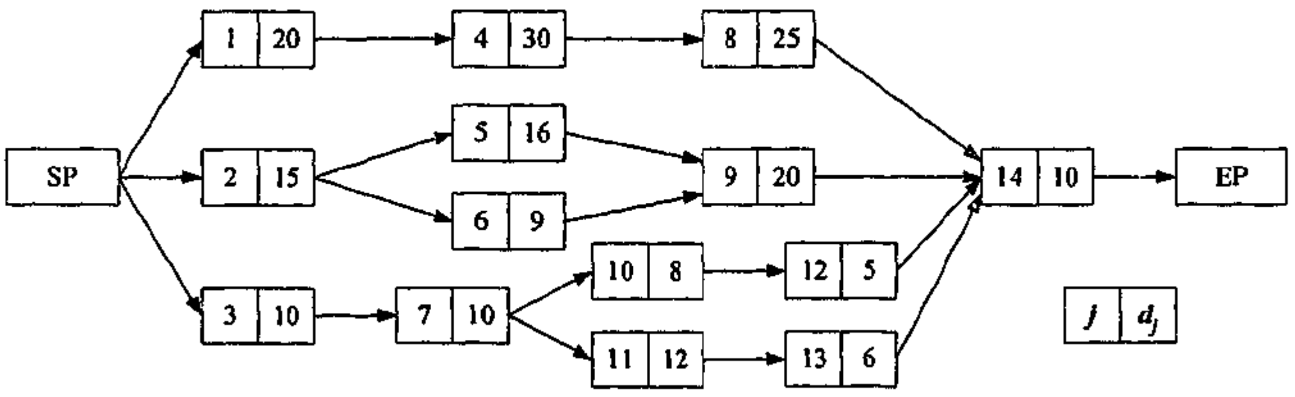


Figure 3.2 Activity-on-node network of overhaul project

Although some researchers have applied the technique of *activity-on-arc network* to the construction of project scheduling problems (Elmaghraby 1977, Syslo 1984; Willis 1985b), this technique has some disadvantages as compared to the *activity-on-node network.* First, the construction of this network is difficult since a number of dummy activities must be considered for the logical meanings of their precedence relations. Second, when the number of dummy activities increases, the representation of this network becomes more complicated and confused (Elmaghraby and Kamburowsky, 1990). Therefore, the *activity-on-node network* is more straightforward since the precedence relations among activities are easily visualised without adding any dummy activities. Most project management software packages opt to use the *activity-on-node* network to represent precedence relations of a project (Klein, 2000c). In subsequent sections and chapters, all the cases of project scheduling problems presented in this thesis will use the *activity-on-node network.*

### 3.2.4 Critical Path Method

Critical Path Method (CPM) is a project network analysis technique used in the environment where there are no resource limitations, to identify the critical path(s) in the project network. The critical path is the longest path in the network and any delays of activities on the critical path will prolong the whole project completion time. This technique has been popularly applied to project scheduling since the 1960's (Kelley, 1961, 1963, Carruthers and Battersby 1966, Thomas 1969).

To determine the critical path(s) in a project, firstly, forward and backward passes will be applied to the project network, to calculate the earliest start and finish times, and the latest start and finish times for each activity, then *the total slack time* (or *total float*) for each activity can be computed. The total slack time of an activity is the amount of time by which the activity can be delayed without affecting the critical path time for a project.

The forward pass determines the earliest start and finish times of each activity under precedence constraint in the project network. The earliest start time ($EST_j$) of an activity *j* is the earliest time at which the activity *j* can be commenced, given that all its predecessors are completed. For example in Figure 3.3, by the forward pass, Activities 1, 2, 3 start at 0 time point since there are no predecessors, and their earliest finish times, $EFT_1$, $EFT_2$ and $EFT_3$ are same as its durations shown in Figure 3.3. Activity 9 has two predecessors, activities 5 and 6, but its two predecessors have different earlier finish times, as shown in Figure 3.3, $EFT_5$ and $EFT_6$ are 31 and 24 respectively so that the successor, activity 9, can be started only when its last predecessor (activity 5) is completed, thus $EST_9$ is 31 and is equal to $EFT_5$. Of course, the earliest finish time ($EFT_9$) of activity 9 will be its start time (31) plus its duration (20) so that $EFT_9$ is 51 (31+20). As demonstrated in the example of Figure 3.3, the earliest start time for each activity is equivalent to the latest of the earliest finish times among the set of earliest finish times of its all predecessors. The earliest start time of the activity will be 0 only if it has no predecessors. The earliest finish time for each activity equals its earliest start time plus its duration. Let the duration of activity *j* denote $d_j$ and a set of predecessors of an activity *j* symbolise $P_j$. Therefore, $EST_j$ and $EFT_j$ can be described mathematically as

$$EST_j = \begin{cases} \max\{EFT_i \mid i \in P_j \neq \phi\} \\ 0, \text{ if } P_j = \phi \end{cases} \quad j=1,2,\dots,J \tag{3.2}$$

$$EFT_j = EST_j + d_j, \quad j=1,2,\dots,J \tag{3.3}$$

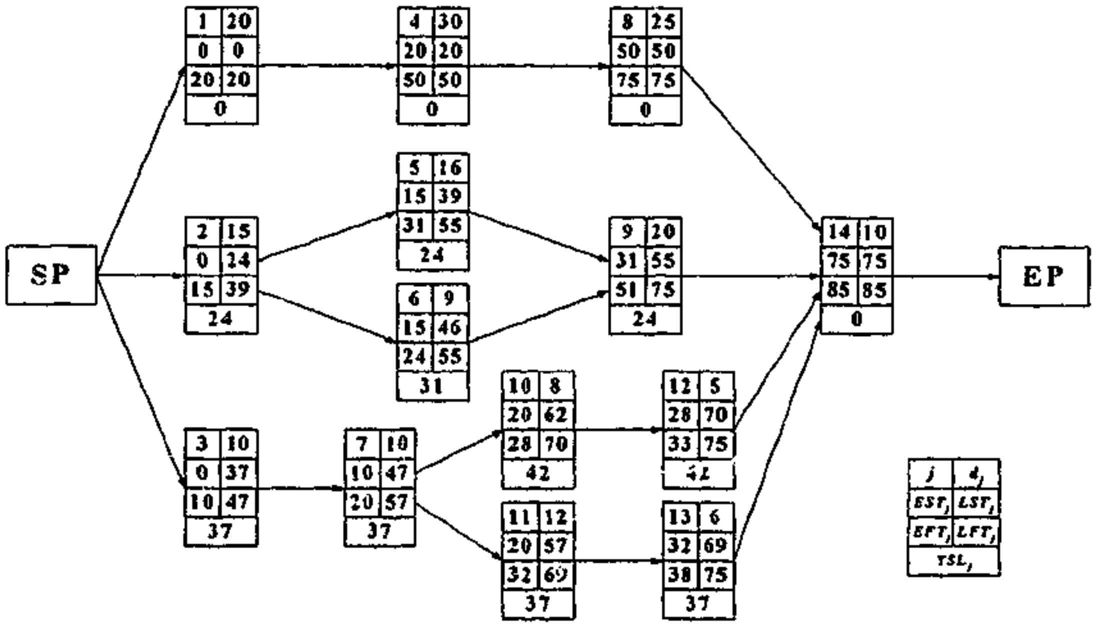


Figure 3.3 Results from critical path method

Through the backward pass, the latest start ($LST_j$) and latest finish times ($LST_j$) of each activity $j$ can be calculated by backward recursion from the last activity to the first one with precedence-feasibility in the project network. In Figure 3.3, the last activity, activity 14, should have no successor(s) and its latest start ($LST_{14}$) and finish times ($LFT_{14}$) will be 75 and 85, the same as its earliest start and finish times. Activity 14 has 4 predecessors, activities 8, 9, 12, 13 in the backward recursion, and the latest finish time of these predecessors are all 75, equal to the earliest start time of activity 14 and their latest start time will be the latest finish time minus its duration, as displayed in Figure 3.3. There are two successors, activities 10 and 11, in activity 7. In this case, the smaller number 57 will be chosen as the latest finish time of activity 7 between two latest start times ($LST_{10}$ = 62, $LST_{11}$ = 57) of its successors, and its latest start time will be its latest finish time subtracted from its duration. Through the example in Figure 3.3, the latest start and finish times for each activity can be expressed by mathematical formula. Let $S_j$ be a set of successors of activity $j$ and the latest start and finish times for each activity in a project are shown as

$$LFT_j = \begin{cases} \min\{LST_i \mid i \in S_j \neq \phi\} \\ EFT_j, \ \text{if } S_j = \phi \end{cases} \qquad j=1,2,\ldots,J \tag{3.4}$$

$$LST_j = LFT_j - d_j \quad j=1,2,\ldots,J \tag{3.5}$$

The *total slack time* (or *total float*) of activity $j$ is a time window within which the activity can experience delays without affecting the whole project completion time. However, those activities with zero total slack time are called *critical activities* or *activities on the critical path*. If any of these critical activities is postponed, the project completion time will be prolonged. The total slack time ($TSL_j$) of activity $j$ is its latest start time minus its earliest start time or is equal to its latest finish time subtracted from its earliest finish time. The total slack time of an activity $j$ is represented as

$$TSL_j = LST_j - EST_j \ \text{ or } \ TSL_j = LFT_j - EFT_j \qquad j = 1, 2, \ldots, J \tag{3.6}$$

By formula (3.6), the total slack time ($TSL_j$) for all activities can be calculated and is shown in the example of Figure 3.3. As noted in Figure 3.3, activities 1,4,8 and

14 have zero total slack time, indicating the path <1-4-8-14> is a critical path in the network. Through the determination of the critical path(s) in the network, more attention can be focused on these critical activities to make sure that none these activities will be delayed.

### 3.2.5 Resource-Constrained Project Scheduling Problem

RCPS problem is a project scheduling problem under the constraints of both limited resources and precedence relations. In such scheduling, the basic performance measure is completion of the project as early as possible. Of course, it becomes a difficult and more complex problem to be tackled when a number of activities compete for several scarce resources.

In general, the RCPS problem can be described as: a project with $J$ activities, labelled as $j$ = 1, 2, ..., $J$. The duration of activity $j$ is denoted as $d_j$, and its start and finish times are symbolised as $ST_j$ and $FT_j$ respectively. There are $K$ resources with $R_k$ $(1 \le k \le K)$ availabilities in a project, and the requirement of a resource for each activity is $r_{jk}$. A pair of activities indicating their precedence relationship, is signified as $(i, j)$, and $H$ denotes a set of pairs of activities in precedence constraints. $A_t$ stands for the set of activities which are still in progress at the time point $t$. In the RCPS, non-preemption of any activities is allowed whilst they are being performed, that is, no activity can be interrupted during its process until completed. Mathematically, the minimization of project completion time of the RCPS problem can be expressed as

$$\min f = FT_J \tag{3.7}$$

subject to

$$FT_j - d_j \ge FT_i \quad \forall (i, j) \in H, \tag{3.8}$$

$$\sum_{i \in S_t} r_{ik} \le R_k \quad t = 1, 2, ..., FT_J; k = 1, 2, ..., K \tag{3.9}$$

The objective function given in (3.7) indicates the minimization of project completion time by means of minimizing the finish time of the last activity $J$. Equation (3.8) represents the precedence constraints indicating that activity $j$ can only be started when all its predecessor(s), $i$, have been completed. Equation (3.9) incorporates the resource constraints in order to satisfy that resource requirements that processed activities do not exceed the resource availabilities at time period $t$.

As noticed above, the RCPS problem can be formulated as a linear program with many constraints. Such a problem becomes increasingly complex as its size increases. There is no algorithm known that is able to find an optimal schedule for any instance of RCPS in polynomial time (Blazewicz 1983). Detailed reviews of RCPS can be found in Özdamar and Ulusoy (1995), Herroelen *et al.* (1998), Herroelen *et al.* (1999), Baptiste and Le Pape (1999), Brucker *et al.* (1999) Neumann and Zimmermann (1999a), and Herroelen *et al.* (2001).

### 3.2.6 Extended Models

The standard RCPS model discussed above is a powerful model, covering most common situations in practice. However, variant models, derived from the standard RCPS model combined with other objective functions, have also been used by other researchers, although they may not be as common as the standard RCPS. This section merely summarises these variant extension models, proposed in the literature because the detailed overview would be beyond my research scope.

#### 3.2.6.1 Preemptive Resource-Constrained Project Scheduling Problem

Preemptive resource-constrained project scheduling (PRCPS) was introduced by Weglarz (1979) and Slowinski (1980). It allows activity $j$ to be interrupted before its termination. This phenomenon is called "preemption of activities". In PRCPS, it is assumed that activities can only be preempted (interrupted) at an integer time point and that these activities are re-continued later on without any additional cost. To achieve the preemption of activities, the duration $d_j$ of an activity $j$ will be split in each basic time unit ($f = 1, 2, \ldots, d_j$). $FT_{j,f}$ denotes the time at which an activity $j$ which is preempted during its duration of $f$ periods. Obviously, $FT_{j,d_j}$ is the finish time of an activity $j$ because activity $j$ is stopped when its duration is just over. Therefore the PRCPS problem can be expressed as

$$\min FT_{J,d_J} \tag{3.10}$$

subject to
$$FT_{i,d_i} \leq FT_{j,0} \quad \forall (i,j) \in H \tag{3.11}$$

$$FT_{j,f} + 1 \leq FT_{j,f} \quad j=1,2,\ldots,J; f=1,2,\ldots,d_j \tag{3.12}$$

$$\sum_{i \in S_t} r_{ik} \leq R_k \quad t=1,2,\ldots,FT_{J,d_J}; k=1,2,\ldots,K \tag{3.13}$$

The objective function is to minimize the project completion time that is the same as the minimization of the finish time of the last activity $J$. Equation (3.11) ensures that the start time of activity $j$ ($FT_{j,0}$) cannot be smaller than the last unit of duration of its predecessor $i$ in order to satisfy precedence relations and Equation (3.12) indicates that the restart time of activity $j$ must be at least one time unit later than its interrupted time. Resource constraints in Equation (3.13) are the same as one in RCPS. Some approaches in PRCPS can be referred to the works of Demeulemeester and Herroelen (1996a) and Bianco *et al.* (1999).

### 3.2.6.2 Time-Constrained Project Scheduling Problem

In some applications, the project deadline $\overline{T}$ is given. However, the resources provided initially may be not enough for the completion of a project by the deadline. To guarantee that the completion of the project meets the deadline, additional quantities of these resources may require to be allocated in certain periods. The objective for the project is to minimize the total additional cost of adding extra resources.

To define this objective function, the cost of an additional unit of resource type $k$ ($1 \leq k \leq K$), symbolised as $u_k$ is predetermined, and the additional amount of resource type $k$ taken in period $t$ ($t = 1, 2, \ldots, \bar{T}$) is denoted as $a_{kt}$. The objective function can be written as

$$\min f = \sum_{k=1}^{K} u_k \times \sum_{t=1}^{\bar{T}} a_{kt} \tag{3.14}$$

Let $x_{tk}$ be 0-1 variable. When resource type $k$ is taken in period $t$, $x_{tk}$ will be 1, otherwise it will be 0. To ensure that excess of initial amounts of resources are not provided, the constraints (3.9) in RCPS model is now replaced by

$$\sum_{t=1}^{\bar{T}} x_{tk} \times \sum_{i \in S_t} r_{ik} - a_{kt} \leq R_k, k=1,2,\ldots,K; t=1,2,\ldots,\bar{T} \tag{3.15}$$

Although this kind of scheduling may be useful in the circumstance defined above, few papers describing its use appear in the literature (Möhring 1984, Yau and Ritchie 1990, Demeulemeester 1995, Kolisch 1995) and it may not be as common as RCPS model in practice. This problem is also an NP-hard problem due to its nature (Deckro and Hebert 1989, Klein 200b).

### 3.2.6.3 Resource Levelling Problem

The resource levelling problem and the time-constrained project scheduling problem differ only in the ways in which objective concerns are met although both approaches aim to schedule so as to minimize the costs of resources in meeting a given project deadline. However, the resource levelling problem attempts to make the usage of various resources as level as possible in the schedule, without violating the project deadline. Within the limits imposed by the project deadline, it may be wise to use resource levelling to "flatten" peak demands on resources because adding extra resources may lead to extra costs in the project.

Levelling usage of resources is equivalent to minimising the variation of resources usage for the problem. Lova *et al.* (2000) proposed the mean coefficient of variation ($E(\delta)$) for setting up the objective function. $x_k$ is the amount of resource $k$ required by the scheduled activities and $K$ is the number of resources ($k = 1, 2, \ldots, K$). The mean coefficient of variation can be defined as

$$E(\delta)=\frac{\sum_{k=1}^{K}\frac{\sqrt{\text{var}(x_k)}}{E(x_k)}}{K} \tag{3.16}$$

There are a number of contributions for the resource levelling problem with various approaches (Woodworth and Willie 1975, Goulter and Ramlogan 1985, Harris 1990, Bandelloni *et al.* 1994, Brinkmann and Neumann 1996, Neumann and Zimmermann 1999b).

#### 3.2.6.4 The Net Present Value Problem

Another type of scheduling problem is concerned with cash flow during the performance of a project. Cash outflows are incurred when activities are executed and cash inflows happen when payments are received on completion of some specific activities, or of an entire project. From the financial point of view, an appropriate schedule needs to be created so that the objective function, the net present value of a project, will be maximized. Russell (1970) first introduced the net present value of cash flow in a project where resource constraints were not considered. Such unconstrained problems have been conducted by Grinold (1972), Bey *et al.* (1981), Smith-Daniels (1986), Elmaghraby and Herroelen (1990), Yang *et al.* (1993), Herroelen and Galens (1993), and Kazaz and Sepil (1996).

Doersch and Patterson (1977) proposed the resource-constrained net present value problem. To define the objective function, let $cf_j$ be cash flow incurred at the end of performing an activity $j$ and $ir$ denote the interest rate. $x_{jt}$ is the 0-1 variable. If an activity $j$ is completed at period $t$, $x_{jt}$ will be 1, and otherwise, $x_{jt}$ will be 0. $EF_j$, $LF_j$ are the earliest and latest finish times of an activity $j$. The objective function may be formulated as

$$\text{Max } f = \sum_{j=1}^{J} \sum_{t=EF_j}^{LF_j} cf_j \times (1+ir)^{-t} \times x_{jt} \tag{3.17}$$

The constraints may be different depending on the properties and natures of various real problems. A number of techniques for exact approaches and heuristics are developed for different conditions in the net present value problem (Russell 1986, Smith-Daniels and Aquilana 1987, Smith-Daniels and Smith-Daniels 1987, Patterson *et al.* 1989, Patterson *et al.* 1990, Yang *et al.* 1993, Ulusoy and Özdamar 1995, Pinder and Marucheck 1996, Icmeli and Erenguc 1996, Baroum and Patterson 1996, Sepil and Ortac 1997, Padman *et al.* 1997, De Reyck and Herroelen 1998). A survey of recent developments in the net present value problem for these models has been conducted by Herroelen *et al.* (1997), Dayanand and Padman (1999), and Baroum and Patterson (1999).

#### 3.2.6.5 Multi-objective Project Scheduling Problem

Sometimes project scheduling requires the consideration of several objectives simultaneously rather than the single objective of the models discussed above. Slowinski (1981) proposes that several conflicting criteria such as project completion time, costs and maximum lateness could be accommodated using linear programming. Nabrzyski and Weglarz (1995) present Tabu Search combined with the knowledge base in solving a multi-objective project scheduling problem. Hapke *et al.* (1998) developed the Pareto Simulated Annealing in the search for "a good compromise" among the several objectives (criteria) in order to obtain Pareto-optimal solutions. Decision support systems are also introduced in the multi-objective project scheduling problem by Davis *et al.* (1992), Rys *et al.* (1994).

### 3.2.6.6 Multi-Project Scheduling Problem

In practice, it may be possible that several dependent projects have to be scheduled in parallel, sharing common resources. Such a circumstance, Pritsker *et al.* (1969) first proposed the multi-project scheduling problem, in which, all projects' networks are represented in one super-network by adding a super-dummy start and end notes, and due dates of activities are also imposed on its single project network.

Different heuristic approaches have been presented by Kurtulus and Davis (1982), Kurtulus and Narula (1985), Kim and Schniederjans (1989), Tsubakitani and Deckro (1990) and Ohmae *et al.* (1992). Speranza and Vercellis (1993) proposed a hierarchical model-based approach to decompose two stages: planning and scheduling decision. Deckro *et al.* (1991) developed an integer decomposition approach to solve the competition for scarce resources in the schedule of the multi-project. Heuristic scheduling combined with 'cost-benefit' scheduling policies was addressed by Lawrence and Morton (1993). Integer programming using the CPM technique and linear programming involving aggregate analysis were proposed by Moccellin (1989), and Kim and Leachman (1993) respectively.

## 3.3 Multi-Mode Resource-Constrained Project Scheduling Problem

The standard RCPS problem assumes that each of the activities making up a project can only be performed in one manner or mode of execution because each activity has only one fixed duration with its corresponding fixed resource requirements in the schedule. Therefore, the standard RCPS model is a typical single mode project scheduling problem. However, in many real word situations, it may be possible for activities to be performed in a number of alternative modes of execution. Elmaghraby (1977) first introduced more realistic RCPS model considering multiple modes out of which one mode might be chosen for execution of an activity. Each mode represents an individual duration with its own corresponding resource requirements.

In the standard Multiple Mode Resource-Constrained Project Scheduling (MMRCPS), once an activity starts in one of its several modes, the activity must finish in that mode, and any mode change or preemption during its process is not allowed. In this model, precedence relations and resource constraints must also not be violated. The scheduling taken in this case is to try to find a start time as well as one execution mode for each activity such that the project completion time can be minimised. This model is more flexible and realistic. It allows several execution options from which one choice is made for an activity. For instance, an activity may be carried out by 3 machines and 4 workers in 2 days or by 2 machines and 3 workers in 3 days or by 1 machines and 2 workers in 5 days. The activity in this case has 3 options (formally called three execution modes) from which to select, but which mode is actually chosen from these 3 execution modes depending on the status of availabilities of resources in that scheduled time.

The MMRCPS model can be described as 0-1 linear programming. Let activity $j$ ($j = 1, 2, \ldots, J$) be performed in $m$ execution mode ($m = 1, 2, \ldots, M_j$) with its duration $d_{jm}$ until it finishes at time period $t$ ($t = 1, 2, \ldots, \overline{T}$) where $\overline{T}$ is an upper bound of the project completion time that can be expressed as

$$\overline{T} = \sum_{j=1}^{J} \min_{m=1}^{M_j} \{d_{jm}\} \tag{3.18}$$

The upper bound of the project completion time defined in Equation (3.18) sums up the smallest durations selected among its modes for all activities. Therefore, the project completion time is impossibly beyond the upper bound $\overline{T}$ and, in the worse-case scenario, the project completion time is same as $\overline{T}$. However, the binary decision variables $x_{jmt}$ can be defined as

$$x_{jmt} = \begin{cases} 1, & \text{if activity } j \text{ is scheduled in mode } m \text{ and finished at time period } t \\ 0, & \text{otherwise} \end{cases}$$

Using above-defined parameters and 0-1 variables, the MMRCPS problem is formulated as

$$\min f = \sum_{m=1}^{M_J} \sum_{t=EFT_J}^{LFT_J} t \times x_{Jmt} \tag{3.19}$$

subject to

$$\sum_{m=1}^{M_j} \sum_{t=EFT_j}^{LFT_j} x_{jmt} = 1 \qquad j = 1,2,\ldots,J \tag{3.20}$$

$$\sum_{m=1}^{M_i} \sum_{t=EFT_i}^{LFT_i} t \times x_{imt} \le \sum_{m=1}^{M_j} \sum_{t=EFT_j}^{LFT_j} (t - d_{jmt}) \times x_{jmt} \qquad j = 1,2,\ldots,J; i \in P_j \tag{3.21}$$

$$\sum_{j=1}^{J} \sum_{m=1}^{M_j} r_{jmk} \times \sum_{q=\max\{t, EFT_j\}}^{\min\{t+d_{jm}-1, LFT_j\}} x_{jmt} \le R_k \qquad k \in R; t = 1,2,\ldots,\overline{T} \tag{3.22}$$

$$\sum_{j=1}^{J} \sum_{m=1}^{M_j} k_{jmk} \times \sum_{t=EFT_j}^{LFT_j} x_{jmt} \le R_k \qquad r \in N \tag{3.23}$$

The objective function (3.19) is to minimise the project completion time. The constraint set (3.20) makes sure that only one mode and one unique finish time are assigned to each activity. Constraints (3.21) take the precedence relations into account. For renewable resources $R$, the level of usage taken by the activities at period $t$ must not exceed the total resource availabilities that are constrained by (3.22). Finally, the constraint set (3.23) limits the total resource consumptions of non-renewable resources, $N$, through the whole project life-span.

MMRCPS is a powerful model that is capable of mapping realistic situations occurring in many projects and has the great benefit of representing different alternatives for performing a certain activity (Li and Willis 1991, Hartmann 1999). At the same time, incorporating different modes into activities enlarges the solution space for search. MMRCPS can be decomposed into two sub-problems: the mode-assignment problem and the standard scheduling problem and, it is NP-hard in the strong sense (Kolisch 1995, Maniezzo and Mingozzi 1999).

## 3.4 Exact Algorithms

In general, exact algorithms developed for combinatorial optimisations in RCPS and MMRCPS can be divided into two categories: linear programming and branch and bound methods. Although exact algorithms may seem to be unable to handle the project scheduling problem while its size increases due to NP-hardness, these algorithms are still very efficient in obtaining optimal solutions for problems with small sizes. In the following section, these exact algorithms will be discussed.

### 3.4.1 Linear Programming

A number of various linear programming formulations have been developed for RCPS and MMRCPS in trying to obtain mathematical optimisation. Use of the mathematical standard software package, can yield an accurate optimal solution for small sized project scheduling problems without requiring any other algorithmic knowledge.

In terms of single mode RCPS, a very early version of 0-1 programming formulation was developed by Pritsker *et al.* (1969). Patterson and Huber (1974) presented 0-1 programming with a "horizon-varying" approach in which, lower-bound is employed which is increased by one each time unit until a feasible solution is found. Patterson and Roth (1976) proposed 0-1 programming combined with implicit enumeration to improve the computational time. Alvarez-Aaldés and Tamarit (1993) developed mixed-integer programming. In this formulation, the resource incompatible set is defined to be a set of activities amongst which a precedence relation exists. The collection of all minimal resource incompatible sets will be determined in order to avoid resources conflicts while these activities may be processed in parallel for competing resources. Mingozzi *et al.* (1998) present a new formulation of 0-1 programming where subsets of activities amongst which no precedence relations are specified in order to satisfy resource constraints whilst these activities may be scheduled in parallel. Carlier and Néron (2000) developed linear programming with some lower bounds that concerned resource capacity.

In the MMRCPS model, Elmaghraby (1977) proposes parametric linear programming to determine the optimal activity durations in achieving the project completion time as soon as possible under resource constraints. Patterson *et al.* (1989) formularise MMRCPS as 0-1 programming. Moder *et al.* (1983) develop non-negative integer programming concerned with cost and duration at the same time. De *et al.* (1997) offer a dynamic programming formulation for multiple execution modes.

For very small-sized project scheduling problems, formulations of linear programming provide a simple and straightforward manner in obtaining optimal solutions, through running mathematical standard software. Unfortunately, such approaches are not at all well suited for computing benchmark solutions since the computational times are much too high. As a consequence, the branch-and-bound type of exact algorithms is more effective and efficient in computational time than linear programming approaches. Branch-and-bound is probably most widely used to develop various exact algorithms for solving project scheduling problems. (Herroelen *et al.* 1998, Demeulemeester and Herroelen 2002). The following will present branch-and-bound schemes.

### 3.4.2 Branch-and-Bound Schemes

The technique of branch-and-bound, applied to project scheduling is a well-known meta-strategy for solving such combinatorial optimization problems as described in Chapter 6 (Klein 2000). Many approaches developed are based on this technique, combined with different schemes. The description of branch-and-bound and commonly existing schemes is explained below.

#### 3.4.2.1 Description of Branch-and-Bound

Branch-and-bound is a technique designed to avoid full enumeration so as to eliminate the computation of unnecessary solutions. Generally speaking, it involves two processes, *branching* and *bounding*.

In the *branching* process, when a project is scheduled, an initial scheduling problem can be divided into several sub-problems and these sub-problems may be continuously further subdivided over and over again. Such a process can be visualized as the creation of a tree with multiple levels. Any sub-problem is represented as a node that corresponds to a partial schedule (a schedule of a subset of activities). As viewed in Figure 3.4, the *root node*, on the first level of the tree, represents the initial schedule problem containing the set of all solutions, and a *branch* is a path from the root node leading to any other node of the tree. However, the branching process will be stopped when only some sub-problems do not need further branches, as its optimal solution is already known, or it can be efficiently computed. As shown in Figure 3.4, no further branches are created from nodes 5, 6, 7, 8, 10, 11, 12 and 13 and they are called unbranched nodes or final nodes. Each final node represents a single solution. In the case of Figure 3.4, Node 0 contains 8 solutions. In the end, all possible final nodes are eventually created so that the optimal solution may be obtained.

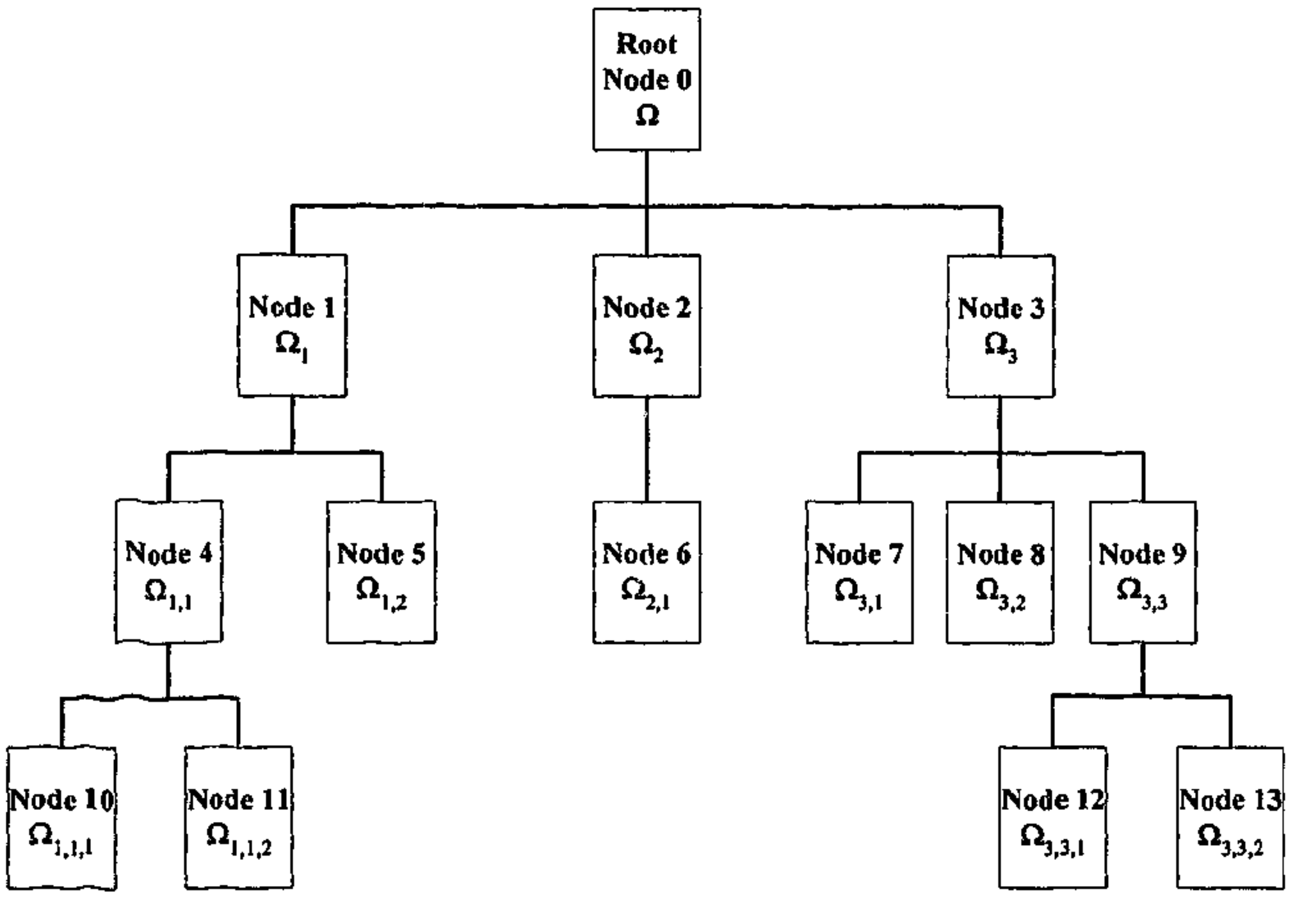


Figure 3.4 The tree structure of branch-and-bound

The *bounding process* provides a possible measure to recognize an optimal solution avoiding all unnecessary enumeration, thus saving the enormous computational time. If a feasible solution obtained at node *i* during the branching process, is smaller than or equal to the lower bound of every final node, this solution will be an optimal one and there is no need for further branching. Through these two processes, the branch-and-bound approach will often find an optimal solution without enumerating all solutions explicitly (Agin 1966, Smith 1984, Johnson 1988).

Most branch-and-bound schemes employ two basic search strategies, *depth-first* (or *backtracking*) or *best-first* (*frontier search* or *skiptracking*), or a combination of these two.

In the *depth-first* search strategy, one of the nodes that has not been further branched is selected to create one new node under that node in the new branch. If this node does not contain an optimal solution or does not contain a solution that is less than a solution from any preceding branches, the branching process from that node is stopped and then going up along that node to a branch that has not been fully explored, and then from that branch further down, in order to investigate whether any better or optimal solutions exist.

The *best-first* search strategy is to select the node that has not been further branched, but that has the lowest or best lower bound. However, this search strategy requires great memory storage but an optimal solution is always obtained in the end of search.

### 3.4.2.2 Precedence Tree

The *precedence tree* approach is based on early-start schedule at each level of the branch-and-bound tree and attempts to schedule the eligible activities from a set of the activities whose predecessors have been finished as soon as possible, without the violation of resource constraints. For MMRCPS, an eligible activity is selected, followed by the feasible mode assignment. Then the earliest predecessor is found so as to enable determination of the next scheduled time at the next level of branch, until the last activity is reached during branching. However, the *depth-first* search strategy is commonly applied in the *precedence tree* approach.

The basic idea of this approach was originally from the technique developed for effectively solving integer programming for RCPS proposed by Talbot and Patterson (1978), in which the enumeration process is to attempt scheduling the activities with their earliest precedence and resource-feasibility from the list of eligible activities at each level of the enumeration tree. Patterson *et al.* (1989) employed this idea, developed an algorithm called *precedence tree* procedure. Sprecher (1994) and Sprecher and Drexl (1998) restructured the version of Patterson *et al.* (1989) and proposed a straightforward approach with some bounding criteria for solving MMRCPS.

### 3.4.2.3 Extensive Alternatives vs Mode and Extension Alternatives

Stinson *et al.* (1978) introduced extensive alternatives that are sets of eligible activities whose predecessors have been completed so that resource requirements of a set of activities do not exceed the current resource availabilities. In their approach, through the enumeration of all extensive alternatives at each level of the branch-and-bound tree, the optimal solution can ultimately be found. Mingozzi *et al.* (1998) proposed an approach that is slightly different from Stinson *et al.* (1978). In their approach, new lower bounds will be obtained by relaxing some constraints, such as feasible subsets of activities and resource limitation, in the process of creating the brand-and-bound tree.

The approaches based on extensive alternatives solve only single-mode RCPS problems. To deal with MMRCPS problems, Hartmann and Drexl (1998) adopted the concept of mode alternatives from Sprecher *et al.* (1997) combining it with the idea of extensive alternatives proposed by Stinson *et al.* (1978). The approach based on such a combination is called *mode and extension alternative*. In this approach, sets of eligible activities will be confirmed when a decision point is determined at a level of the branch-and-bound tree. Then the set of mode alternatives will assign the modes to the sets of eligible activities so that an extensive alternative is selected before branching to the next level.

#### 3.4.2.4 Delay Alternatives vs Mode and Delay Alternatives

The concept of delay alternatives was originally proposed by Christofides *et al.* (1987) for the examination of lower bounds in linear programming for solving RCPS problem. Demeulemeester and Herroelen (1992) enhanced this concept and proposed their approach to the branch-and-bound tree where minimal delay alternatives are taken into account when renewable resources conflict. In their approach, some activities in the eligible set may be removed and not be added in the active set at a decision point of level *g* in order to minimize the partial schedule delay when resources are not enough for all of these activities.

For MMRCPS, Sprencher *et al* (1997) proposed the idea of mode and delay alternatives, adopting the approach of Demeulemeester and Herroelen (1992) for the single-mode RCPS in combination with the notion of a mode alternative. All eligible activities at the decision point have already been assigned a mode. A mode alternative is a mapping which is assigned to each activity and a mode for eligible activities is then fixed before being scheduled.

#### 3.4.2.5 Other Exact Approaches for Single and Multiple Modes

In the above discussion, some exact approaches have not been mentioned, such as *forbidden set* and *schedule schemes* for single mode and *dominance rules* for multiple modes, These will be briefly addressed below. However, a few other approaches such as *float splitting*, *immediate selection* are not briefly addressed here as they exceed the scope of this review.

Igelmund and Radermacher (1983a, b) introduced the notion of a *forbidden set* into their approach in which, a *forbidden set* is defined as a set of activities that can be scheduled simultaneously because there do not exist precedence relations among the activities in that set. However, the approach considers only the minimal forbidden sets and only these sets will be enumerated in the brand-and-bound tree.

For *schedule schemes*, there are a few variations. In the version of Brucker *et al.* (1998), each node of the enumeration tree represents a set of feasible schedules and any pair of activities $(i, j)$ in the feasible schedule will hold only one out of the three relations: either $i$ precedes $j$, or $j$ precedes $i$, or both $i$ and $j$ are performed concurrently for at least one time unit. Therefore, the form of the schedule scheme associated with a node of the branch-and-bound tree will be such above mentioned relations. In the binary version of the *schedule schemes* approach (Demeulemeester and Herroelen 2002), two possibilities are considered at each level of the branch-and-bound tree. One branch has two activities that are scheduled simultaneously for at least one time unit. However, the other branch combines the possibilities in which, activity $i$ precedes activity $j$ or activity $j$ precedes activity $i$.

There are a number of *dominance rules* that are based on bounding criteria in order to speed up the enumeration procedures in branch-and bound tree. *Time windows* as a bounding criterion was proposed by Sprecher (1994) and Sprecher *et al.* (1997), in which, the latest finish time of each activity will be computed when an upper bound of the project completion time and the modes of shortest duration are given. Then the non-delayability rule will examine its precedence tree. A *preprocessing* idea was suggested by Sprecher *et al.* (1997). In their approach, two bounding rules, *data reduction* and *nonrenewable resource* rules are implemented by preprocessing so as to improve the schedule of MMRCPS. Sprecher *et al.* (1997) defined a *multi-mode left-shift rule* as a reduction of an activity finish time without changing the modes and finish times of other activities, resulting in a feasible schedule. Hartmann (1999) proposed *an order swap rule,* in which the start or finish times of two activities $i$ and $j$ are interchanged or swapped without the violation of precedence and resource constraints. If that scenario can be performed, the current partial schedule will be dominated. Sprecher and Drexl (1998) defined *a cutset rule* for MMRCPS. A cutset, *CS* is a subset of activities scheduled in partial schedule, *PS*. If $CS(PS_{cur}) = CS(PS_{prev})$, the start time of activity *j*, $ST_j \geq$ maximum partial schedule time, $f^{max}(PS_{prev})$ and at the same time, nonrenewable resources are available, then the current partial schedule, $PS_{cur}$ will be dominated. The *immediate selection rule* was originally developed by Demeulemeester and Herroelen (1992) for the single-mode case and was late adapted by Sprecher *et al.* (1997) for MMRCPS, in which there is an eligible activity *j* with fixed mode $m_j$ that cannot be simultaneously processed with any unscheduled activity in its any mode, then activity *j* is the only eligible activity that needs to be selected for scheduling. Therefore, one branching alternative is examined instead of testing all. To avoid the enumeration of duplicate schedules, Sprecher (1994) proposed the *enumeration rule*, in which if two activities *i* and *j* have been scheduled at the previous and current level of the tree and result in the same start times for the both activities, then the current partial schedule will be accepted in the enumeration.

## 3.5 Heuristics

In practice, the majority of real project sizes range from medium to large. Such sizes make the scheduling problem more complex mathematically when many constraints are imposed. These are typical NP-hard problems. From a theoretical point of view, any exact approaches are unable to handle such computationally complex problems.

In reality, many project scheduling problems may not require accurate solutions in terms of optimization because of the unpredictable events that may occur in the process of the project. Heuristic approaches, based on priority rules, always provide feasible solutions in a simpler and faster manner although they may not guarantee an optimal solution. However, a set of priority rules may often be applied to RCPS problem in heuristics in order to select a better solution among all the results.

In the following section, *serial* and *parallel* schedule schemes will be explained first. They are the core of heuristic procedures for RCPS. This discussion will be followed by the introduction of a number of priority rules for the determination of activity priorities in scheduling, then *forward* and *backward* scheduling are briefly addressed and, finally, common priority rule-based heuristics are described.

## 3.5.1 Scheduling Schemes

*Serial* and *parallel* scheduling schemes are commonly used in priority rule based on heuristics for RCPS. They are the core of heuristic procedures in project scheduling. Both schemes generate a feasible solution by processing a partial schedule in a stage-wise manner by employing priority rules to select one or more activities from the set of eligible activities for scheduling.

To clarify these two schemes, some definitions and symbols used in the description of these schemes are explained here. Partial schedule, $PS_n$ means that only a number of activities, and not all activities in a project, have been scheduled at some stage $n$. The remaining set, $R_n$ is for those activities that are ineligible as candidates for scheduling at stage $n$. The decision set, $D_n$ keeps only the activities that can be considered for scheduling at stage $n$. The active set, $A_n$ is used to place the activities that have been scheduled but have not been completed at stage $n$. However, the complete set, $C_n$ stores the activities that have been or are just completed at stage $n$. $R$, $\pi R_r$ and $K_r$ denote a set of resource types, the current and total resource availability of a resource type $r$ respectively. $v(j)$ signs the priority value of activity $j$.

### 3.5.1.1 Serial Scheduling Scheme

The *serial scheduling scheme* was proposed by Kelly (1963). This scheme places only one activity into the partial schedule at each stage until all activities in a project are scheduled to obtain a feasible complete schedule. Therefore, a complete schedule in this scheme will be constructed through $J$ stages if a project contains $J$ activities $(1, 2, ..., J)$.

The procedure of the *serial scheduling scheme* can be described as shown in Figure 3.5. At the beginning of the scheduling, the partial schedule is null, and the remaining set, $R_n$ holds a set of all activities in a project, except for those activities without predecessors. Before scheduling, the decision set, $D_n$ initially contains only those activities that have no predecessors, and the current resource availabilities, $\pi R_r$ should be equal to the total resource availabilities for all types.

In this scheme, $J$ stages are processed. In each stage, only one activity is selected from the decision set, $D_n$ and however, which activity is selected depending on the priority value of a certain priority rule. When an activity is determined, the start, $ST_j$ and finish, $FT_j$ times of activity $j$ can be computed. This activity will be placed in the partial schedule, $PS_n$ at stage $n$. If any activities are completed from the active set, $A_n$ at stage $n$, these activities can be put in the complete set, $C_n$. Before moving to the next stage, information such as the active set, $A_n$, decision set, $D_n$ and remaining set, $R_n$, requires to be updated.

```
Initialisation: $PS_n := \phi; C_n := \phi; A_n := \phi; R_n := \{J\} \setminus \{j \mid P_j \in \phi\};$
                $D_n := \{j \mid j \in J, P_j \subseteq \phi\}; \pi K_r := K_r; r \in R;$
Begin
  For $n = 1$ to $J$
    Select $j^* := \underset{j \in D_n}{\text{extremum}}\{v(j)\};$
    Compute the start time $ST_{j^*}$;
    $FT_{j^*} = ST_{j^*} + d_{j^*};$
    $PS_n := PS_{n-1} \cup j^*;$
    If $j \in A_n$ is completed
      $C_n = C_{n-1} \cup j;$
    End If
    Update $A_n$, $D_n$, $R_n$, $\pi K_r$
  Next $n$
End
```

Figure 3.5 Serial scheduling scheme

### 3.5.1.2 Parallel Scheduling Scheme

The *parallel scheduling scheme* was proposed by Brooks and White (1965). In this scheme, one or more eligible activities will be scheduled at each decision time point. The decision time point is that time point which is the earliest finish time of an activity among these already scheduled activities. This scheme should have at most $J$ stages if there are $J$ activities (1, 2,..., $J$).

To clarify the scheme, the detail of such a schedule is addressed here through Figure 3.6. At the beginning of the scheduling scheme, decision time point, $t_n$, is zero, and the partial schedule, $PS_n$, active set, $A_n$ and complete set, $C_n$ are all null sets. The decision set, $D_n$ initially holds activities that have no predecessors. However, the remaining set, $R_n$ contains all activities except for those without predecessors at the beginning of scheduling. Of course, the current resource availability, $\pi K_r$ for any resource, $r$ should be as same as the total resource availability, $K_r$ at beginning. During scheduling, one decision time point, $t_n$ will be determined first at each stage $n$, and equals the earliest finish time of the activity or activities in the active set, $A_{n-1}$ at stage $n$-1. Then the active set, $A_n$ is temporally updated with the removal of the activity or activities that finish at $t_n$ at stage $n$, and these activities will be shifted from the active set to the complete set, $C_n$ at stage $n$. At the same time, the resource availabilities, $\pi K_r$ for every resource, $r$ is also updated. Once the decision time point, $t_n$ is determined at stage $n$, a number of activities, based on their priority values, will be selected from the decision set, $D_n$ one by one, if resources are available to them. As shown in Figure 3.6, if the loop condition satisfies that the decision set, $D_n$ contains some eligible activities as well as the current resource availabilities, $\pi K_r$ ($r \in R$) are available to the activity $j^*$ that will be selected, the start time, $ST_{j^*}$ of that activity $j^*$ is equal to the decision time point $t_n$ and the finish time $FT_{j^*}$ is the start time of activity $j^*$ plus its duration. Then resource availabilities are updated and the activity $j^*$ is put in the active set, $A_n$ and partial schedule, $PS_n$ respectively. If the loop condition is still satisfied, the 2nd highest priority of activity $j^*$ will be chosen following the same procedure of the calculation of the start and finish times of activity $j^*$ and so on until the loop condition is unsatisfied. Then the partial schedule goes to the next stage. The complete schedule will be reached if all activities are in the active or complete sets.

```
Initialisation :$t_n := 0; PS_n = A_n = C_n := \phi; D_n := \{j \mid j \in J, P_j \in \phi\};$
            $R_n := \{J\} \setminus \{j \mid P_j \subseteq \phi\}; \pi K_r := K_r; r \in R$
Begin
  For $n = 1$ to $N \leq J$
    $t_n = \min_{j \in A_{n-1}} \{FT_j\}$
    $A_n = A_{n-1} \setminus \{j \mid j \in A_{n-1}, FT_j = t_n\}$
    $C_n = C_{n-1} \cup \{j \mid j \in A_{n-1}, FT_j = t_n\}$
    Update $\pi K_r \; \forall r \in R, D_n, R_n$
    Do While $D_n = \phi$ and $\pi K_r \geq R_{j^* r} \; \forall r \in R$
      $j^* := \text{extremum}_{j \in D_n} \{v(j)\}$
      $ST_{j^*} = t_n$
      $FT_{j^*} = ST_{j^*} + d_{j^*}$
      Update $\pi K_r \; \forall r \in R$
      $A_n := A_n \cup j^*$
      $PS_n := PS_n \cup j^*$
    Loop
  Next $n$
End
```

Figure 3.6 Parallel scheduling scheme

The two scheduling schemes do not dominate each other, because one scheme cannot always yields a better solution than the other (Cooper 1977 and Kolisch 1996b). However, the serial scheduling scheme may seem slightly worse than the parallel scheduling scheme in terms of the average deviation from optimal solutions (Alvarez-Valdes and Tamarit 1989, Klein, 2000c).

### 3.5.2 Priority Rules

Varieties priority rules have been proposed over many years. These rules are used to rank the activities in the decision set so that the selection of activities is based on the priority sequence for any scheduling schemes. More precisely, a priority rule is a mapping that assigns each activity $j$ in the decision set a priority value, $v(j)$, along with such information to determine the sequencing selection among these activities in the decision set for scheduling. These rules have been studied and investigated over the last three decades (Davis and Patterson 1975, Cooper 1976, 1977, Alvarez-Valdés and Tamarit 1989, Boctor 1990, Kolisch 1996a, Thomas and Salhi 1997). These rules can be classified into six main categories based on the information employed to calculate the priority value $v(j)$ (Alvarez-Valdés and Tamarit 1989, Kolisch 1995).

- *Activity itself based priority rules*
- *Network based priority rules*
- *Critical path based priority rules*
- *Resource based priority rules*
- *Composite priority rules*
- *Random based priority rules*

In the following, some important and frequently used priority rules in each category will be introduced. Some rules are adopted from job shop scheduling. Some come from PERT/CPM methods and others are particularly designed for RCPS.

### 3.5.2.1 Activity Based Rules

In this category, the information considered for priority rules is only related to the activity itself and does not consider the activity's relations to the project.

a. Shortest Processing Time (SPT)

   $\text{Min } d_j, \forall j \in D_n$

This priority rule originally comes from machine scheduling. The idea is to prioritise the activities that have the shortest duration so that the activities with the longer duration are never waiting on short ones.

b. Longest Processing Time (LPT)

$\text{Max}\, d_j,\ \forall j \in D_n$

The primary purpose is to start the activities with the longest processing time first so as to try to avoid the overall scheduling delay.

### 3.5.2.2 Network Based Rules

These rules rank priorities of activities based on the network structure of a project. They are based on the precedence relationship between activities. Before introducing these rules, first let *H* denote the set of precedence constraints which can be represented by pairs of activities (*i,j*)

a. Most Immediate Successors (MIS)

$\text{Max}\,|S_j|,\ \forall j \in D_n,\ S_j = \{i \mid (i,j) \in H\}$

This rule specifies that the activity that has most successors waiting on scheduling will be started first so that its successors have a chance to start scheduling.

b. Most Total Successors (MTS)

$\text{Max}\,|S_j^*|,\ \forall j \in D_n, S_j^* = \{i \mid i \in \text{the activities on a path from } j \text{ to its all successors}\}$

Where $S_j^*$ denotes all successors of activity *j*. This rule does not only count the immediate successors, but all the successors of activity *j*. This rule prioritises the activity with the most successors so as to prevent subsequence delay to all its successors.

c. Great Rank Positional Weight (GRPW)

$\text{Max}\,(d_j + \sum_{i \in S_j} d_i),\ \forall j \in D_n$

This rule originally comes from job shop scheduling. It tries to select the activity that has great total durations of the activity *j* plus all its immediate successors because it will take longer to finish these activities in scheduling.

d. Great Rank Positional Weight* (GRPW*)

$\text{Max}\,(d_j + \sum_{i \in S_j^*} d_i),\ \forall j \in D_n$

This rule differs from rule c that is **GRPW*** sums up the duration of activity *j* as well as the durations of its total successors.

### 3.5.2.3 Critical Path Based Rules

This category focuses on the critical path in terms of producing priority rules. These rules are based on results of the forward and backward passes by calculating the forward and backward critical path.

a. Earliest Start Time (EST)

$\text{Min}\,EST_j,\ \forall j \in D_n$

This well-known rule is often applied to scheduling problems. Based on this rule, the activity that has the earliest start time, among others in decision set, will be scheduled first due to the concern about the critical path.

b. Earliest Finish Time (EFT)

$\text{Min}\,EFT_j = \text{Min}\,(EST_j + d_j),\ \forall j \in D_n$

This rule attempts to let the activity that will be finished earlier be scheduled first so as to get these activities out of way earlier.

c. Latest Start Time (LST)

$\text{Min}\,LST_j,\ \forall j \in D_n$

This measure gives the higher priority to the activity that has the earliest late start time first because scheduling this activity after its LST means a delay with respect to the completion time in the critical path.

d. **Latest Finish Time (LFT)**

$\text{Min}\, LFT_j = \text{Min}\,(LST_j + d_j),\ \forall j \in D_n$

This rule is adopted from job shop scheduling in response to the earliest due date. This rule tries to schedule activities before their due date in the schedule.

e. **Minimum Slack (MSLK)**

$Min\,(LST_j - EST_j),\ \forall j \in D_n$

This is the most popular rule concerning the minimum slack time in the critical path. When the activity is delayed, its slack time is reduced by the amount of delay. To reduce the delay of the whole schedule, the critical activity with smallest slack time should be scheduled earlier.

#### 3.5.2.4 Resource Based Rules

This category of priority rules is concerned with the resource requirements of different activities. Scheduling different orders of activities yields significantly different resource arrangements. Constraint of total resource capacities greatly affects the project completion time. Because of such concerns, this type of priority rules focuses on the resource issue.

Greatest Resource Demand (**GRD**)

$\text{Max}\, d_j \sum_{r \in R} k_{jr},\ \forall j \in D_n$

This rule assigns priority to the eligible activities on the basis of the product of the duration of activity $j$ and the sum of total resource requirements of that activity. This allocates higher priorities to potential resource-bottleneck activities based on their greater resource demands.

#### 3.5.2.5 Composite Priority Rules

To overcome the disadvantage of only relying on a single type of information, this category of priority rules combines the information from the above categories. These priority rules are obtained by computing a weighted sum of individual priority values in the other categories.

a. Weighted Resource Utilisation Ratio and Precedence (WRUP)

$$\text{Max}\,(w_1 \times |S_j| + w_2 \times \sum_{r\in R} \frac{k_{jr}}{K_r}),\ \forall j \in D_n$$

This rule concerns itself with the number of successors of activity $j$ as well as the average resource usage over all resource types for activity $j$. Therefore, this rule takes into account the network information as well as the resource information. The weights are given by the user, to the 1st and 2nd term in the above formula to emphasize which one is of greater concern - the number of its successors or resource usage.

b. Improved Resource Scheduling Method (IRSM)

$$\min(\max\{0, E_{(j,i)} - LST_i \mid (i,j) \in AP_n\}),\ \forall AP_n = \{(i,j) \mid i,j \in D_n, i \neq j\}$$

Where $E_{(j,i)}$ is the earliest time to schedule activity $i$, if activity $j$ is started at the decision time point $t_n$ for any activity pair $(i,j)$ in the decision set, $D_n$. $AP_n$ is the set of all activity pairs in the decision set at stage $n$. The idea of this rule is to try to avoid delay in all other activities in the decision set for scheduling when activity $j$ is scheduled at the decision time point $t_n$ at stage $n$.

c. Worst Case Slack (WCS)

$$\min(LST_j - \max\{E_{(i,j)} \mid (i,j) \in AP_n\},\ \forall AP_n = \{(i,j) \mid i,j \in D_n, i \neq j\}$$

By this rule, all activity pairs in the decision set are examined to calculate how much slack is inherent in each activity, and then the activity with the smallest worst case slack will be selected for scheduling.

### 3.5.2.6 Random Based Priority Rules

Applying any of the above priority rules, sometimes several activities may have the same priority values. In this case, an extra rule has to be used as a tie-breaker, to further determine the priority sequence of these activities. This makes the determination of activity priorities more complex.

**Random** priority rule, RNDPR avoids the priority conflict where several activities may have the same priority value as any of the above introduced priority rules is employed. This rule is the way of giving all different priorities to the activities randomly, so that there are no same values in some activities. This rule is simpler and easier than other rules in the determination of the priorities of activities, saving unnecessary computation time (Pan *et al.* 2001b, Pan *et al.* 2001c).

### 3.5.3 Forward and Backward Scheduling

#### 3.5.3.1 Forward Scheduling

The scheduling schemes presented in sections 3.5.1.1 and 3.5.1.2 are scheduled in the forward direction. They are examples of forward scheduling. More precisely, when the activities of a project are presented in the way of the project network, a schedule which starts with the activities that have no predecessors, will be selected at the time 0, based on priorities. When the scheduling time gradually increases, other successors may be eligible to start after their predecessors are completed, and then the forward scheduling will terminate when the last activity is reached.

#### 3.5.3.2 Backward Scheduling

In contradistinction to the forward scheduling, backward scheduling schedules activities in the reverse direction along with the project network, that is, a schedule starts with the last of the activities that have no successors, and gradually scheduling activities from backwardness to forwardness until the first of the activities that have no predecessors are reached. However, the result of the start time for each activity gained through backward scheduling, is a dummy start time. The real start times of activities need to be adjusted by left shift, that is the start time of each activity should be the project completion time obtained by backward scheduling, minus its dummy start time.

### 3.5.4 Proposed Priority Rule-Based Heuristics

Priority rule-based heuristics are made up of at least two components, a scheduling scheme and a priority rule. However, a number of different priority-rule based heuristics can be yielded by one or more priority rules combined with one or both scheduling schemes and different direction passes. Such priority-rule based heuristics have been widely applied for solving RCPS problems because of the reasonable computation time and easy implementation.

#### 3.5.4.1 Single Pass Methods

Any of single pass methods employs one scheduling scheme, *serial or parallel scheduling* with only one particular priority rule to obtain one feasible schedule. This is the one of oldest priority rule-based heuristics. The single pass methods require very little time, even for a very large project. However, they do not often get good results in terms of the minimization of project completion time. There are a number of examples of such heuristics applied to RCPS in literature (Davies 1973, Elsayed 1982, Kolisch 1996a, Patterson 1976, Thesen 1976, Whitehouse and Brown 1979, Comer *et al.* 1997).

#### 3.5.4.2 Multi-Pass Methods

As mentioned in the section 3.5.4.1, any of the single pass methods yields only one feasible schedule that may be the worst solution. However, different scheduling schemes combined with different priority rules in various ways may result in various schedules, from amongst which a better solution may be obtained. Such heuristics are called multi-pass methods. The most common multi-pass methods, *multi-priority rule* *methods*, *forward-backward scheduling methods* and *sampling methods* will be briefly described below.

#### 3.5.4.2.1 Multi-priority Rule Methods

Multi-priority rule methods employ any scheduling scheme(s) with different priority rules to produce a number of different schedules so as to improve the quality of a solution. There are several approaches in such methods. The basic approach is to use one scheduling scheme in conjunction with only one priority rule at a time so that *m* individual priority rules will yield *m* different schedules. The second approach applies a scheduling scheme with a combination of various priority rules at the same time, producing many more schedules. Yet another approach relies on using both scheduling scheme with different priority rules at a time. Multi-priority rule methods are proposed by Patterson (1973), Elsayed (1982), Elsayed and Nasr (1986) Ulusoy, G. and L. Özdamar (1989), Boctor (1990), Khattab and Choobineh (1990, 1991).

#### 3.5.4.2.2 Forward-Backward Scheduling Methods

Forward-backward scheduling methods apply a scheduling scheme with a priority rule by scheduling activities in forward and backward directions. Various versions of such methods are proposed by Li and Willis (1992) with the serial scheduling scheme, Özdamar and Ulusoy (1996a, 1996b) with the parallel scheduling scheme and Lova *et al.* (2000) with the parallel scheduling scheme for multiproject scheduling.

#### 3.5.4.2.3 Sampling Methods

Sampling methods usually use one scheduling scheme and one priority rule at a time. However, different schedules are obtained by biasing the selection of the priority rule through a random mechanism. There are two major approaches, *biased random sampling* and *regret based biased random sampling*, usually distinguished by a concern with the way of determining the selection probability of an activity.

Biased random sampling usually employs one priority rule, based on which, the probability of the selection of an activity is calculated. Let $v(j)$ denote the priority value of activity $j$ and $p(j)$ is the probability of choosing activity $j$ for scheduling. If the bigger value obtained from a priority rule is considered a higher priority, the formula of probability of the selection of activity $j$ can be represented as

$$p(j) = \frac{v(j)}{\sum_{i \in D_n} v(i)} \tag{3.24}$$

Conversely, if the smaller value is regarded the higher priority of activity, the formula of the probability of the selection of activity $j$ should be written as

$$p(j) = \frac{1}{(v(j) \times \sum_{i \in D_n} 1/v_i)} \tag{3.25}$$

Biased random sampling methods have been presented by Wiest (1967), Alvarez-Valdés and Tamarit (1989), Cooper (1976).

Regret based biased random sampling calculates the probability of the selection of activities indirectly using regret values. The regret value of activity $j$ is the absolute difference between its priority and the worse priority in the decision set. If the higher value obtained from a priority rule is the higher priority of activity $j$, its regret value is calculated as

$$r(j) = v(j) - \min_{i \in D_n} v(i) \tag{3.26}$$

Conversely, if the smaller value gained from a priority rule is considered the higher priority of an activity, the regret value of activity $j$ is computed as

$$r(j) = \max_{i \in D_n} v(i) - v(j) \tag{3.27}$$

In the calculation of the probability of the selection of activity $j$, the constant $\varepsilon > 0$ is added in the both numerator and denominator to assure that the selection probability for each activity in the decision set is greater than zero. The parameter $\alpha$ is also used in the formula of the selection probability of activity $j$ to determine the level of bias. A higher value of $\alpha$ will cause no bias, while a smaller $\alpha$ will introduce much bias in random activity selection. The probability of the selection of activity $j$ can be represented as

$$p(j) = \frac{[r(j) + \varepsilon]^{\alpha}}{\sum_{i \in D_n} [r(i) + \varepsilon]^{\alpha}} \qquad (3.28)$$

The application of and experiments on regret based biased random sampling methods for RCPS can be found in Drexl (1991), Kolisch (1995), Kolisch and Drexl (1996), Kolisch (1996b).

## 3.6 Decision Support Systems

Decision support systems (DSS) are a specific class of computerised information system that may include an expert system or artificial intelligence (AI) intended to help the decision maker (DM) solve realistic project scheduling problems. Recent developments in DSS for project scheduling can be grouped into two categories: the *primary DSS* and *DSS built with AI.*

Anthonisse *et al.* (1988) first suggested the use of the concept of DSS in RCPS problems. In the primary stage, a number of problem models are defined and criteria are set up so as to enable the DM to fit the real problem into a suitable model for scheduling. Zhang (1998) proposed the image construction method for the DSS so as to visualise all important data, enabling the DM easily to manipulate these data to obtain a better result in project scheduling. Tomasz, *et al.* (1994) presented a DDS of MIPS prototype version that is a graphic-oriented interactive system for solving multiobjective project scheduling where the DM will obtain an approximate schedule, based on heuristics selected from the list offered by the system. However, the schedule may be further improved by shifting some activities around according to the real situation and the DM's knowledge. Nowicki and Smutnicki (1994) developed a

DSS, in which the DM makes decisions in the three stages: (1) whether the activity risks violation of feasibility; (2) whether to let the activity be placed in the decision set, and (3) lastly to select a priority rule in the system. Norbis and Smith (1996) developed an interactive DSS where the DM is able to change the relative priorities of activities as well as priority rules in the interactive scheme in order to get a better schedule.

Ulusoy and Özdamar (1996) proposed a framework of an interactive scheduling system. In this system, several objectives (criteria) such as the project completion time and net present value are available for the DM to choose, depending on the requirements of a real situation. In the system, the scheduling mechanism primarily determines the start times of activities and specific operating modes. The final solution will be obtained by the DM testing the primary schedule using the intelligent constraint based analysis through a "what-if" routine. Arinze and Partovi (1992) developed an expertise-based DSS built with the CPM technique and it permitted the determination of a good schedule under different circumstances, in terms of the minimisation of the project completion time. Nabrzyski and Weglarz (1995, 1999) presented an expert rule-based approach cooperating with a tabu search algorithm for the case of multiobjective project scheduling problems. Schirmer (2000) developed an integrated case-based reasoning DSS in the benefit of selecting better algorithms for a real RCPS problem by analysing several criteria on the algorithmic performance.

## 3.7 Metaheuristics

This section gives the overview of metaheuristics proposed for RCPS. Metaheuristics have been widely used since 1980s, particularly for solving hard combinatorial optimisation problems. RCPS certainly falls into this category.

Metaheuristics are a class of approximate methods that is not intended to explore the whole solution space. This class of heuristics may be concisely described as "walk through neighbourhoods", a search trajectory through the solution domain of a problem.

In the following subsections, a number of the main metaheuristics proposed for RCPS problems, *genetic algorithms* (GA), *simulated annealing* (SA), *tabu search* (TS), *neural networks* and *ant colony optimization* (ACO), will be briefly described and reviewed.

### 3.7.1 Genetic Algorithms

Genetic Algorithms (GA) were initially introduced by Holland (1975) for the process of biological evolution. In project scheduling, a GA starts with a population of *n* solutions. A number of initial solutions may be generated by applying one single pass or multi-pass methods, and then new solutions are produced by using genetic operators. When the population is reached, the fittest solutions survive to make up the next generation while the others are discarded. This process is repeated until the predetermined number of generations is reached.

GAs for single mode RCPS have been proposed by Leon and Balakrishnan (1995), Lee and Kim (1996), Cheng and Gen (1998), Hartmann (1998) and Alcaraz and Maroto (2001) with different representations of solutions. GAs for multimode resource-constrained project scheduling with the objective of the minimisation of the project completion time are also presented by López *et al.* (1996), Özdamar (1999) and Hartmann (2001). Mori and Tseng (1997) applied GA to multimode stochastic project scheduling problems.

### 3.7.2 Simulated Annealing

Simulated annealing (SA) was introduced by Kirkpatrick *et al.* (1983). It originated from the process of metal cooling gradually to a low energy state. An initial solution can be obtained by a scheduling scheme with a priority rule. A neighbour solution is generated by slightly perturbing the current one. If the new solution is better than the current one, the new solution is definitely accepted. If the new solution is worse than the current one, the new solution is accepted only if it meets a certain probability criterion.

SA based heuristics for RCPS have been proposed in various presentation schemes with different scheduling methods by Sampson and Weiss (1993), Boctor (1996b), Lee and (1996), Cho and Kim (1997), Liu and Wang (2000), in terms of the minimisation of the project completion time. Yang (1995) presented SA for the maximisation of the project net present value. Józefowska *et al.* (2001) applied SA for multi-mode resource-constrained project scheduling.

### 3.7.3 Tabu Search

Tabu search (TS), developed by Glover (1989, 1990), is a neighbourhood search approach using a steepest descent/mildest ascent method in order to move the current solution to a neighbouring one. Then the best solution is chosen. To avoid the possibility of moving back to the same local optima, the short-term and/or long term memories can be employed to record the move status.

Pinson *et al.* (1994) proposed three TS heuristics with the serial scheduling scheme. Lee and Kim (1996) presented a TS based on a random key representation scheme. Baar *et al.* (1998) developed two TS heuristics. The first one employs the serial scheduling scheme with the priority list representation scheme whilst the second one uses the encoding procedure. All these approaches are concerned with the minimisation of the project completion time. Icmeli and Erenguc (1994) presented a TS for RCPS involving cash inflows and outflows for the maximisation of the net present value.

### 3.7.4 Neural Networks

The human brain is complex biological network of hundreds of billions of special cells called neurons. These neurons send information back and forth to each other through connections so as to enable learning, analysis, and prediction. From such a biological base, Hopfield (1982) and Hopfield and Tank (1985) proposed neural networks for solving optimisation problems. Neural networks are formed from hundreds of simulated neurons that are connected in much the same way as the brain's neurons, to learn and analyse the outcome by gathering various information. In optimisation problems, the network has a capability of jumping from one "energy basin" to another in the energy landscape in order to escape from the local optima. The external neuron provides a form of transient feedback.

Vaithyanathan and Ignizio (1992) proposed a neural network to modelling RCPS for minimising the project completion time. Zhu and Padman (1999) developed a neural network to induce the relationship between problem [illegible]arameters and heuristic performance in selecting heuristic approaches for RCPS. [illegible] (1998) suggested two networks for RCPS. The primary network makes the calculation of weig[illegible] [illegible]or each process and the secondary network makes a proposal for modification of schedu[illegible] based on the primary network.

## 3.7.5 Ant Colony Optimization

Ant colony optimisation (ACO), as a multi-agent approach, was first proposed by Dorigo *et al.* (1996), and Dorigo and Di Caro (1999a, 1999b) to solve combinatorial optimisation problems. It was first successfully applied to the travelling salesman problem (TSP) (Dorigon and Gambardella 1997a, 1997b).

ACO was inspired by the observation of the behaviour of real ant colonies by pioneers, Dorigo *et al.* (1996). Because ants are blind, they often follow the pheromone trail from a food source to the nest, and then vice versa. Pheromone is a substance deposited on ground as an individual ant passes by. Ants often follow their mates' pheromone trails to look for food and to go back the nest. As time goes by, a number of trails may be available from the nest to a food source. However, a short path will be finally detected by its heavier pheromone trail because ants realise that they spend less time using the path than other trails. Analogously, for ACO, if "artificial ants" found a good solution, they will mark their paths by depositing an amount of pheromone. The following ants of the next generation are attracted by the pheromone, and search in the solution space for good solutions, marked by greater deposits of pheromone.

For RCPS, the idea of ACO is to employ an ant algorithm for deciding which activity from the decision set should be scheduled using any scheduling schemes. There are only two published papers about applying ACO to RCPS with a similar approach (Bautista and Pereira 2002, Merke *et al.* 2002). Both papers apply the serial scheduling scheme in ACO. For the selection of an activity, the ant uses the heuristic information as well as pheromone information. The heuristic information for activity $j$ is the priority information by applying a specific priority rule whereas the pheromone information is the information about good solutions found by former ants. Therefore both the heuristic and the pheromone information indicate how good the schedule will be in scheduling activity $j$. Both papers try to find near optimal solutions where the project completion time is minimised.

## 3.8 Project Scheduling under Uncertainty

Project scheduling has been widely used in industry and public organisations to plan and manage projects. However, information related to the projects is often predetermined to be crisp data. In reality, uncertainty often inherently occurs in many projects. Precise information is seldom available in some contexts as it is difficult to predict how much time activities of a project will take to complete. Such uncertainty around activities can be of probabilistic or possibilistic nature.

For the nature of randomness, only when sufficient information is available about observations of past performance of an activity, the technique of probabilistic distributions can be applied to the activity to obtain the activity durations in the presentation of probability (Harrison and Tamaschke 1993).

Quite often the activities of a project that are similar to the activities for which past information is available may be often of limited relevance in practice. Because of lack of sufficient information about these activities, it is impossible for the DM to specify the duration of the activities. As a consequence, the duration of activities may be intuitively estimated in linguistic terms by the DM, using their knowledge from the limited available information, their experience and judgement. Linguistic terms such as "approximately" and "from... to" are an easy way of estimating activity duration times relying on human thinking and judgement. For instance, activity 1 may be performed in "approximately" 5 days, and the duration of activity 2 could be "from 3 to 6 days". Clearly, these terms are imprecise in nature, leading to a range of possible values rather than a definite estimate of activity duration times. Fuzzy set theory is an effective way of dealing with such imprecision and vagueness (Zadeh 1978, Yeh *et al.* 1999a, 1999b).

## 3.9 Current Research in Fuzzy Project Scheduling

In the real world, many project scheduling problems are often inherently uncertain because of lack of precise information about activity duration times or because of very limited availability of past information about similar types of activities. In such circumstances, it is difficult to precisely estimate activity duration times. For this reason, more and more academic researchers have become aware of this kind of uncertainty and take this issue into consideration in the field of project scheduling to resolve difficulties in real-world project scheduling instances.

Fuzzy set theory, as it applies to project scheduling has three categories: (a) Fuzzy project planning and scheduling without concern about resource constraints; (b) RCPS with fuzziness; and (c) MMRCPS in a fuzzy environment. In this section, these three categories of fuzzy project scheduling will be reviewed. The remainder of this section highlights the contributions made in my PhD research to fuzzy project scheduling under resource constraints in both single and multiple executive modes of an activity.

Prade (1979) first introduced the concept of fuzzy sets and combined fuzzy sets into program evaluation and review technique (PERT) in project scheduling. Chanas and Kamburowski (1981) presented detail of estimating the project completion time through the so-called FPERT technique using fuzzy sets. McCahon and Lee (1988) proposed project network analysis methods to determine fuzzy project completion with the degree of criticality of each network path. Later, McCahon (1993) applied the PERT technique as an approximation in fuzzy project network analysis. Rommelfanger (1994) presented a new method for determining the start and slack times of activities in network analysis and compared his method with other well-known fuzzy network analysis techniques in the literature. Fargier *et al.* (2000) integrated a series-parallel graphs method into fuzzy PERT for project scheduling. Wang and Fu (1996) introduced inflation concerns into scheduling and developed four different fuzzy project scheduling models. All these methods proposed by researchers have not taken resource constraints into consideration.

For fuzzy RCPS, Hapke and Slowinski (1993), Hapke *et al.* (1994) proposed a fuzzy single-pass parallel heuristic with twelve priority rules. Later Hapke and Slowinski (1996) presented a similar approach, but added the serial scheme to fuzzy RCPS. Lorterapong (1994), proposed a fuzzy CPM method with priority rules for resource allocations in fuzzy RCPS. In this approach, three criteria: (a) minimising project completion time; (b) maximising resource utilisation; and (c) minimising resource interruption, are used to measure performance in scheduling. Wang (1999) presented a fuzzy scheduling procedure with beam search for resolving practical-sized RCPS problems. Hapke *et al.* (2000) developed the fuzzy Pareto SA to meet multiple objectives in fuzzy RCPS with the application to scheduling of an agriculture project for minimising three objectives: (a) the project completion time, (b) the deviation of the average resource utilisation, and (c) the total project cost.

Considering a fuzzy environment in MMRCPS, Hapke *et al.* (1999) extended their approach to multiple objectives in fuzzy single-mode RCPS (Hapke *et al.* 2000) to handle multi-modes for MMRCPS. Özdamar and Alanya (2001) proposed a nonlinear mixed binary mathematical formulation combined with four priority rule-based heuristics in an attempt to minimise the project completion time for fuzzy MMRCPS with the case study of software development project.

In my research, I introduced the $\alpha$-cut concept of fuzzy sets into project scheduling to express the DM's degree of confidence in estimating activity duration times (Willis *et al.* 1999). This concept is practically useful in refecting the vagueness of estimating activity duration times in cases where the DM knows the properties of activities very well, and trains the technical and general workers so as to match the similar skill levels. This concept allows DMs to express their different confident levels in assessing activity duration times depending on whether the DM knows well about the nature of an activity itself, the reliability of machine and equipment, and the skills of workers and technicians that will participate in the activity.

To solve multiple objectives in a fuzzy environment of RCPS, I developed a fuzzy goal programming model incorporating DSS so as to enable the DM to make modifications through an interactive interface, based on their experience and the circumstance that they know to have produced a significantly realistic schedule in the past (Willis *et al.* 1998, 1999).

To obtain a practical and reasonable solution with a simple and effective computational effort, I developed a fuzzy rule-based heuristic approach that employs a set of ten priority rules incorporating one scheduling scheme (Yeh *et al.* 1999c, Pan *et al.* 2001a). This approach has been applied to a real overhaul project scheduling problem in the dredging industry, producing a good scheduling solution. The background knowledge of dredging refers to Pan *et al.* (1998). This approach is simple and straightforward for any practically-sized RCPS problems involving uncertain activity duration times modelled by fuzzy numbers.

Although priority rule-based heurists are simple and straightforward in the resolution, they do not always provide good solutions in terms of optimisation. To gain a good solution, I developed two versions of fuzzy GA for fuzzy RCPS: (a) fuzzy pure GA, and (b) a fuzzy hybrid GA that combines a tabu mechanism. In both approaches, the scheduling problem is formulated as a special chromosome where the number of genes matches the number of activities. In the formulation, the location in a chromosome acts as the activity number, and a value in each location represents the priority value of that activity. There is a special mechanism built into both approaches to always assign different values to each activity, thus avoiding producing the same priority values in some activities when a priority rule is applied. In addition, under such a formulation, all solutions generated in the evolutional process are always feasible. Moreover, both versions of GA have intelligent search strategies in guiding the search for obtaining the approximate globally optimal solution. The experiment shows that the GA with a tabu mechanism is superior to the pure GA as it avoids generating some chromosomes that have been generated recently so as to save enormous unnecessarily computational time. (Pan *et al.* 2001b, 2001c, 2003a).

In this research, I also developed another two versions of SA for fuzzy RCPS: (a) pure SA, and (b) SA with a tabu mechanism. In both approaches, the solution coding is well designed to reflect the features of fuzzy RCPS as well as searching requirements. Such a coding representation can initially indicate the objective function while being processed. Secondly, the current solution coding can be manipulated easily for generating a neighbourhood solution without any distortion. Thirdly, under such a solution coding, the recalculation of the overall schedule for neighbourhood solutions is avoided, thus dramatically reducing computational time in the neighbouring searching. As the experiment has demonstrated, the SA with a tabu mechanism surpasses the pure SA if the tabu size is chosen properly because the SA with tabu can avoid going back to the previous solution spaces as so to extend more diverse solution spaces in finding an approximate globally optimal solution (Pan *et al.* 2002; 2003b)

In comparison with the fuzzy single-mode RCPS, fuzzy MMRCPS has a greatly practical significance in representing the different ways of performing an activity in a project. However, such a problem may probably be the most difficult scheduling problem because of the complexity of complete NP-hardness in a strong sense. In my PhD studies, I have spent a significant time in developing a number of algorithms for MMRCPS in a fuzzy environment, including (a) the fuzzy priority rule-based heuristic, (b) the fuzzy GA, (c) the fuzzy GA with tabu mechanism, (d) the fuzzy SA, and (e) the fuzzy SA combined with tabu mechanism. All these algorithms were extended from the algorithms I developed for fuzzy single-mode RCPS. To deal with MMRCPS, two subproblems need to be properly resolved together: (a) the mode-assignment problem, and (b) the resource-constrained scheduling problem. The mode assignment is an important part of MMRCPS. A number of options for mode selection are built into each algorithm. These are: (a) the mode assigned to the activities may not be changed for a number of iterations, (b) the mode assignment for one or more activities may be changed randomly, (c) the modes may be swapped among several activities, (d) modes may be changed entirely for all activities of a project. The options for mode assignment are determined by the current search status depending on whether the current solutions are improved recently during search, based on the intelligence rule base built in the system which can be modified and predetermined by the DM. The intensive studies and experiments have been conducted to investigate the behaviours of these algorithms for finding an approximate globally optimal solution in search process (Pan *et al.* 2001a, 2003c, 2003d).

Fuzzy RCPS and fuzzy MMRCPS are important issues raised in project scheduling as fuzziness often inherently exists in such problems, and it cannot be ignored. As indicated in the review above, there are only a limited number of papers published in fuzzy project scheduling, of which most publications are not concerned with resource constraints (Yeh *et al.* 1999c). There is a scarcity of published papers dealing with fuzzy RCPS in both single and multiple modes. Therefore, project scheduling with fuzziness is still a new and challenging field. Vigorous research is required to develop new effective and efficient approaches to problem solving within realistic frameworks.

## 3.10 Concluding Remarks

This chapter has reviewed project scheduling from its earliest to the most recent development, particularly in resource-constrained project scheduling for single and multiple modes. This field has been under the academic research for nearly four decades. Various approaches have been developed to resolve project scheduling problems in a more meaningfully practical sense.

In terms of exact algorithms, most approaches are based on the idea of the branch-and-bound technique, enumerating exhaustively to find an optimal solution for scheduling. These approaches are mathematically complex as well as very time demanding in computation. Although these approaches may be able to find an exact optimal solution in very small-sized problems, they are unable to handle practically-sized project scheduling problems encountered in the real word.

To find ways of dealing with project scheduling problems of any size, a number of heuristic approaches based on priority rules have been developed for prioritising the eligible activities in competing scarce resources during scheduling. These approaches are basically grouped into two categories: (a) single pass methods, and (b) multi-pass methods. These approaches are simple, and require far less computational time. However, they may produce a poor solution for the scheduling problem in terms of optimisation.

To obtain an approximate globally optimal solution in project scheduling of practical sizes, some researchers have made a great effort to develop a number of metaheuristics. These include: (a) genetic algorithms, (b) simulated annealing, (c) tabu search, (d) neural networks, and (e) ant colony optimisation. As the review has shown, there are only a handful of papers published in the development of these approaches for resolving project scheduling problems. More work still needs to be carried out to provide realistic approaches based on the idea of metaheuristic methodologies for different practical demands.

The above-mentioned approaches regard the project information as crisp without uncertainty. However, the reality is that often activities similar to those in any given project may not have been previously performed or the information about them is hardly available. Uncertainty based on the possibilistic nature of activities often inherently exists in many real projects. In recent years, researchers start being cognizant of the nature of vagueness and imprecision that is innate in many real project scheduling problems. However, as the review has shown, more efforts need to be made in fuzzy project scheduling, since there are not many publications available on this subject, particularly about resource constraints. More attention is urgently needed to focus on the circumstances of fuzziness in order to provide effective and efficient frameworks with greatly practical significance in resolving many project scheduling problems in realistic settings.

# Chapter 4

# Fuzzy Multi-Mode Resource-Constrained Project Scheduling (FMMRCPS)

## 4.1 Introduction

Project scheduling with resource constraints is an important issue in project managerial decision making and appears frequently in a large scope of real-world situations. Because of the importance of practical applications, Kelley (1963), Lambourn (1963) and Wiest (1964) superimposed resource constraints upon project scheduling. Generally speaking, the problem of this kind can be characterised as the need to allocate scarce resources over time to a set of activities of a project where precedence relationships exist, in order to satisfy a specific goal or objective such as minimising project cost, minimising project completion time, or maximising net present value of a project.

RCPS has been the focus of much research for the last four decades since it was first introduced in project scheduling. However, a vast number of publications have assumed that each activity can only be performed in a single manner, whereby the activity has only one duration time with its resource requirements. In practice, however, activities of a project may be performed in one of a number of possible manners, because an activity may be performed in various duration times by different workable resource settings. This scheduling problem has great practical significance in representing different ways of performing a certain activity. To meet realistic situations, Elmaghraby (1977) first introduced a project scheduling model, called multiple mode resource-constrained project scheduling (MMRCPS) where an activity may have several executive modes. Each mode is represented by its own duration with the corresponding resource requirements. Later Slowinski and Weglarz (1978) suggested the multi-mode preemptive case where activities can be interrupted while being processed, and resource-duration tradeoffs can be expressed as a continuous function. These scheduling problems are presumed to be deterministic in nature.

Because of their nature, uncertainty often subsists in many real projects. To deal with uncertainty, some stochastic approaches have been suggested in RCPS (Davis and West 1987, Garavelli and Pontrandolfo 1995, Fernandez *et al.* 1996, 1998, Tsai and Gemmill 1998, Golenko-Ginzburg and Gonik 1998, and Gutjahr 2000). However, this uncertainty can only be captured in sufficiently quantifying the information observed in the past. In reality, many activities in a project may never have been preformed or precise information regarding the activities is sparse. The uncertainty of this kind can only be captured in linguistic terms, representative of subjective judgements based on human knowledge and experience. Fuzzy set theory is an effective way of handing such uncertainty (Zadeh 1965, Zimmermann 1986). Fuzzy multi-mode resource-constrained project scheduling (FMMRCPS) is the main issue considered in my PhD research.

To address FMMRCPS, relevant literature on current developments in MMRCPS, including fuzziness, is presented first. Next, this chapter addresses the practical significance of FMMRCPS, followed by a presentation of the problem description for FMMRCPS.

## 4.2 Current Developments in MMRCPS

Since Elmaghraby (1977) extended RCPS to the more realistic, MMRCPS model, research efforts have been made to propose various approaches for this model because of its power in mapping many situations in practice. As part of my PhD research on MMRCPS, current developments of MMRCPS in both deterministic and non-deterministic will be reviewed in this section.

Exact approaches developed for MMRCPS are mainly based on enumeration schemes and branch-and-bound procedures. Talbot (1982) presented a two-stage solution approach using enumerations and eight priority rules. Patterson *et al.* (1989)

introduced a precedence tree, allowing systematic enumeration of mode assignment and start time of activities. Speranza and Vercellis (1993) proposed the depth-first search in the branch-and-bound procedure. Sprecher *et al.* (1997) applied mode alternatives with static and dynamic search tree reduction techniques in enumeration scheme. Sprecher and Drexl (1998) presented new dominance criteria for the branch-and-bound procedure. Heilmann (2003) proposed a depth-first search based on the branch-and-bound technique with a branching strategy where branching rules are selected dynamically. Brucker and Knust (2003) developed a destructive lower bound method based on both constraint propagation and linear programming. However, MMRCPS is NP-hard in the strong sense because two decisions have to be made: (a) mode assignment, and (b) the sequence of activities in scheduling. The experiments reported show that exact approaches may only solve problems to optimality where sizes are small with up to 30 activities (Bouleimen and Lecocq 2003).

Heuristic approaches often provide MMRCPS problems with feasible solutions. Boctor (1993) proposed a single-pass approach with partial mode assignments. Slowinski *et al.* (1994) applied a multi-pass approach with sampling to MMRCPS problems. Drexl and Grünewald (1993) proposed a biased random sampling approach with a serial scheduling scheme. Boctor (1996a) presented a parallel scheduling scheme, choosing a set of nondominated eligible activities when a lower bound of prolonging the project completion time is calculated. Bianco *et al.* (1998) proposed a heuristic solution approach based on the mode graph technique and the interactive solution scheme for determining mode and activity sequence. Sprecher and Drexl (1998) employed the branch-and-bound procure as a heuristic process. Although these heuristic approaches always produce feasible solutions, optimal or near optimal solutions may not be guaranteed.

To achieve better results for MMRCPS, some local search methods and metaheuristic approaches have also been proposed. Kolisch and Drexl (1997) proposed a new local search, in which a feasible solution is initially found, a single-neighbourhood search is then performed on the set of feasible mode assignments. Finally improved solutions will be found in the intensification phrase. Mori and Tseng (1997) proposed a GA, which assignes activity order sequence and mode randomly to produce a number of solutions in the first place, then applies genetic operators to these solutions. However, some generated solutions may be not feasible in the process of their GA approach. Hapke *et al.* (1998) proposed a Pareto SA combined with the light beam search for multiple objectives. De Reyck *et al.* (1998) presented a tabu search procedure based on decomposition of the problem into a mode assignment and fixed mode RCPS phrases. Özdamar (1999) presented a GA approach, applying 15 priority rules, to determine the priority for each activity. Pan and Yeh (2003a) developed a GA approach combined with a tabu mechanism, to obtain better solutions than are available using a GA alone. Józefowska *et al.* (2001) presented two versions of SA: (a) no penalty function, and (b) with penalty function. Both versions applied three neighbourhood generation mechanisms: (a) neighbourhood shift, (b) mode change, and (c) combined move with the former two mechanisms. Hartmann (2001) developed a GA incorporating two local search methods: (a) one local search designed to deal with feasibility of MMRCPS, and (b) the other local search used to improve scheduling solutions found in GA. Jozefowska *et al.* (2002) proposed an SA and tabu search for cash flow in MMRCPS. Bouleimen and Lecocq (2003) developed an SA approach using two embedded search loops for exploring neighbourhood solutions by alternating activities and modes.

In cases where a fuzzy environment is taken into account for MMRCPS, the term, FMMRCPS is used here as specified in Section 4.1. Hapke *et al.* (1999) proposed the fuzzy Pareto SA for solving multiple objectives with a two-stage solution procedure: (a) the Pareto SA procedure is used to generate a sample of the set of approximately non-dominated schedules in weak comparison rule (WCR), and (b) the interactive procedure is used to search for a schedule that is the best compromise among conflicting fuzzy objectives over the sample. Özdamar and Alanya (2001) proposed a simple GA where four priority rules are employed for determining priorities of eligible activities, competing for scarce resources. In my PhD research, I presented a fuzzy rule-based heuristic approach where a set of ten priority rules are employed incorporating the parallel scheduling scheme (Yeh *et al.* 1999c, and Pan *et al.* 2001a). I developed two versions of GA approaches: (a) the pure GA, and (b) GA combined with tabu mechanism (Pan *et al.* 2001b, 2001c, 2003a). We found experimentally that GA with tabu mechanism generally outperforms the pure

GA. I also developed two versions of SA approaches: (a) the pure SA, and (b) SA with tabu mechanism for minimising project completion time (Pan *et al.* 2002, 2003b) The experiment demonstrates that SA with tabu mechanism works better than the pure SA if the size of tabu is predetermined properly in terms of effective and efficient computational efforts.

Although various approaches have been developed to MMRCPS including exact, heuristic, and metaheuristic approaches, publications focusing on a fuzzy environment are seldom available. This motivates me to research a more realistic model of MMRCPS with fuzziness in this challenging field.

## 4.3 Practical Significance of FMMRCPS

RCPS has been widely used in industries, engineering, sciences and management. RCPS commonly exists in many projects and scientific undertakings, requiring scheduling in an efficient way under limited resources so as to reach the specific goal of completing the project or work such as time or budget. Therefore, RCPS has been an important part of project management. Nowadays, RCPS permeates almost every corner of practical life and has become the topic of interdisciplinary research.

The basic RCPS assumes that each activity can only be performed in one single manner. This basic model is unable to cover the possibility that an activity may be optionally performed under different resource requirements. The generalised RCPS, called MMRCPS, enables covering both single executive mode and multiple alternative modes for any activities. Thus MMRCPS accommodates a wide range of project scheduling in practice.

MMRCPS is already a powerful model for mapping many realistic circumstances occurring in a project (Hartmann 1999). However, uncertainty, an important issue in project scheduling, is not incorporated in MMRCPS, because activities contained in a project may never have been carried out, or because information on those activities is inadequate. Imprecise information in a project often exists in practical applications. The issue of vagueness of activity duration times in a project is addressed here, and this ambiguous phenomenon must be properly handled. Hapke *et al.* (1999) first proposed FMMRCPS, which model enables the capture of all possibly practical situations occurring in a project. FMMRCPS has the great practical advantage of representing all possible aspects of circumstances for project scheduling. It is the most powerful and generalised model of RCPS.

## 4.4 Problem Description of FMMRCPS

The FMMRCPS dealt with in my PhD research is given below. A project consists of $J$ activities (1, 2,..., $J$) where activity duration times are uncertain because of lack of imprecise information on activities. The precedence relationships of activities in a project can be represented by an acyclic activity-on-node (AON) network (Crandall 1973), as shown in the detailed example of AON network of the overhaul project presented in Figure 3.2. To keep generality of the topological order in activities of a project, the activity number is always larger than that of all its predecessors. Each activity $j$ may be performed or executed in one of a number of alternative modes. The type of resources consumed by activities is considered renewable, and a detailed explanation about this type of resources has been given in Section 3.2.1.1. The reason for only taking renewable resources into account is that, this type of resources is the most commonly used in project scheduling. Modes of an activity are to be sorted in non-decreasing durations. It is assumed that, once an activity is started in one of its several execution modes, the activity must be finished in that mode and any mode change or interruption during its process is not allowed.

In the following subsections of this chapter, the objective of FMMRCPS in my research will be discussed. To clearly describe the model of FMMRCPS, the assessment of fuzzy duration times for activities and mathematical presentation of the problem will be presented.

### 4.4.1 Objective of FMMRCPS

Project scheduling has attracted growing attention from both academics and practitioners, and it is popularly applied in a broad number of realms. Appropriate single and multiple objectives are put forward by researchers and the DM alike to meet specific demands in a wide diversity of project scheduling environments. To deal with different objectives for performing project scheduling, a number of objectives have been proposed for real-word applications to match individual criterion or criteria requirements.

Time-based objectives are the most common measurements, including (a) minimising project completion time (Bell and Park 1990, Böttcher *et al.* 1999, Bianco *et al.* 1999, Brucker and Knust 2000), (b) minimising weighted delays if due dates are given (Sprecher and Drexl 1998), and (c) minimising the maximum tardiness (Baker 1974). In resource-based objectives, resource-cost is of most concern because the schedule of activities influences the cost indirectly via the status of resource usage over time. These objectives can be: (a) minimising resource levelling (Neumann and Zimmermann 2000), (b) minimising the cost by both levelling resources and delaying project completion time within deadline (Mason and Moodie 1971), and (c) minimising the resource investment, where current resources are limited, in order to avoid delay in project completion time (Demeulemeester 1995, Möhring 1984). Financially-based objectives can be concerned with: (a) maximising the net present value (Elmaghraby and Herroelen 1990, Baroum and Patterson 1996), (b) minimising the total project cost (Karshenas and Haber 1990). In some cases, multiple objectives are compromised so as to meet a specific circumstance in project scheduling (Hapke *et al.* 1999).

In my PhD research, both single and multiple objectives are considered for FMMRCPS. For multiple objectives, both (a) project completion time, and (b) project cost are taken into consideration because these two objectives are looked upon as two important issues from the point of view of practical value. The approach developed by fuzzy goal programming with soft constraints is applied to FMMRCPS to get the best compromise between these conflicting objectives in fuzzy project scheduling. Later, I develop a number of both heuristic and metaheuristic approaches, mainly focusing on the single objective of the minimisation of project completion time in FMMRCPS. Although a number of objectives can be applied to project scheduling, the minimisation of project completion time is the most common concern in practical situations (Sprecher *et al.* 1995, Kolisch and Padman 2001). Quite often projects are scheduled so as to utilise the current capacity of resources of organisations without hiring any extra resources, and the goal is to attempt to complete a project as soon as possible.

### 4.4.2 Fuzzy Activity Duration and $\alpha$-cut

In many realistic circumstances, precise information on activities of a project is often seldom available. DMs often determine an impossibly sharp duration time for an activity, instead of intuitively describing in linguistic terms such as 'most likely', 'unlikely', and 'approximately'. In fact, the evaluation of activity duration times by DMs reflects a subjective assessment through their knowledge and experience that can be expressed as a fuzzy number. A membership function of the fuzzy number can effectively describe this assessment. My experience in undertaking a number of projects whilst having worked in industry, has led me to believe that triangular and trapezoidal fuzzy numbers are a convenient and practical way of representing activity duration times in any situation, and are easily and intuitively described by DMs using their linguistic terms.

The fuzzy duration time, $\tilde{d}_j$ of activity $j$ is represented by a triangular fuzzy number $\tilde{A}$, denoted as $(a_1, a_2, a_3)$, shown as in Figure 4.1. $a_1$ and $a_3$ represent the unlikely duration times of activity $j$ as the lower and upper bounds of the support of the fuzzy number for reflecting the fuzziness of the DM's assessment. $a_2$ is the assessment value for the most likely duration time with a degree of the membership being 1. The membership function of fuzzy duration time, $\tilde{d}_j$ of activity $j$ can be represented in formula (4.1). For example, the duration of activity 5, is estimated by the DM in linguistic terms so that this activity is most likely to be completed in approximately 10 days and is most unlikely finished within 7 days or beyond 14 days. Therefore the fuzzy duration time $\tilde{d}_5$ of activity 5 can be written as $\tilde{d}_5 = (7, 10, 14)$.

$$\mu_{\tilde{d}_j}(t) = \begin{cases} 0 & t < a_1 \text{ or } t > a_3, \\ \dfrac{t - a_1}{a_2 - a_1} & a_1 \le t \le a_2, \\ \dfrac{a_3 - t}{a_3 - a_2} & a_2 \le t \le a_3. \end{cases} \tag{4.1}$$

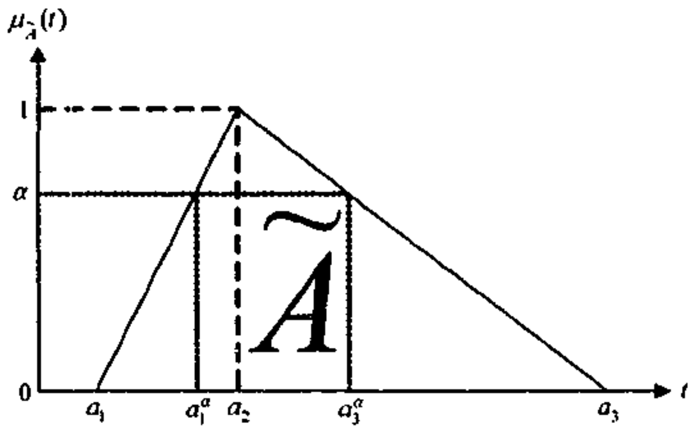


Figure 4.1 Triangular Fuzzy Number and its $\alpha$-cut

If the fuzzy duration time, $\tilde{d}_j$ of activity $j$ is represented by a trapezoidal fuzzy number $\tilde{B}$, the approximate distribution of linguistic values can be expressed as 4-tuples $(b_1, b_2, b_3, b_4)$ shown as in Figure 4.2. $b_1$ and $b_4$ represent the unlikely duration times of activity $j$ as the lower and upper bounds of the support of the fuzzy number for reflecting the vagueness of the DM's judgement. The interval $[b_2, b_3]$ is the evaluation value for the most likely duration time, whose degree of the membership is 1. Let $\mu_{\tilde{B}}^L(t)$ be the strictly continuous left spread for interval $[b_1, b_2]$ and $\mu_{\tilde{B}}^R(t)$ be the strictly continuous right spread for interval $[b_3, b_4]$. The membership function of fuzzy duration time, $\tilde{d}_j$ of activity $j$ in Figure 4.1, can be expressed in formula (4.2). To describe the fuzzy duration time of an activity, applied to trapezoidal fuzzy numbers, one example is given here. The duration of activity 3, assessed by the DM is that activity 3 is most likely accomplished approximately from 5 to 8 days, and is unlikely to be completed within 3 days or in more than 11 days. The duration $\tilde{d}_3$ of the activity can be expressed as $\tilde{d}_3 = (3, 5, 8, 11)$.

$$\mu_{\tilde{B}}(x) = \begin{cases} \mu_{\tilde{B}}^{L}(t) = \dfrac{x - a_1}{a_2 - a_1}, & a_1 \le t \le a_2 \\ 1, & a_2 \le t \le a_3 \\ \mu_{\tilde{B}}^{R}(t) = \dfrac{a_4 - x}{a_4 - a_3}, & a_3 \le t \le a_4 \\ 0, & \text{otherwise} \end{cases} \tag{4.2}$$

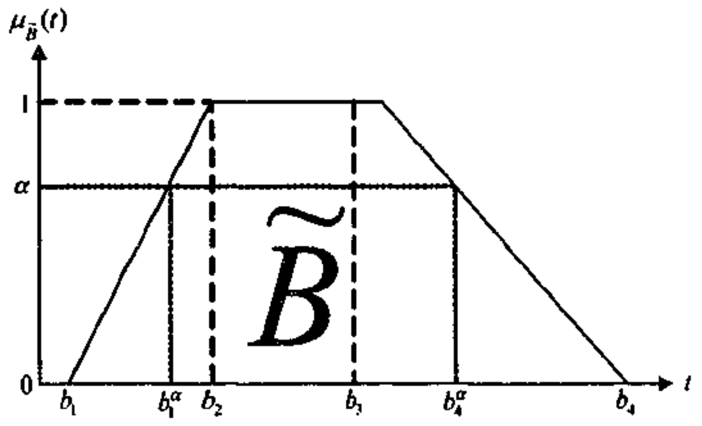


Figure 4.2 Trapezoidal Fuzzy Number and its $\alpha$-cut

The $\alpha$-cut concept is useful to express the DMs view on the minimum degree of acceptance when assessing activity duration times for a specific project. The level of $\alpha$-cut depends on the DMs' confidence in their assessments of activity duration times according to their knowledge and perceptions of the project situation. A larger $\alpha$ value indicates a higher degree of confidence for the DMs with their assessments in narrower intervals symmetrically distributed around the most possible value of the evaluation. An $\alpha$-cut of the fuzzy duration time interval $\tilde{d}_j^\alpha$ of an activity $j$ can be mathematically expressed as:

$$\tilde{d}_j^\alpha = \{t, \mu_{\tilde{d}_j}(t) \ge \alpha, \alpha \in [0,1]\} \tag{4.3}$$

### 4.4.3 Mode representation

In determining mode options for scheduling FMMRCPS, a given set of various allowable performing modes may be assigned to a specific activity. These performing modes for an activity are called executive modes of an activity. Each mode of an activity is characterised by a duration time and amounts of different resources required for the activity. For example, an activity may have three executive modes. In mode 1, the activity can be finished in the fuzzy period of (3, 6, 8), requiring 6 workers and 4 machines. For mode 2, the activity can be completed in the fuzzy period of (4, 8, 10), demanding 4 workers and 3 machines. In mode 3, the activity can be accomplished in the fuzzy period of (6, 11, 14), needing 4 workers and 2 machines. That is, there are three options of executive mode, of which the activity is allowed to choose one. Different mode selections for activities result in significant differences in scheduling solutions.

Activity $j$ may consist of a set of executive modes, $\{1, 2, \ldots, M_j\}$. activity $j$ can be processed in mode $m$ of $M_j$ possible modes with fuzzy duration time $\tilde{d}_{jm}$, requiring amounts of a set of resources $\{r = 1, 2, \ldots, R | k_{jmr}\}$ where $R$ is the number of renewable resources. Therefore a mode of an activity presents a specific performance for the activity characterised by its individual fuzzy duration and the corresponding resource requirements.

### 4.4.4 Mathematical Description

In this subsection of the mathematical description, I mainly analyse FMMRCPS for the single objective of minimising the project completion time, addressing important issues for this particular type of problem while formulating the FMMRCPS model. All the heuristic and metaheuristic approaches I developed are based on this model. Research involving multiple objectives for FMMRCPS will be presented in Chapter 6.

In terms of minimising the project completion time, the goal is to make an attempt to find a schedule that enables a project to be completed as soon as possible This is achieved, by assigning which of a number of executive modes of activities to best achieve the primary objective under the conditions of precedence relationships and resource constraints. To better understand the nature of FMMRCPS, this problem is presented in mathematical form in this subsection. A given project comprises of $J$ activities ($j$ =1, 2,..., $J$), which are subject to the constraints of precedence relationships and resource limitation. Activity $j$ can be executed in mode $m$ of $M_j$ modes ($m$ = 1, 2,..., $M_j$) with its corresponding resource requirements $\{ k_{jmr}, \forall r \in R \}$. Once an executive mode is assigned to an activity and it is scheduled, any mode changes or interruption of the activity is not permitted, that is, the activity becomes *non-preemptive*. Let $\tilde{t}$ be the fuzzy time period, and $x_{jmt}$ be a binary variable, whose value is defined as:

$$x_{jm\tilde{t}} = \begin{cases} 1 & \text{if activity } j \text{ is performed in mode } m \text{ at time } \tilde{t} \\ 0 & \text{otherwise} \end{cases} \tag{4.4}$$

The fuzzy project completion time, defined as $\tilde{H}$, is heuristically determined by adding fuzzy duration times of all activities of a project in mode 1 (the mode with shortest finish time of activities) only and it can be calculated by:

$$\tilde{H} = \sum_{j=1}^{J} \tilde{d}_{jm}, \qquad m = 1 \tag{4.5}$$

Let $EF\tilde{T}_j$ and $LF\tilde{T}_j$ be the fuzzy earliest and latest finish times of activity $j$ respectively. Section 3.2.4 has shown how to calculate the earliest and latest finish times of an activity. FMMRCPS for minimising fuzzy project completion time can be formulated as follows:

$$\min \sum_{m=1}^{M_J} \sum_{\tilde{t}=EF\tilde{T}_J}^{LF\tilde{T}_J} \tilde{t} \times x_{Jm\tilde{t}} \tag{4.6}$$

subject to $$\sum_{m=1}^{M_j}\sum_{\tilde{t}=EF\tilde{T}_j}^{LF\tilde{T}_j} x_{jm\tilde{t}} = 1 \qquad j=1,2,\ldots,J \tag{4.7}$$

$$\sum_{m=1}^{M_j}\sum_{\tilde{t}=EF\tilde{T}_j}^{LF\tilde{T}_j}(\tilde{t}-\tilde{d}_{jm})\times x_{jm\tilde{t}} - \sum_{m=1}^{M_h}\sum_{\tilde{t}=EF\tilde{T}_h}^{LF\tilde{T}_h}\tilde{t}\times x_{hm\tilde{t}} \geq 0 \quad j=1,2,\ldots,J \tag{4.8}$$

$$\sum_{j=1}^{J}\sum_{m=1}^{M_j} k_{jmr}\sum_{s=\tilde{t}}^{\tilde{t}+\tilde{d}_{jm}-1} x_{jms} \leq K_r \quad r=1,2,\ldots,R;\ \tilde{t}=1,\ldots,\tilde{H} \tag{4.9}$$

As given above, an FMMRCPS problem can be expressed as the fuzzy 0-1 programming model with fuzzy constraints. Objective function (4.6) is to ensure that the project will be completed as soon as possible. Constraints (4.7) allow an activity to be scheduled only once over the whole project. Constraints (4.8) describe the precedence relationships among activities of a project; Constraints (4.9) state that the resource usage cannot exceed the resource availability. As indicated by the FMMRCPS model, there are a great number of fuzzy constraints when |J| >3, as two decisions are required to be made: (a) selecting activity modes, and (b) determining the sequence of activities in FMMRCPS, and, at the same time, satisfying precedence relationships and resource constraints. Because of the complexity of solving two difficult decision problems at the same time, FMMRCPS becomes a combinatorial optimisation problem in complete NP-hardness (Hall 1997, Motwani 1997).

It is important to seek some practical ways of resolving FMMRCPS problems of realistic sizes with the objective of minimising the project completion time. Based on this viewpoint, I have spent significant time developing a number of approaches to solving any practically-sized FMMRCPS problems where an approximately optimal schedule can be obtained efficiently and effectively.

## 4.5 Concluding Remarks

This chapter has mainly addressed FMMRCPS with the objective of minimising the project completion time. FMMRCPS raised here has the significance of practical value, being applicable to most possible circumstances that exist in realistic projects. Time uncertainty is often intrinsic in many real projects. Furthermore, many activities of a project may be performed in several executive modes, rather than only single mode. All these issues have made for resolving scheduling with great complexity as shown in the mathematical formulation.

Although an FMMRCPS problem can be formulated as a 0-1 fuzzy programming model, this method may have great limitation in its ability to tackle FMMRCPS problems of any realistic sizes. Due to such limitations, a number of approaches have been developed in resolving practically sized FMMRCPS problems where a schedule approximating an optimal solution can be found in an efficient and effective way.

# Chapter 5

# Fuzzy Forward and Backward Scheduling in Project Scheduling

## 5.1 Introduction

This chapter focuses on *the fuzzy scheduling mechanism* that is incorporated in both a heuristically rule-based and four metaheuristic approaches I develop, for dealing with the single objective of minimising project completion time in FMMRCPS. Also another approach I developed for resolving multiple objectives in FMMRCPS is one in which *the fuzzy scheduling mechanism* is not employed. This will be presented in Chapter 6. A heuristically rule-based approach comprises both *a set of heuristic rules* and *the fuzzy scheduling mechanism* whereas the four metaheuristic approaches are commonly made up of both *the perturbation algorithm* and *the fuzzy scheduling mechanism*. Therefore, *the fuzzy scheduling mechanism* is the one basic component embedded in those approaches. Either *a set of heuristic rules* or *the perturbation algorithm* functions to decide on what activities become schedulable and what mode is assigned to these activities. *The fuzzy scheduling mechanism* is used to schedule the activities when they and their corresponding modes are determined by means of either *a set of heuristic rules* or *the perturbation algorithm*.

The basic idea of *the fuzzy scheduling mechanism* is similar to the parallel scheduling scheme used for deterministic data. The detail of the parallel scheduling scheme has been presented in Sections 3.5.1.2 and 3.5.2.2. *Fuzzy scheduling mechanism* has two scheduling mechanisms, fuzzy forward and backward scheduling. Fuzzy forward or backward scheduling operates in stage-wise fashion to schedule a number of the activities under the resources constraints in the forward or backward direction, thus extending a partial fuzzy schedule in each stage in the chosen direction until all the activities in a project have been scheduled. In *fuzzy scheduling* *mechanism*, both fuzzy forward and backward scheduling generates two different complete fuzzy schedules at a time, keeping a better solution and discarding a worse one.

This chapter first introduces the notation that will be used to describe *fuzzy scheduling mechanism*, and secondly, presents both fuzzy forward and backward scheduling for FMMRCPS in Sections 5.3 and 5.4. The chapter ends with concluding remarks.

## 5.2 Notation

*Fuzzy scheduling mechanism* acts to create a feasible schedule by step-wise extension of a partial fuzzy schedule when a set of eligible activities and their corresponding modes are determined by applying a *perturbation algorithm*. Such a mechanism is incorporated in each of these approaches. To clearly present both fuzzy forward and backward scheduling, the definitions of four disjointed sets and the notation used in fuzzy scheduling schemes, are explained as follows.

**Definition 1:** *Decision Set -- $D(\tilde{t}_n)$*

The decision set $D(\tilde{t}_n)$ is the set where the activities of a project are considered schedulable at fuzzy time point $\tilde{t}_n$ in stage $n$ of *the fuzzy scheduling mechanism*. In fuzzy forward scheduling, the activities are eligible to be put in $D(\tilde{t}_n)$ at fuzzy time point $\tilde{t}_n$ of stage $n$ only when their predecessors have been completed whilst resources are also available to them at that scheduled time $\tilde{t}_n$, or when these activities have no predecessors at the beginning of the scheduling (time point 0 of stage 1). The condition for placing activities into $D(\tilde{t}_n)$ in fuzzy backward scheduling, is that these activities have no successors at the initially scheduled time point 0 of stage 1, or their successors have been completed and current resources are available to them at time point $\tilde{t}_n$ of stage $n$.

**Definition 2:** *Active Set* — $A(\tilde{t}_n)$

The active set $A(\tilde{t}_n)$ in either fuzzy forward or backward scheduling is the set in which are placed those activities that start to be scheduled or have been scheduled but not yet been completed, at fuzzy scheduled time point $\tilde{t}_n$ of stage *n*.

**Definition 3:** *Complete Set* — $C(\tilde{t}_n)$

The complete set $C(\tilde{t}_n)$ is a specific set for storing the activities that have been completed at fuzzy scheduled time point $\tilde{t}_n$ of stage *n* regardless of whether the fuzzy forward or backward scheduling is used. If some activities are just completed at fuzzy time point $\tilde{t}_n$ of stage *n*, they are then moved from $A(\tilde{t}_n)$ to $C(\tilde{t}_n)$.

**Definition 4:** *Remaining Set* — $R(\tilde{t}_n)$

The remaining set $R(\tilde{t}_n)$ is to store those activities that are not scheduled at fuzzy scheduled time point $\tilde{t}_n$ of stage *n* in either fuzzy forward or backward scheduling. There are two possibilities for their remaining in $R(\tilde{t}_n)$ of stage *n*. One is that these activities have not been selected for $D(\tilde{t}_n)$ in stage *n*. The other is that even though some activities have been placed in $D(\tilde{t}_n)$ in the beginning of stage *n*, they are not scheduled at scheduled time $\tilde{t}_n$ because the rest of the resources are not adequate, and they now are returned to $R(\tilde{t}_n)$ before moving to the next stage $n + 1$ in *the fuzzy scheduling mechanism*.

**Definition 5:** *A Partial & Complete Schedule*

A partial fuzzy schedule is a schedule where only a subset of *J* activities of a project have been scheduled at fuzzy scheduled time point $\tilde{t}_n$ either in fuzzy forward or backward scheduling. In the next stage $n + 1$ of a partial schedule, fuzzy scheduled time point $\tilde{t}_{n+1}$ is equal to the earliest finish time of an activity among these in $A(\tilde{t}_n)$ in the previous stage *n*. A number of stages of partial schedules are carried on for, at most, *J* stages until all activities are moved to $C(\tilde{t}_n)$. When all the activities of a project have been scheduled, such a schedule is called a complete schedule, and the fuzzy project completion time should be the same as the fuzzy finish time of the activity that is finished last.

**Definition 6:** *Scheme A*

In *the fuzzy scheduling mechanism*, if resources are not available to one selected eligible activity in $D(\tilde{t}_n)$, the other eligible activity in $D(\tilde{t}_n)$ is not to be scheduled even though there are enough resources for that activity. The rest of the non-scheduled activities in $D(\tilde{t}_n)$ must be moved back to $R(\tilde{t}_n)$, then fuzzy forward or backward scheduling goes to the next stage $n + 1$.

**Definition 7:** *Scheme B*

If one selected activity from $D(\tilde{t}_n)$ cannot be scheduled due to insufficient resources, the next highest priority activity, for which there are sufficient resources currently available, is chosen as a candidate for scheduling. If there are no such activities, the rest of the activities in $D(\tilde{t}_n)$ should be returned to $R(\tilde{t}_n)$ before moving to the next stage of the fuzzy scheduling.

**Remark 1:** *Fuzzy Arithmetic in the Fuzzy Scheduling Mechanism*

Triangular and trapezoidal fuzzy numbers, as well as crisp numbers, are involved in FMMRCPS. If these three types of numbers are used at the same time, all numbers must be converted to trapezoidal fuzzy numbers. However, if triangular fuzzy numbers and crisp numbers exist in FMMRCPS, all these numbers should be consistently changed to the triangular fuzzy numbers. Fuzzy arithmetic addition and subtraction operate on fuzzy numbers during stage-wise scheduling in *the fuzzy scheduling mechanism*, by formulae (2.23), (2.24), (2.30) and (2.31). An example for fuzzy number conversions is given in Example 5.1, and the operation of the fuzzy arithmetic applied to fuzzy forward and backward scheduling is demonstrated in Examples 5.2 and 5.3 respectively.

**Example 5.1** *Fuzzy Number Conversions for Consistency*

(1) Different kinds of numbers are all converted into trapezoidal fuzzy numbers for consistency in fuzzy arithmetic because triangular and trapezoidal fuzzy numbers, as well as crisp numbers, are involved in scheduling. Let $\tilde{d}_1$, $\tilde{d}_2$ and $d_3$ be duration times of activities 1, 2 and 3 of a project given as.

$$\tilde{d}_1 = (4, 6, 8),\ \tilde{d}_2 = (5, 8, 10, 13) \text{ and } d_3 = 6$$

They should be converted as

$$\tilde{d}_1 = (4, 6, 6, 8),\ \tilde{d}_2 = (5, 8, 10, 13) \text{ and } d_3 = (6, 6, 6, 6)$$

(2) All numbers are changed to triangular fuzzy numbers because triangular fuzzy numbers as well as crisp numbers are used in scheduling. Let $\tilde{d}_1$ and $d_2$ be duration times of activities 1 and 2 given as

$$\tilde{d}_1 = (3, 5, 7) \text{ and } d_2 = 6$$

They are all converted as

$$\tilde{d}_1 = (3, 5, 7),\ d_2 = (6, 6, 6)$$

**Example 5.2** *The Calculation of Fuzzy Start and Finish Times in Forward Schedule*

To compute the fuzzy start $\widetilde{ST}_j$ and finish $\widetilde{FT}_j$ times of activity *j* in fuzzy forward scheduling at fuzzy scheduled time point $\tilde{t}_3$, fuzzy duration times $\tilde{d}_4$ and $\tilde{d}_6$ of eligible activities 4 and 6 from $D(\tilde{t}_3)$, are given as

$$\tilde{t}_3 = (4, 6, 9, 13),\ \tilde{d}_4 = (2, 4, 6, 8),\ \tilde{d}_6 = (1, 3, 6, 9)$$

Their fuzzy start and finish times can be calculated as

$$\widetilde{ST}_4 = (4, 6, 9, 13),\ \widetilde{FT}_4 = \widetilde{ST}_4 + \tilde{d}_4 = (4, 6, 9, 13) + (2, 4, 6, 8) = (6, 10, 15, 21)$$

$$\widetilde{ST}_6 = (4, 6, 9, 13),\quad \widetilde{FT}_6 = \widetilde{ST}_6 + \tilde{d}_6 = (4, 6, 9, 13) + (1, 3, 6, 9) = (5, 9, 15, 22)$$

**Example 5.3** *The Calculation of Fuzzy Start and Finish Times in Backward Schedule*

To calculate the fuzzy start $\widetilde{ST}_j$ and finish $\widetilde{FT}_j$ times of activity $j$ in fuzzy backward scheduling, both fuzzy addition and subtraction are involved in fuzzy arithmetic. Fuzzy duration times $\tilde{d}_3$ and $\tilde{d}_4$ of activities 3 and 4 are given. After the scheduling, fuzzy project completion, $PC\tilde{T}$ and the dummy finish time[1] of activity $j$, $D\tilde{F}_j$, are obtained as

$$\tilde{d}_3 = (0, 2, 5, 7) \text{ and } \tilde{d}_4 = (2, 3, 4, 6)$$

$$PC\tilde{T} = (19, 24, 26, 29),\ D\tilde{F}_3 = (7, 9, 13, 14),\ D\tilde{F}_4 = (6, 9, 15, 17)$$

The start and finish times are computed as

$$\widetilde{ST}_3 = PC\tilde{T} - D\tilde{F}_3 = (19, 24, 26, 29) - (7, 9, 13, 14) = (5, 11, 17, 22)$$

$$\widetilde{FT}_3 = \widetilde{ST}_3 + \tilde{d}_3 = (5, 11, 17, 22) + (0, 2, 5, 7) = (5, 13, 22, 29)$$

$$\widetilde{ST}_4 = PC\tilde{T} - D\tilde{F}_4 = (19, 24, 26, 29) - (8, 9, 17, 19) = (0, 7, 17, 21)$$

$$\widetilde{FT}_4 = \widetilde{ST}_4 + \tilde{d}_4 = (0, 7, 17, 21) + (2, 3, 4, 6) = (2, 10, 21, 27)$$

**Note 1:** The dummy finish time is the finish time of an activity obtained through fuzzy backward scheduling and the real finish time should be calculated as above.

**Remark 2:** *Fuzzy Ranking in the Fuzzy Scheduling Mechanism*

In *the fuzzy scheduling mechanism*, fuzzy numbers in $A(\tilde{t}_{n-1})$ of the previous stage $n$-1, are required to be compared to find the smallest fuzzy number in $A(\tilde{t}_{n-1})$ as the fuzzy scheduled time point $\tilde{t}_n$ in this new scheduling stage $n$. Furthermore, after having terminated the fuzzy scheduling, fuzzy ranking is also required to find the biggest fuzzy number in $A(\tilde{t}_n)$ as the fuzzy project completion time by using Formulae (2.37), (2.38) and (2.41). I have reformulated Formula (2.37) into Formulae (2.39) and (2.40) for particularly ranking triangular and trapezoidal fuzzy numbers. The reason for applying these formulae to fuzzy ranking has been addressed in

Section 2.5. The triangular and trapezoidal fuzzy number comparisons are given in Examples 5.4 and 5.5 respectively.

**Example 5.4** *Triangular Fuzzy Number Comparison*

(1) To determine the new scheduled time point $\tilde{t}_n$ in stage $n$, it is necessary to find the earliest fuzzy finish time of the activities in $A(\tilde{t}_{n-1})$. Let $F\tilde{T}_3$, $F\tilde{T}_4$ and $F\tilde{T}_6$ be fuzzy finish times of activities 3, 4 and 6 in $A(\tilde{t}_{n-1})$ given as

$$F\tilde{T}_3 = (4, 8, 10),\ F\tilde{T}_4 = (5, 7, 10),\ F\tilde{T}_6 = (5, 8, 11)$$

Having applied Formulae (2.39) and (2.38), the fuzzy ranking indices, $R(F\tilde{T}_j)$ for the fuzzy finish times of these three activities, are obtained as

$$R(F\tilde{T}_3) = 7.35,\ R(F\tilde{T}_4) = 7.34,\ R(F\tilde{T}_6) = 8.02$$

By fuzzy ranking rules (2.41), $F\tilde{T}_4$ is chosen as the new scheduled time point $\tilde{t}_n$.

(2) To acquire the fuzzy project complete time, it is necessary to find the fuzzy finish time at which the last activity or activities in the schedule are finished. It is assumed that activities 8, 10 and 11 are placed in $A(\tilde{t}_n)$ of the final scheduling stage $n$ and their finish times, $F\tilde{T}_8$, $F\tilde{T}_{10}$ and $F\tilde{T}_{11}$ are given as

$$F\tilde{T}_8 = (41, 52, 61),\ F\tilde{T}_{10} = (39, 51, 60),\ F\tilde{T}_{11} = (38, 52, 61)$$

Having applied formulae (2.39) and (2.38), the ranking indices $R(F\tilde{T}_j)$ of fuzzy finish times of these activities are shown as

$$R(F\tilde{T}_8) = 51.34,\ R(F\tilde{T}_{10}) = 50.00,\ R(F\tilde{T}_{11}) = 51.02$$

Based on the ranking rules (2.41), the biggest fuzzy number is $F\tilde{T}_8$. Therefore, activity 8 is finished last in the whole project, and the fuzzy project completion time, $PC\tilde{T}$ should be as

$$PC\tilde{T} = F\tilde{T}_8 = (41, 52, 61)$$

**Example 5.5** *Trapezoidal Fuzzy Number Comparison*

(1) To determine the fuzzy scheduled time point $\tilde{t}_n$, fuzzy ranking is required to find the earliest fuzzy finish times among these activities in $A(\tilde{t}_{n-1})$. The fuzzy finish times $F\tilde{T}_2$, $F\tilde{T}_3$ and $F\tilde{T}_5$ of activities 2, 3 and 5 in $A(\tilde{t}_{n-1})$ are given as

$$F\tilde{T}_2 = (6, 9, 13, 15),\ F\tilde{T}_3 = (7, 9, 12, 14),\ F\tilde{T}_5 = (7, 10, 13, 15)$$

Using formulae (2.40) and (2.38) for trapezoidal fuzzy numbers, the ranking indices for these three fuzzy numbers $R(F\tilde{T}_2)$, $R(F\tilde{T}_3)$ and $R(F\tilde{T}_5)$, are computed as

$$R(F\tilde{T}_2) = 10.73,\ R(F\tilde{T}_3) = 10.51,\ R(F\tilde{T}_5) = 11.81$$

Having employed the ranking rules (2.41), the smallest fuzzy number is determined to be

$$F\tilde{T}_3 = (7, 9, 12, 14)\ \therefore \text{ the scheduled time point } \tilde{t}_n = F\tilde{T}_3$$

(2) To obtain fuzzy project completion time, fuzzy finish times of the activities in $A(\tilde{t}_n)$ of the final stage will be compared to find the last finish time of the last activity or activities in a project. The finish time of the last activity or activities will be designated as the fuzzy completion time of this project. The finish times of activities 7, 9, 12 in $A(\tilde{t}_n)$ of the last stage of scheduling are give as

$$F\tilde{T}_7 = (35, 41, 47, 54),\ F\tilde{T}_9 = (32, 42, 47, 55),\ F\tilde{T}_{12} = (31, 36, 47, 55)$$

Through fuzzy ranking formulae (2.40) and (2.38) for trapezoidal fuzzy numbers, the ranking indices $R(F\tilde{T}_7)$, $R(F\tilde{T}_9)$ and $R(F\tilde{T}_{12})$ for fuzzy finish times of activities 7, 9 and 12 are obtained as

$$R(F\tilde{T}_7) = 44.30,\ R(F\tilde{T}_9) = 44.15,\ R(F\tilde{T}_{12}) = 42.35$$

Having applied the fuzzy ranking rules (2.41), apparently $R(F\tilde{T}_7) = 44.30$ is the biggest fuzzy number among these three fuzzy number.

$$\therefore PC\tilde{T} = F\tilde{T}_7 = (35, 41, 47, 54)$$

The following notation is used for describing *the fuzzy scheduling mechanism* as one component in each of my approaches I have developed.

| | |
|---|---|
| $r$ | = one of renewable resources; |
| $R$ | = a set of renewable resources consumed in a project; |
| $K_r$ | = the total availability of renewable resource $r$; |
| $\pi K_r$ | = the left-over capacity of renewable resource $r$; |
| $k_{jmr}$ | = the consumptive requirement of resource $r$ by performing activity $j$ in mode $m$; |
| $\widetilde{d}_{jm}$ | = the fuzzy duration time by performing activity $j$ in mode $m$; |
| $S\widetilde{T}_j$ | = the fuzzy start time of activity $j$; |
| $F\widetilde{T}_j$ | = the fuzzy finish time of activity $j$; |
| $P_j$ | = a set of immediate predecessors of activity $j$; |
| $S_j$ | = a set of immediate successors of activity $j$; |
| $D\widetilde{S}_j$ | = the dummy fuzzy start time of activity $j$ obtained through fuzzy backward scheduling, that is not the real start time of activity $j$; |
| $D\widetilde{F}_j$ | = the dummy fuzzy finish time of activity $j$ made by fuzzy backward scheduling, that is not the real fuzzy finish time; |
| $PC\widetilde{T}$ | = fuzzy project completion time. |

The above four disjointed sets and the notation defined will be used in presenting both fuzzy forward and backward scheduling in the following two sections.

## 5.3 Fuzzy Forward Scheduling

In fuzzy forward scheduling, eligible activities from $D(\tilde{t}_n)$ are scheduled in the forward direction by stage-wise movement. There are, at most, $J$ stages in scheduling if there are $J$ activities in a project. The process of fuzzy forward scheduling is given in Figure 5.1. As shown in Figure 5.1, an inner loop is embedded in the outer loop. The outer loop controls the whole process of the fuzzy scheduling to ensure all the activities of a project will be scheduled in order to gain a complete schedule, and the inner loop specifically deals with only one stage of the scheduling, in which, some eligible activities are selected from $D(\tilde{t}_n)$ for scheduling. To clearly address the specific purpose of each block of fuzzy forward scheduling, four main blocks are marked with four steps underlined in bold in Figure 5.1.

**Step (a)** — *Determine the Scheduled Time Point* $\tilde{t}_n$ *and* $D(\tilde{t}_n)$ *in The New Stage*

The main purpose of the block of step (a) is to compute the new fuzzy scheduled time point $\tilde{t}_n$, and to set up $D(\tilde{t}_n)$ for this new scheduling stage $n$. The new fuzzy scheduled time point $\tilde{t}_n$ chosen in stage $n$ should be same as the earliest fuzzy finish time of activity $j$ among $A(\tilde{t}_{n-1})$ in the previous stage $n$-1. To find out this earliest fuzzy finish time, the fuzzy finish times of all the activities in $A(\tilde{t}_{n-1})$ will be ranked through the fuzzy raking method given in Remark 2. Any activities finished at this earliest finish time (which becomes the fuzzy scheduled time point $\tilde{t}_n$ in stage $n$), are all removed from $A(\tilde{t}_{n-1})$ to $C(\tilde{t}_n)$, so that $A(\tilde{t}_n)$ is temporarily updated in stage $n$, and, at the same time, some resources taken by these activities are released. To set up new $D(\tilde{t}_n)$ in this stage of the scheduling, activities eligible for placement in $D(\tilde{t}_n)$ are those whose predecessors have been finished and whose resource requirements do not exceed the currently available resources. $R(\tilde{t}_n)$ is also temporarily updated by removing those activities that have been placed in $D(\tilde{t}_n)$.

```
INITIALISATION:
$n := 1$, $\tilde{t}_n := 0$, $\pi K_r := K_r\ \forall r \in R$, $D(\tilde{t}_n) := \{j \mid P_j = \phi, j \in J\}$,
$A(\tilde{t}_n) = C(\tilde{t}_n) := \phi$, $R(\tilde{t}_n) := \{j \mid P_j \neq \phi, j \in J\}$   goto Step (b)
DO WHILE $n < J \cap R(\tilde{t}_n) \neq \phi$
    Step (a)
        $\tilde{t}_n := \min\{\widetilde{FT}_j \mid j \in A(\tilde{t}_{n-1})\}$;
        $A(\tilde{t}_n) := A(\tilde{t}_{n-1}) \setminus \{j \mid j \in A(\tilde{t}_{n-1}) \cap \widetilde{FT}_j = \tilde{t}_n\}$;
        $C(\tilde{t}_n) := C(\tilde{t}_{n-1}) \vee \{j \mid \widetilde{FT}_j = \tilde{t}_n\ \forall j \in A(\tilde{t}_{n-1})\}$;
        $\pi K_r = \pi K_r + \sum_{j \in A(\tilde{t}_{n-1}) \cap \widetilde{FT}_j = \tilde{t}_n} k_{jmr}\ \forall r \in R$;
        $D(\tilde{t}_n) := \{j \in J \setminus [A(\tilde{t}_n) \bigcup C(\tilde{t}_n)] \mid P_j \subseteq C(\tilde{t}_n) \cap k_{jmr} \leq \pi K_r\ \forall r \in R\}$;
        $R(\tilde{t}_n) := \{J\} \setminus D(\tilde{t}_n) \cup A(\tilde{t}_n) \cup C(\tilde{t}_n)$
    Step (b)
        Priority ordering and mode assignments in $D(\tilde{t}_n)$
        DO WHILE $D(\tilde{t}_n) \neq \phi$
            Select $j \in D(\tilde{t}_n)$
            FOR Scheme A
              IF $k_{jmr} \geq \pi K_r\ \forall r \in R$
                  goto Step (c)
                ELSE
                  goto Step (d)
              ENDIF
            FOR Scheme B
              IF $k_{jmr} \geq \pi K_r\ \forall r \in R$
                  goto Step (c)
                ELSEIF the next highest priority $j^* \cap k_{j^*mr} \geq \pi K_r\ \forall r \in R$
                  goto Step (c)
                  ELSE
                      goto Step (d)
              ENDIF
    Step (c)
        $\widetilde{ST}_j := \tilde{t}_n$; $\widetilde{FT}_j := \widetilde{ST}_j + \tilde{d}_{jm}$; $D(\tilde{t}_n) := D(\tilde{t}_n) \setminus j$; $A(\tilde{t}_n) := A(\tilde{t}_n) \vee j$;
        $\pi K_r = \pi K_r - \sum_j k_{jmr}\ \forall r \in R$
        ENDLOOP
Step (d)
    $D(\tilde{t}_n) := \phi$, and update $R(\tilde{t}_n)$;
    $n := n + 1$
ENDLOOP
```

Figure 5.1 Fuzzy forward scheduling

**Step (b)** — *Select A Number of Eligible Activities from* $D(\tilde{t}_n)$

The block function of step (b) is to select a number of activities from $D(\tilde{t}_n)$ in the manner of scheme A or B. Schemes A and B have been introduced in Definitions 6 and 7 of Section 5.2. For scheme A, one activity is selected at a time if this activity satisfies two conditions simultaneously: (a) having the highest priority in $D(\tilde{t}_n)$ at a time, and (b) resources available to this activity. Such a process is repeated until no more eligible activities in $D(\tilde{t}_n)$ satisfy these two conditions. For scheme B, if the resource requirements of a selected activity with the highest priority in $D(\tilde{t}_n)$ exceed the currently available resource, the next highest priority activity with resource consumption not surpassing currently available resources is selected until $D(\tilde{t}_n)$ is empty or no more activities with resource requirements are less than currently available resources.

**Step (c)** — *Schedule A Selected Activity at A Time*

In step (c), one activity selected at a time is scheduled and its start and finish times are calculated as shown in Remark 1, then this activity is placed in $A(\tilde{t}_n)$ and resources used by this activity are reduced from currently total available resources. This process is repeated until there are no more selected activities from $D(\tilde{t}_n)$ in this stage.

**Step (d)** — *Update* $D(\tilde{t}_n)$ *and* $R(\tilde{t}_n)$, *and Get Ready for the Next Stage*

The function of step (d) is to move the remaining activities from $D(\tilde{t}_n)$ to $R(\tilde{t}_n)$ when no more remaining activities in $D(\tilde{t}_n)$ are eligible for selection. At this point, the stage counter $n$ is incremented by one ($n := n + 1$) to terminate the inner loop for this stage and to get ready for the next stage of the fuzzy scheduling.

In what presented above, the functions of the four main blocks for fuzzy forward scheduling have been explained individually. To better understand how fuzzy forward scheduling operates as a whole, the whole process of the scheduling is presented below.

Before starting fuzzy forward scheduling, initialisation is made for variables and parameters. In the beginning of the scheduling, 1 and 0 are first assigned to stage $n$ and the scheduled time point $\widetilde{t}_n$ respectively. Of course, currently available resources are intact and equal to the total available resources initially provided for a project, and the activities initially in $D(\widetilde{t}_1)$ are only those that have no predecessors. $A(\widetilde{t}_1)$ and $C(\widetilde{t}_1)$ are empty sets in the beginning of stage 1, and $R(\widetilde{t}_1)$ contains all activities except for those in $D(\widetilde{t}_1)$. After having been initialised, those activities stored in $D(\widetilde{t}_1)$ will be selected and processed in step (b) until no eligible activity remains in $D(\widetilde{t}_1)$ or no more activities are left in $D(\widetilde{t}_1)$. Then the next stage, $n$ starts with step (a) at the beginning of the outer loop if, and only if, the number of stage $n$ is less than the number of activities $J$ ($n < J$) and if $R(\widetilde{t}_n)$ is not empty ($R(\widetilde{t}_n) \neq \phi$). The new fuzzy scheduled time point $\widetilde{t}_n$ will be finalised through ranking of fuzzy finish times of all the activities in $A(\widetilde{t}_{n-1})$ for the previous stage $n$-1. Then $D(\widetilde{t}_n)$ is determined so as to be able to move to step (b) to schedule eligible activities from $D(\widetilde{t}_n)$ through either schemes A or B or both. $D(\widetilde{t}_n)$, $R(\widetilde{t}_n)$ and the stage counter $n$ are updated If there are no more eligible activities in $D(\widetilde{t}_n)$ or no more activities in $D(\widetilde{t}_n)$. If $R(\widetilde{t}_n)$ is empty, the whole process of the fuzzy scheduling is terminated.

At the completion of fuzzy forward scheduling, the fuzzy project completion time can be obtained by determining the time at which the last activity or activities are finished in a project. That is, the largest fuzzy number among these fuzzy finish times of the activities in $A(\widetilde{t}_n)$ of the last stage $n$ can be determined by applying the fuzzy ranking method. The detail of fuzzy ranking has been given in Remark 2. Therefore, the fuzzy project completion time can be mathematically expressed as

$$PC\widetilde{T} = \max\{F\widetilde{T}_j \mid j \in A(\widetilde{t}_n) \wedge R(\widetilde{t}_n) = \phi\} \tag{5.1}$$

## 5.4 Fuzzy Backward Scheduling

Fuzzy backward scheduling is the other way of scheduling in *the fuzzy scheduling mechanism*. The backward scheduling operates in exactly the opposite direction of fuzzy forward scheduling. Fuzzy backward scheduling starts from the end of activities in the project network until the beginning of activities is reached. That is, the activities without their successors are scheduled first in the backward direction moving toward the beginning of activities by stage-wise scheduling. Figure 5.2 presents the process of fuzzy backward scheduling. As shown in Figure 5.2, there are two loops in fuzzy backward scheduling. The outer loop governs the whole process of the scheduling to ensure that all the activities in a project are included from the beginning to the final stage of scheduling, whereas the inner loop manages a specific stage of scheduling while $D(\tilde{t}_n)$ has been determined in that stage. The function of each step in fuzzy backward scheduling is presented below.

**Step (a)** — *Initialisation*

Step (a) is the initialisation of the beginning of scheduling: assigning the initial values 1 and 0 to the stage counter $n$ and the beginning of scheduled time point $\tilde{t}_n$ respectively; locating the activities that have no successors into $D(\tilde{t}_1)$; ensuring $A(\tilde{t}_1)$ and $C(\tilde{t}_1)$ are initially null; placing all the activities of a project into $R(\tilde{t}_1)$ except for those already in $D(\tilde{t}_1)$. Then eligible activities in $D(\tilde{t}_1)$ are selected and processed in step (c).

**Step (b)** — *Determine* $\tilde{t}_n$ *&* $D(\tilde{t}_n)$

Step (b) deals with the new stage of scheduling when the previous stage is completed. Step (b) starts with the outer loop if, and only if, two following loop conditions are satisfied: (1) the stage counter $n$ must be less than the number of activities contained in a project $J$, and (2) $D(\tilde{t}_n)$ is not empty. The new fuzzy scheduled time point $\tilde{t}_n$ is set the same time as the earliest dummy fuzzy finish time of the activities in $A(\tilde{t}_{n-1})$ of the previous stage $n$-1. Both dummy start $D\tilde{S}_j$ and finish $D\tilde{F}_j$ times of activity $j$ are not actual start $S\tilde{T}_j$ and finish $F\tilde{T}_j$ times of activity $j$.

They are only obtained by the backward scheduling calculations. However, they are used to determine the new scheduled time point $\tilde{t}_n$ in fuzzy backward scheduling. When the fuzzy scheduled time point $\tilde{t}_n$ is determined, the activities finished at that time point $\tilde{t}_n$ in the backward scheduling are removed to $C(\tilde{t}_n)$, and $A(\tilde{t}_n)$ is updated. The resources used by these activities are released. The new decision set $D(\tilde{t}_n)$ is set up for the step (c) and the remaining set $R(\tilde{t}_n)$ is updated.

**Step (c)** — *Get Eligible Activities from* $D(\tilde{t}_n)$

Step (c) begins with the inner loop. It commences if, and only if, the loop condition of the decision set $D(\tilde{t}_n)$ is not empty. There are two schemes, A and B, available for selection by the user in fuzzy backward scheduling. The details about Schemes A and B have been addressed in Definitions 6 and 7 of Section 5.2 as well as presented in step (b) of fuzzy forward scheduling in Section 5.3. If an eligible activity exists in $D(\tilde{t}_n)$, this selected activity goes to step (d) for scheduling. If there are no more eligible activities in $D(\tilde{t}_n)$, the process skips from the inner loop to step (e).

**Step (d)** — *Schedule the Eligible Activity Selected in Step (c)*

Step (d) schedules one specific activity at a time that has been selected from $D(\tilde{t}_n)$ in step (c). The dummy fuzzy start time $D\tilde{S}_j$ of the eligible activity $j$ is assigned as the fuzzy scheduled time point $\tilde{t}_n$ in this stage $n$, and the dummy fuzzy finish time $D\tilde{F}_j$ of this activity is its dummy start time $D\tilde{S}_j$ plus its fuzzy duration time $\tilde{d}_{jm}$ performed in mode $m$. Then this activity is moved from $D(\tilde{t}_n)$ to $A(\tilde{t}_n)$. The currently available resources are also updated by reducing the amount of the resources consumed by this activity. This loop then cycles to step (c) if an eligible activity in $D(\tilde{t}_n)$ exists.

```
Step (a) (Initialisation)
  $n := 1$, $\tilde{t}_n := 0$, $\pi K_r := K_r \ \forall r \in R$, $D(\tilde{t}_n) := \{j \mid S_j = \phi, j \in J\}$,
   $A(\tilde{t}_n) = C(\tilde{t}_n) := \phi$, $R(\tilde{t}_n) := \{j \mid S_j \neq \phi, j \in J\}$  goto Step (c)
Step (b) (Determining $\tilde{t}_n$ & $D(\tilde{t}_n)$)
  DO WHILE $n < J \cap R(\tilde{t}_n) \neq \phi$
        $\tilde{t}_n := \min\{D\tilde{F}_j \mid j \in A(\tilde{t}_{n-1})\}$;
        $A(\tilde{t}_n) := A(\tilde{t}_{n-1}) \setminus \{j \mid j \in A(\tilde{t}_{n-1}) \cap D\tilde{F}_j = \tilde{t}_n\}$;
        $C(\tilde{t}_n) := C(\tilde{t}_{n-1}) \vee \{j \mid D\tilde{F}_j = \tilde{t}_n \ \forall j \in A(\tilde{t}_{n-1})\}$;
        $\pi K_r = \pi K_r + \sum_{j \in A(\tilde{t}_{n-1}) \cap D\tilde{F}_j = \tilde{t}_n} k_{jmr} \ \forall r \in R$;
        $D(\tilde{t}_n) := \{j \in J \setminus [A(\tilde{t}_n) \cup C(\tilde{t}_n)] \mid P_j \subseteq C(\tilde{t}_n) \cap k_{jmr} \leq \pi K_r \ \forall r \in R\}$;
        $R(\tilde{t}_n) := \{J\} \setminus D(\tilde{t}_n) \cup A(\tilde{t}_n) \cup C(\tilde{t}_n)$
    Step (c) (Part of the inner loop for selecting eligible activities from $D(\tilde{t}_n)$)
        Priority ordering and mode assignments in $D(\tilde{t}_n)$
        DO WHILE $D(\tilde{t}_n) \neq \phi$
            Select $j \in D(\tilde{t}_n)$
            FOR Scheme A
              IF $k_{jmr} \geq \pi K_r \ \forall r \in R$
                  goto Step (d)
                ELSE
                  goto Step (e)
              ENDIF
            FOR Scheme B
              IF $k_{jmr} \geq \pi K_r \ \forall r \in R$
                  goto Step (d)
                ELSEIF the next highest priority $j^* \cap k_{j^*mr} \geq \pi K_r \ \forall r \in R$
                  goto Step (d)
                  ELSE
                      goto Step (e)
              ENDIF
        Step (d) (Scheduled the activity selected in step (c))
            $D\tilde{S}_j := \tilde{t}_n$; $D\tilde{F}_j := D\tilde{S}_j + \tilde{d}_{jm}$; $D(\tilde{t}_n) := D(\tilde{t}_n) \setminus j$; $A(\tilde{t}_n) := A(\tilde{t}_n) \vee j$;
            $\pi K_r = \pi K_r - \sum_j k_{jmr} \ \forall r \in R$
        ENDLOOP
  Step (e) (Update information if there are no more eligible activities in this stage)
     $D(\tilde{t}_n) := \phi$, and update $R(\tilde{t}_n)$;
     $n := n + 1$
  ENDLOOP
```

Figure 5.2 Fuzzy backward scheduling

**Step (e)** — *Update* $D(\tilde{t}_n)$, $R(\tilde{t}_n)$ *and Stage Counter n*

Step (e) updates relevant information after it skips out from the inner loop when there are no more eligible activities in $D(\tilde{t}_n)$. The rest of activities in $D(\tilde{t}_n)$ are removed to $R(\tilde{t}_n)$ so as to vacate $D(\tilde{t}_n)$ and the stage counter is increased by 1. Now the process is ready to move to the next stage if some activities still remain in $R(\tilde{t}_n)$. Otherwise, fuzzy backward scheduling is terminated.

Once the whole process of fuzzy backward scheduling is completed, the fuzzy project completion time can be obtained as the fuzzy time at which the last activity or activities are finished in the backward scheduling. That is, the fuzzy project completion time equals the biggest fuzzy number among these dummy fuzzy finish times of the activities in $C(\tilde{t}_n)$ of the final scheduling stage. The detail of fuzzy number comparison has been addressed in Remark 2 of Section 5.2 and also presented in Examples 5.4 and 5.5 of Section 5.2 respectively. The fuzzy project completion time can be mathematically defined as

$$PC\tilde{T} = \max\{D\tilde{T}_j \mid j \in A(\tilde{t}_n) \wedge R(\tilde{t}_n) = \phi\} \tag{5.2}$$

Because only dummy fuzzy start and finish times of the activities in a project are acquired through the backward scheduling, rather than their real start and finish times, some extra conversion work is required to change the dummy times of the activities to their actual fuzzy start and finish times. The fuzzy start time of an activity should be the fuzzy project completion time minus its dummy fuzzy finish time, and the fuzzy finish time of an activity should be equated with its fuzzy start time plus its fuzzy duration time performed in mode *m*. Both the fuzzy start and finish times of activity *j* can be respectively expressed as

$$S\tilde{T}_j = PC\tilde{T} - D\tilde{F}_j \quad \forall j \in J \tag{5.3}$$

$$F\tilde{T}_j = S\tilde{T}_j + d_{jm} \quad \forall j \in J,\ m \in M \tag{5.4}$$

## 5.5 Concluding Remarks

*The fuzzy scheduling mechanism* is a common component incorporated in both a fuzzy heuristically rule-based approach and four fuzzy metaheuristic approaches I developed for resolving FMMRCPS with the single objective of minimising the project completion time, in an environment where activity times are fuzzy. The mechanism schedules eligible activities with their associated modes both of which are decided by *a set of heuristic rules* or a *perturbation algorithm* in a fuzzy heuristically rule-based or a fuzzy metaheuristic approach.

The *fuzzy scheduling mechanism* contains both fuzzy forward and backward scheduling, dealing with both crisp and fuzzy duration times of activities involved in scheduling. Fuzzy forward scheduling is a stage-wise way of scheduling eligible activities with their associated modes in the forward direction whereas fuzzy backward scheduling starts with the end of activities and their associated modes, gradually scheduling eligible activities toward the beginning of activities in the project network.

The advantage of employing both fuzzy forward and backward scheduling is that it generates two different schedules in which the same conditions are applied at a time. It gives a great opportunity for generating a diversity of fuzzy scheduling results by running the *fuzzy scheduling mechanism* a number of times.

# Chapter 6

# Fuzzy Project Scheduling with Multiple Objectives

## 6.1 Introduction

Project scheduling sometimes requires considering several performance measures such as time, budget and resources simultaneously to meet specific objectives. Due to such concerns, Slowinski (1981) proposed multiple objectives for RCPS. Norbis and Smith (1988) adopted a hierarchical ranking method for RCPS to rank the importance of each individual objective among three objectives: (a) project completion time, (b) due-date, and (c) resource utilisation. Deckro and Hebert (1990), and Rajagopal *et al.* (1997) used a goal programming model to compromise the conflicting objectives of: (a) project completion time, and (b) project cost, in a RCPS problem. Li and Willis (1992), and Ulusoy and Özdamar (1994) developed iterative algorithms with dispatching rules for minimising the project completion time and maximising NPV simultaneously. Davis *et al.* (1992) proposed an iterative decision support approach for the DM to examine explicitly the minimisation of two objectives (that is, the project completion time and the average resource usage), during the whole scheduling process. Viana and de Sousa (2000) developed SA and Tabu search for minimising three objectives: (a) the project completion time, (b) the mean weighted lateness of activities, and (c) the sum of the violation of resource availability, to be compromised in MMRCPS. All these approaches were developed on the basis of deterministic information.

In the real world, many realistic projects are often confronted with uncertainty because of lack of sufficient information on activity duration times in a project. Hapke *et al.* (1999) proposed the Pareto SA to FMMRCPS for minimising three objectives simultaneously, including (a) the project completion time, (b) the total project cost, and (c) the average deviation from the average resource usage. I developed a fuzzy goal programming model for solving two commonly used objectives, including (a) project completion time, and (b) project cost, using the $\alpha$-cut concept in the environment of fuzzy activity times (Willis *et al.* 1999).

This chapter focuses on FMMRCPS with multiple objectives as one part of my research I have undertaken. Section 6.2 addresses the generalised characteristics of FMMRCPS for dealing with multiple objectives. Section 6.3 presents a real case of dredge repair that is a typical FMMRCPS problem faced with two objectives of (a) the project completion time, and (b) the project cost. Section 6.4 presents an interactive fuzzy goal programming for multiple objectives in FMMRCPS. Section 6.5 is the result analysis for dredge repair schedule. The chapter ends with conclusion remarks.

## 6.2 Multiple Objectives in FMMRCPS

As reviewed in Section 6.1, project scheduling has traditionally focused on the consideration of a single objective only. There is not much research work reported on multiple objectives both in RCPS and MMRCPS (Özdamar and Ulusoy 1995, Kolisch and Padman 2001). Surprisingly, research work on solving multiple objectives in fuzzy project scheduling is very scant with only two publications found so far in the recent literature. This is the reason, that motivates me to undertake research on multiple objectives in FMMRCPS although my main research focus is on the development of efficient and effective approaches to FMMRCPS with a single objective of minimising project completion time.

FMMRCPS with multiobjective optimisation inherently exists in real-word applications. It represents generalised circumstances of project scheduling with the consideration of multiple objectives where activities can be performed in one of several executive modes under the fuzzy environment. In some situations, a project schedule may need to take several requirements into consideration simultaneously, such as finance, time, and some organisation or industry-specific objectives. A major concern in FMMRCPS with multiple objectives is that all these objectives often conflict with each other. The optimal solution for multiple objectives is not so obvious because there is no single solution that is the best for all the criteria or objectives, but there exists a set of nondominated solutions when considering all the objectives. However, the final solution should be chosen based on the specification of specific requirements in a particular circumstance.

In FMMRCPS with multiple objectives, a project consists of $J$ activities ($j$ =1, 2,..., $J$) under precedence relationships and resource constraints. Activity $j$ may be performed in one mode $m$ of $M_j$ modes ($m$ = 1, 2,..., $M_j$) with its fuzzy duration time $\tilde{d}_{jm}$ and its corresponding resource requirements ( $k_{jmr}, \forall r \in R$ ). The project scheduling may be concerned with a number of objectives ($o$ = 1, 2,..., $O$) simultaneously, requiring that a best compromised solution can be found. Let $\tilde{T}$ be heuristic fuzzy completion time of a project, determined by adding fuzzy duration times of all activities of a project in mode 1 beforehand, and be expressed as:

$$\tilde{T} = \sum_{j=1}^{J} d_{j1} \tag{6.1}$$

Let $\tilde{t}$ be the fuzzy time, and $x_{jm\tilde{t}}$ be a binary variable, defined as:

$$x_{jm\tilde{t}} = \begin{cases} 1 & \text{if activity } j \text{ is performed in mode } m \text{ at time } \tilde{t} \\ 0 & \text{otherwise} \end{cases} \tag{6.2}$$

FMMRCPS with multiple objectives can be formulated as

$$\min/\max \underset{\substack{m\in M_j \\ \tilde{t}\in\tilde{T}}}{\overset{j\in J}{f_o}} (x_{jm\tilde{t}}), \quad o = 1,2,...,O \tag{6.3}$$

Let $exk_r$ be a number of extra units required for resource $r$ ($r \in R$). The constraints of FMMRCPS can be expressed as:

Subject to $$\sum_{m=1}^{M_j} \sum_{\tilde{t}=EF\tilde{T}_j}^{LF\tilde{T}_j} x_{jm\tilde{t}} = 1, \quad j = 1,2,\dots,J \tag{6.4}$$

$$\sum_{m=1}^{M_j} \sum_{\tilde{t}=EF\tilde{T}_j}^{LF\tilde{T}_j} (\tilde{t} - \tilde{d}_{jm}) \times x_{jm\tilde{t}} - \sum_{m=1}^{M_h} \sum_{\tilde{t}=EF\tilde{T}_h}^{LF\tilde{T}_h} \tilde{t} \times x_{hm\tilde{t}} \geq 0, \quad j = 1,2,\dots,J \qquad 6.5)$$

$$\sum_{j=1}^{J} \sum_{m=1}^{M_j} k_{jmr} \sum_{s=\tilde{t}}^{\tilde{t}+\tilde{d}_{jm}-1} x_{jms} - exk_r \leq K_r, \quad r = 1,2,\dots,R;\ \tilde{t} = 1,\dots,\tilde{T} \tag{6.6}$$

Constraints (6.4) ensure that each activity is processed only once in scheduling. Constraints (6.5) guarantee that precedence relationships are met, and constraints (6.6) indicate the required amount of resource $r$ consumed by activities minus the extra amount of resource $r$ does not exceed the available resources provided for a project. The above mathematical description represents generalised characteristics for solving multiobjective FMMRCPS problems. However, the problem formulation or mathematical representation of FMMRCPS with multiple objectives may be different from the above generalised mathematical description because of its specific characteristics and particular requirements.

## 6.3 Dredge Repair Scheduling Problem

### 6.3.1 Dredge Breakdown Repair

The schedule for repair of a dredge after breakdown is often required to consider timing and the budget of repair simultaneously. However, the priority given to the repair time and cost often vary depending on the current circumstances of the dredging company faced with the dredging workload of the waterway and the number of reserve dredges at the time. It needs to produce a better schedule of a dredge repair that satisfies both objectives for the requirements of a particular environment.

Large quantities of silt from the upper reaches of Yangtze River pour downstream and settle in the waterways between the river and the sea, seriously affecting normal waterborne transportation all year around. As a consequence, a number of dredges are required to work in different sites day and night for waterway maintenance (Pan 1997). Because of their heavy-duty work, dredges sometimes suddenly break down and require repair. The schedule of repair should satisfy both objectives of time and cost, set by the management in the company.

Commonly a dredge requiring repair will be examined by experts to determine the number of activities (or jobs) and their duration times. Some components of the dredge may be functional, but may be in poor condition, and will also be repaired or replaced. However, the time required for each job or activity is very difficult to determine precisely because the extent of damage to each components has to be assessed subjectively by experts through their examination. Therefore, the time for each activity is often decided subjectively by the management of the company. For these reasons, it is evident that uncertainty inherently exists in duration times of activities in dredge repair.

The project of dredge repair is a typical FMMRCPS problem with multiobjective concerns where two conflicting objectives (that is, the project completion time and the repair cost) are taken into consideration simultaneously during its scheduling. A method needs to be found to obtain a better schedule in terms of multiobjective optimisation.

## 6.3.2 Problem Formulation

When a dredge breaks down, three departments, dredging, engine and deck of the dredge need to be serviced. The service schedule of the dredge is often required to meet two criteria or objectives: time and cost. These two objectives are main concerns for a dredging company. They conflict with each other and are incommensurable. To arrive at an acceptable compromised solution, fuzzy goal programming is proposed in this research.

The objective function of minimising the project completion time is equal to the last activity finished at time $F\widetilde{T}_n$. Therefore, the objective function can be simply expressed as:

$$\text{Min} f_1 = F\widetilde{T}_n \tag{6.7}$$

In minimising the repair cost (project cost), the amount of resources used by each activity and the penalty for delaying activity $j$ by $l_j$ days are considered as the two main factors. Therefore, the cost of the project is the sum of all the activity costs incurred by using resources plus the penalties caused through delays in activities. Let $k_{jmr}$ and $k^{*}_{jmr}$ be the amounts of resource consumed by activity $j$ performed in mode $m$ from both the available and the extra resource $r$ respectively. $C_r$ and $C^{*}_r$ are the unit costs of the available and the extra resource $r$ respectively. $dp_j$ is the daily penalty rate after due days $\widetilde{l}_i$ in activity $j$. $x_{jm\widetilde{t}}$ is a 0-1 binary variable and defined in formula (6.2). Therefore, the objective function for the project cost can be represented as:

$$\text{Min} f_2 = \sum_{r=1}^{R}\sum_{j=1}^{J}\sum_{m=1}^{M_j} (k_{jmr} \times C_r + k^{*}_{jmr} \times C^{*}_r) \sum_{s=\widetilde{t}}^{\widetilde{t}+d_{jm}-1} x_{jms} + dp_j \times \widetilde{l}_j \tag{6.8}$$

Constraints are set up for the dredge repair to guarantee a feasible schedule that satisfies the two objectives. In addition to the constraints that will be addressed below, there are constraints (6.4) and (6.5) stated in Section 6.2.

The $\alpha$-cut concept is used here to express the DM's view on the minimum degree of acceptance when assessing duration time $\widetilde{d}^{\alpha}_{jm}$ of activity $j$ for executive mode $m$. The level of $\alpha$-cut reflects the DM's confidence in the assessment. The details of the concept of $\alpha$-cut have been given in Sections 2.2.4 and 4.4.2. The fuzzy duration $\widetilde{d}_j$ to be used will be the fuzzy duration time $\widetilde{d}^{\alpha}_{jm}$ in the level $\alpha$-cut considered by the DM's view on the minimum degree of acceptance, and given as:

$$\widetilde{d}_j = \widetilde{d}^{\alpha}_{jm} \qquad j = 1, 2, \ldots, J \tag{6.9}$$

The start time $\widetilde{ST}_j$ and finish time $\widetilde{FT}_j$ of an activity $j$ should satisfy the following expression:

$$\widetilde{ST}_j + \widetilde{d}_j = \widetilde{FT}_j \qquad j = 1,2, \ldots, J \tag{6.10}$$

Let activities $i$ and $j$ be a pair $(i, j)$ of immediate precedence. The precedence between activities should have the following relationships:

$$\widetilde{FT}_i \le \widetilde{ST}_j; \ \ \forall (i,j) : i << j \tag{6.11}$$

Let $rk_{jr}$ and $rk^*_{jr}$ be respectively the percentages of the required amount from both the available and extra resource $r$ consumed by activity $j$. $rk_{jr}$ and $rk^*_{jr}$ should always be equal to 1 while activity $j$ is performed, and given as:

$$rk_{jr} + rk^*_{jr} = 1 \tag{6.12}$$

Let $u_{jmr}$ represent the amount of resource $r$ consumed by activity $j$ in performing mode $m$. Therefore, the amount of resource $r$ required ($u_{jmr}$) by activity $j$ will be the amounts consumed both from the available ($k_{jmr}$) and the extra resource ($k^*_{jmr}$), and given as

$$u_{jmr} = k_{jmr} + k^*_{jmr} \tag{6.13}$$

Let $S_{\widetilde{t}}$ denote the set of activities scheduled at the time $\widetilde{t}$ . To ensure that the amount of resource $r$, allocated to activities if available does not exceed the total available resource originally provided for the project, the following constraint should be satisfied at each time $\widetilde{t}$ .

$$\sum_{j \in S_{\widetilde{t}}} \sum_{m=1}^{M_j} x_{jm\widetilde{t}} \times u_{jmr} \times rk_{jr} \le K_r \quad \forall \widetilde{t} \in \widetilde{T} \tag{6.14}$$

The amount of resource $r$, consumed from the availability of the resource by activity $j$ can be expressed as

$$k_{jmr} = u_{jmr} \times rk_{jr} \tag{6.15}$$

The amount of resource $r$, consumed from the extra resource by activity $j$ is given as

$$k^{*}_{jmr} = u_{jmr} \times rk^{*}_{jr} \tag{6.16}$$

## 6.4 Interactive Fuzzy Goal Programming

The project completion time and the cost are universally regarded as the two main concerns in project scheduling, and will influence the ultimate success or failure of a project. In this case study, they are considered as the project scheduling objectives in the model. But these two objectives conflict with each other and are incommensurable. To deal with FMMRCPS with multiple objectives, fuzzy goal programming (FGP) is developed to generate the schedule of dredge repair having these two objectives. This is because FGP has the capacity of dealing with multiobjective optimization problems to obtain an acceptable compromised solution under fuzzy environment.

Goal programming has been widely used in solving decision making problems with conflicting goals or objectives, but it requires a precise aspiration level for each goal or objective to be assigned. This is often difficult for the DM to do in most practical situations (Pan *et al.* 1998). To overcome this problem, in FGP, the aspiration level is a value range given by the DM to match the company's requirements in practical settings. This iterative FGP model allows the DM to modify and examine the result of the schedule under different objective settings.

As FMMRCPS is a very complex scheduling, there are too many constraints in the FGP model, thus making the solution procedure computationally inefficient in solving the problem in terms of computation. To reduce constraints in the model for improving its computational efficiency, the rules for mode assignment are constructed for supporting decisions on selecting activity modes. The mode will be decided by the rules before processing the interactive FGP model. Which mode will be given is based on current conditions of resource availabilities, the number of activities that are eligible to be processed at this time, and the critical path at this time. To better understand five rules for mode assignment, these rules can be expressed in plain English in Figure 6.1.

**Rules for Mode Selection**

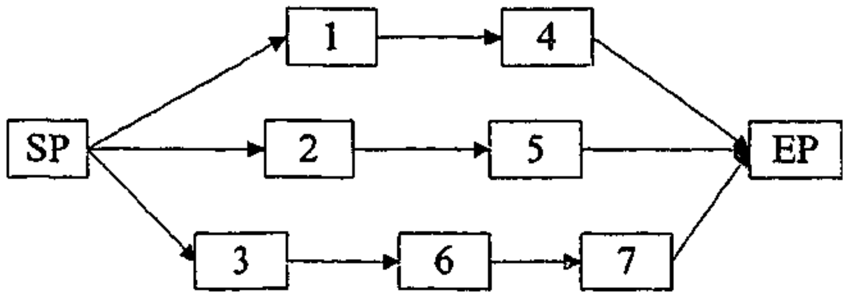
1. If only one activity is scheduled at a given time, then the mode with the shortest duration time is assigned to the activity;
2. If more than one activities are selected at a scheduled time, then modes with the shortest duration times are assigned to the activities on the critical path(s) first, and the non-critical activities are assigned to the modes with the higher level number;
3. If activities selected are not on critical path(s), modes are assigned to these activities to ensure that these activities finish at a similar time;
4. If there a number of activities on critical paths completing for resources, modes assigned to these activities ensure the paths are as closed as possible to minimise the completion time;
5. If there are sufficient resources, all selected activities are assigned to modes with the shortest duration times.

Figure 6.1 Rules for mode assignment expressed in plain English

The interactive FGP model I developed can be easily implemented through the DM's input of required information. This model is flexible, efficient and effective in obtaining the best compromised solution in realistic settings as the range of aspiration level and required information are easily given in the practical sense by the DM, to meet specific requirements for solving multiple objectives in FMMRCPS problems.

## 6.5 Result Analysis of Repair Schedule

The case study presented here is a typical dredge repair project when a dredge broke down on working sites of the waterway in Shanghai. In this project, several jobs or activities are carried on in three departments, mainly dredging, engine, and deck. Let SP and EP represent the start and end of a project of a dredge repair. The precedence network of the dredge repair project can be depicted in Figure 6.2.



Activity 1: clean dumps and dredging parts
Activity 2: clean the rust in decks and bottom
Activity 3: disassemble double engines
Activity 4: service dumps and dredging parts
Activity 5: paint body and decks and dry up
Activity 6: service engines and relative parts
Activity 7: assemble engines and test

Figure 6.2 Dredge repair network

As shown in Figure 6.2, the dredge repair project contains 7 activities. Due to their uncertainty, activity duration times under different modes are assessed by the DM, using triangular fuzzy numbers given in Table 6.1

Table 6.1 Fuzzy duration times of activities in different modes

| Activity Number | Fuzzy duration times of activities in their modes with different $\alpha$-cut | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | $\tilde{d}_{jm}^{0}$ | | | $\tilde{d}_{jm}^{0.5}$ | | | $\tilde{d}_{jm}^{0.8}$ | | |
| | Mode1 | Mode 2 | Mode 3 | Mode 1 | Mode 2 | Mode 3 | Mode 1 | Mode 2 | Mode 3 |
| 1 | (5,7,9) | (8,11,13) | (10,13,16) | [6,8] | [8.5,12] | [11.5,14.5] | [6.6,7.4] | [10.4,11.4] | [12.4,13.6] |
| 2 | (1,3,6) | (3,5,7) | (6,9,11) | [2,4.5] | [4,6] | [7.5,9.5] | [2.6,3.6] | [4.6,5.4] | [8.4,8.6] |
| 3 | (2,5,7) | (4,7,10) | (6,9,11) | [3.5,6] | [5.5,8.5] | [7.5,9.5] | [4.4,5.4] | [6.4,7.6] | [8.4,8.6] |
| 4 | (2,3,4) | (5,7,9) | (7,10,13) | [2.5,3.5] | [6,8] | [8.5,11.5] | [2.8,3.2] | [6.6,7.4] | [9.4,10.6] |
| 5 | (1,3,5) | (2,5,7) | (5,7,10) | [2,4] | [3.5,6] | [6,8.5] | [2.6,3.4] | [4.4,5.4] | [6.6,7.6] |
| 6 | (5,7,10) | (7,10,13) | (9,11,14) | [6,8.5] | [8.5,11.5] | [10,12.5] | [6.6,7.6] | [9.4,10.6] | [10.6,11.6] |
| 7 | (5,8,11) | (7,10,12) | (10,13,16) | [6.5,9.5] | [8.5,11] | [11.5,14.5] | [7.4,8.6] | [9.4,10.4] | [12.4,13.6] |

The resource requirements at each activity and the penalty payment for the delay of activity $j$ are given in Table 6.2. 5 units of resource 1 is available daily and its unit cost per day is \$10. 9 units of resource 2 are available with a daily cost of \$15 per unit. One extra unit of resources 1 and 2 to be supplied will cost \$20 and \$25 daily respectively.

Table 6.2 The resource requirements and penalty payments

| Activity No. | Mode 1 $(k_{j1r})$ | | Mode 2 $(k_{j2r})$ | | Mode 3 $(k_{j3r})$ | | Daily penalty |
|---|---|---|---|---|---|---|---|
| $(j)$ | $k_{j11}$ | $k_{j12}$ | $k_{j21}$ | $k_{j22}$ | $k_{j31}$ | $k_{j32}$ | $dp_j$ |
| 1 | 3 | 4 | 2 | 3 | 1 | 2 | 30 |
| 2 | 3 | 5 | 2 | 5 | 2 | 3 | 20 |
| 3 | 4 | 6 | 4 | 4 | 3 | 3 | 15 |
| 4 | 3 | 3 | 2 | 2 | 1 | 2 | 25 |
| 5 | 4 | 5 | 2 | 3 | 2 | 2 | 40 |
| 6 | 3 | 4 | 2 | 4 | 2 | 3 | 35 |
| 7 | 3 | 4 | 2 | 4 | 2 | 3 | 20 |

The range of two fuzzy goals for the objectives is given by the DM, based on the company's situations of the current workload and reserve dredges as well as the cost that will be provided by the state government. In this example, it is required that the project completion time will range from 10 to 22 days, and the corresponding rough project cost will range from \$1,900 to \$4,200.

Figure 3 shows the results of the project cost at different $\alpha$ levels. As in the beginning of the preparation for the dredge repair project, the goals for both time and cost are much vague during the DM's assessment. After the DM gathers more information on overall conditions of the company by investigations and communications with relevant departments, more confidence is achieved in evaluating the project completion time at certain $\alpha$ levels. Figure 6.3 shows the range of the budget of dredge repair at $\alpha$ levels of 0.5 and 0.8. In the case of $\alpha$ level being 0.8, the project will be completed between 16 and 19 days with the cost ranging from \$2,925 to \$3570. The $\alpha$-cut levels of 0.5 and 0.8 of $\alpha$-cut have often been used by the DM to assess the project completion time in practical applications.

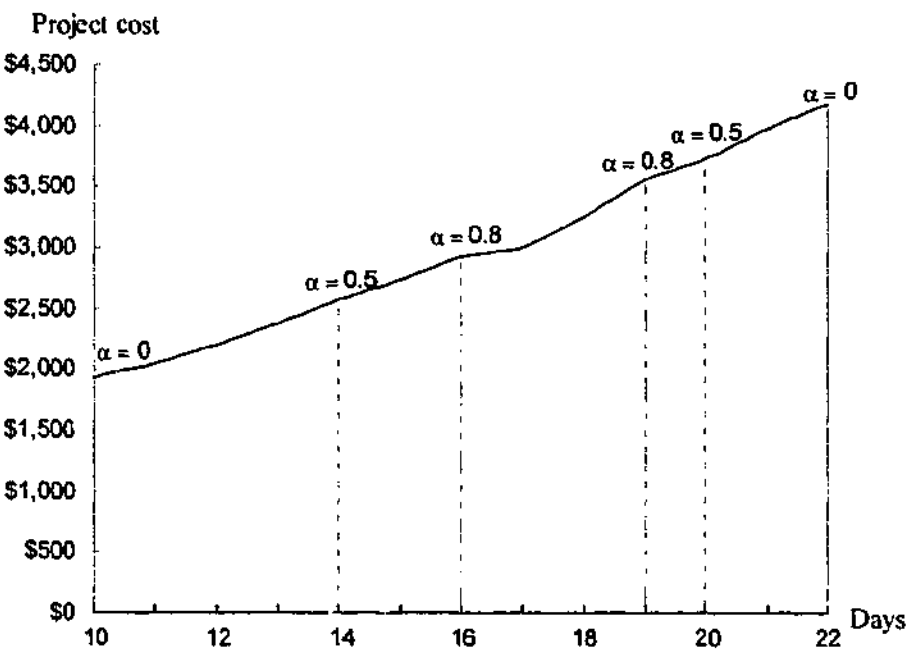


Figure 6.3 The range of the project cost in different α levels

## 6.6 Concluding Remarks

This chapter has discussed FMMRCPS with multiple objectives. In certain circumstances, fuzzy project scheduling is required to satisfy several criteria or objectives to meet the specific needs. These objectives are usually conflicting with each other and incommensurable.

Fuzzy goal programming models allows a loose aspiration level rather than an exact aspiration level to suit practical requirements under a fuzzy environment. In FMMRCPS, multiple modes make scheduling very complex and difficult to solve. To reduce complex constraints in the FGP model, and to speed up computation, a rule-based knowledge has been constructed to decide which mode will be performed for an activity based on the current partial schedule status, the current resource availabilities, and the current critical paths, preset by the experienced DM. This approach is called iterative fuzzy goal programming. This facilitates the implement of FGP in an efficient and effective way, and this model yields the best compromised solution among conflicting objectives.

The FGP model also provides the activities' performance and the states of resource usage in detail over the planing horizon of the project. The schedules indicate the maximum amount of extra resources required in a specified time period so as to preventing the project from being failed. This model gives the DM a significant prescheduling control over the planing horizon of the project under conditions of uncertainty. The model has general applications in multiobjective FMMRCPS.

# Chapter 7

# A Fuzzy Heuristic Approach to FMMRCPS

## 7.1 Introduction

In real-world applications, project scheduling is often inherently uncertain due to the lack of precise information on activity duration times. In addition, activities may have a number of performance options in practice. A realistic model of project scheduling covering all possible situations has been formulated in Chapter 4, referred to as FMMRCPS.

In FMMRCPS, two main issues need to be tackled: (a) uncertainty caused by vagueness, and (b) combinatorial complexity in project scheduling. To deal with such uncertainty, fuzzy set theory is used in FMMRCPS because it has proven to be an effective way of handling such vague information. To handle the computational complexity of multiple mode resource-constrained project scheduling, a heuristic approach is proposed. FMMRCPS is NP-hard in the strong sense, due to the complexity of its combinatorial nature. Any exact approaches may not be able to solve such a problem. However, heuristic approaches provide a simple and straightforward way of solving this problem (Kolisch 1996a, Demeulemeester and Herroelen 2002).

Drexl and Gruenewald (1993) developed a so-called weighted random selection technique for selecting activities and subsequently assigning modes to these selected activities for scheduling in a rule-based heuristic approach. Özdamar and Ulusoy (1994) selected activities and their respective modes using a local constrained based analysis at each decision point. Boctor (1996a) proposed the heuristic activity-mode combination for both choosing the activities and determining their mode from a set of activities. All theses heuristic approaches are based on deterministic data. However, no research on FMMRCPS using heuristic approaches has been reported in the literature. In what follows, I will present one of the several approaches developed in my PhD studies for FMMRCPS, using a fuzzy rule-based heuristic approach.

## 7.2 Fuzzy Heuristic Approach

A fuzzy heuristic approach proposed here is made up of three components: (a) fuzzy scheduling mechanism, (b) priority rules for priorities of activities, and (c) mode assignment policy. The function of the fuzzy scheduling mechanism is to schedule activities of a project either forwards or backwards in a stage-wise fashion under the resources constraints, when both priorities and modes of activities have been determined. The detail of the fuzzy scheduling mechanism has been presented in Chapter 5. Priority rules are employed to resolve the conflict of how to select from a set of eligible activities. The mode assignment policy serves as a means of assigning modes to activities selected from the set of eligible activities.

Priority rules have been discussed and tested by Davis and Patterson (1975), Cooper (1976), Alveres-Valdes and Tamari (1989), and Kolisch (1996a). There is no single priority rule that dominates others. It seems to be a good idea to apply a set of priority rules so as to give different priorities to eligible activities in each instance of fuzzy scheduling, thus generating a variety of schedules. In my fuzzy heuristic approach, any number of priority rules can be used. In the example of the case study presented in the next section, ten popular priority rules are employed, and these rules are concerned with different aspects of the critical path, resources, successors and the network structure. Figure 7.1 lists these 10 priority rules applied in the example.

In Figure 7.1, rules 1 to 5 are based on the critical path involving the fuzzy earliest start time ( $ES\widetilde{T}$ ) and earliest finish time ( $EF\widetilde{T}$ ), the fuzzy latest start time ( $LS\widetilde{T}$ ) and latest finish time ( $LF\widetilde{T}$ ), and the fuzzy slack time. The fuzzy earliest start time ( $ES\widetilde{T}_j$ ) of activity $j$ should be the same as the fuzzy finish time of its last immediate predecessor $i$ completed among its immediate predecessors $P_j$ by the forward pass, and is defined as

$$\widetilde{EST}_j = \max_{i \in P_j}\{\widetilde{EST}_i + \tilde{d}_{im}\} \tag{7.1}$$

1. Fuzzy early start time ($\widetilde{EST}$): $\min_{j\in D(\tilde{t}_n)} \widetilde{EST}_j$
2. Fuzzy early finish time ($\widetilde{EFT}$): $\min_{j\in D(\tilde{t}_n)} \widetilde{EFT}_j$
3. Fuzzy late start time ($\widetilde{LST}$): $\min_{j\in D(\tilde{t}_n)} \widetilde{LST}_j$
4. Fuzzy late finish time ($\widetilde{LFT}$): $\min_{j\in D(\tilde{t}_n)} \widetilde{LFT}_j$
5. Minimum fuzzy slack ($MFSLK$): $\min_{j\in D(\tilde{t}_n)} \widetilde{LST}_j - \widetilde{EST}_j$
6. Greatest fuzzy resource demand ($GFRD$): $\max_{j\in D(\tilde{t}_n)} \tilde{d}_{jm} \sum_{r\in R} k_{jmr}$
7. Shortest fuzzy processing time ($\widetilde{SPT}$): $\min_{j\in D(\tilde{t}_n)} \tilde{d}_{jm}$
8. Longest fuzzy processing time ($\widetilde{LPT}$): $\max_{j\in D(\tilde{t}_n)} \tilde{d}_{jm}$
9. Most immediate successors ($MIS$) $\max_{j\in D(\tilde{t}_n)} |S_j|$
10. Least immediate successors ($LIS$) $\min_{j\in D(\tilde{t}_n)} |S_j|$

Figure 7.1 A set of priority rules

The fuzzy latest finish time $\widetilde{LFT}_j$ is obtained through the backward pass and equals the fuzzy latest start time $\widetilde{LST}_i$ of the immediate successor $i$ started earliest among a set of immediate successors $S_j$ of activity $j$, and is expressed as

$$\widetilde{LFT}_j = \min_{i\in S_j}\{\widetilde{LST}_i\} \tag{7.2}$$

The fuzzy earlier finish time $\widetilde{EFT}_j$ of activity $j$ should be the smallest value of the fuzzy latest finish time $\widetilde{LFT}_i$ of one immediate successor $i$ minus the duration in mode $m$ of that immediate successor among a set of its immediate successors $S_j$, and represented as

$$EF\widetilde{T}_j = \min_{i \in S_j}\{LF\widetilde{T}_i - \widetilde{d}_{im}\} \tag{7.3}$$

The fuzzy slack time $FSLK_j$ of activity $j$ is the time allowing the maximum delay of activity $j$ on a path, which is the difference between the fuzzy latest start time $LS\widetilde{T}_j$ and earliest start time $ES\widetilde{T}_j$, obtained from both the forward and backward passes, and is computed as

$$FSLK_j = LS\widetilde{T}_j - ES\widetilde{T}_j \tag{7.4}$$

In Figure 7.1, rule 6 is based on the total amount of resources consumed by activity $j$ in the fuzzy duration performed in mode $m$, and this rule is similar to the rule presented in Section 3.5.2.4 for deterministic data. Rules 7 and 8 are concerned only with the fuzzy duration of activity $j$ itself performed in mode $m$, whereas rules 9 and 10 are based on the status of the project network, considering the number of immediate successors of activity $j$.

In Figure 7.1, 8 out of 10 priority rules are involved with fuzzy numbers. Both fuzzy arithmetic and fuzzy ranking are required in dealing with these 8 priority rules. The details of fuzzy arithmetic have been presented in Sections 2.4. It is important to note that some lower bounds of $LS\widetilde{T}_j$ and $LF\widetilde{T}_j$ may be negative when fuzzy subtractions are taken through the backward pass. For practical purposes, the negative numbers have no physical meaning, and these negative numbers should be changed to zero. After fuzzy priority values of activities are obtained by applying fuzzy arithmetic to these rules, the fuzzy priority values need to be compared to determine the sequence of small or large in value so as to decide preferences of these activities for scheduling. Although a number of fuzzy ranking methods have been proposed, the reason for only choosing Cheng (1998)'s method of distance means has been addressed in Section 2.5.

Mode assignment is an important decision for selected activities, because different modes assigned to the selected activities can greatly affect the results of the schedule, even for the same sequence of activities. To develop an effective policy for mode assignment, the policy I propose is concerned with the expected entire length of each path at scheduled time $\widetilde{t}_n$. That is, the policy for mode assignment ensures that modes assigned to activities on each path can be completed as soon as possible, and at the same time the fuzzy completion times of all paths are as close to each other as possible, making activities on each path be finished at almost the same time. Let $path_i$ be the $i^{th}$ path in the project network, and $\widetilde{L}_{path_i}$ be the expected entire length of the $i^{th}$ path in the time horizon. $C_{path_i}(\widetilde{t}_n)$ denotes a set of the activities on the $i^{th}$ path, completed at fuzzy scheduled time $\widetilde{t}_n$ of stage $n$ in the forward or backward scheduling. $J_{path_i}$ signifies all the activities on path $i$. Let $j^*_{path_i}(\widetilde{t}_n)$ be the selected activity on path $i$ at scheduled time $\widetilde{t}_n$. The expected entire length of path $i$ is the last activity $j$ finished on the path at fuzzy scheduled time $\widetilde{t}_n$ plus the sum of the fuzzy duration of selected activity $j^*_{path_i}(\widetilde{t}_n)$ assigned to its mode $m$ at time $\widetilde{t}_n$ and the average duration of all possible modes of the other activities that have not been scheduled on path $i$ at that time. Therefore, the expected entire length of path $i$ can be expressed as

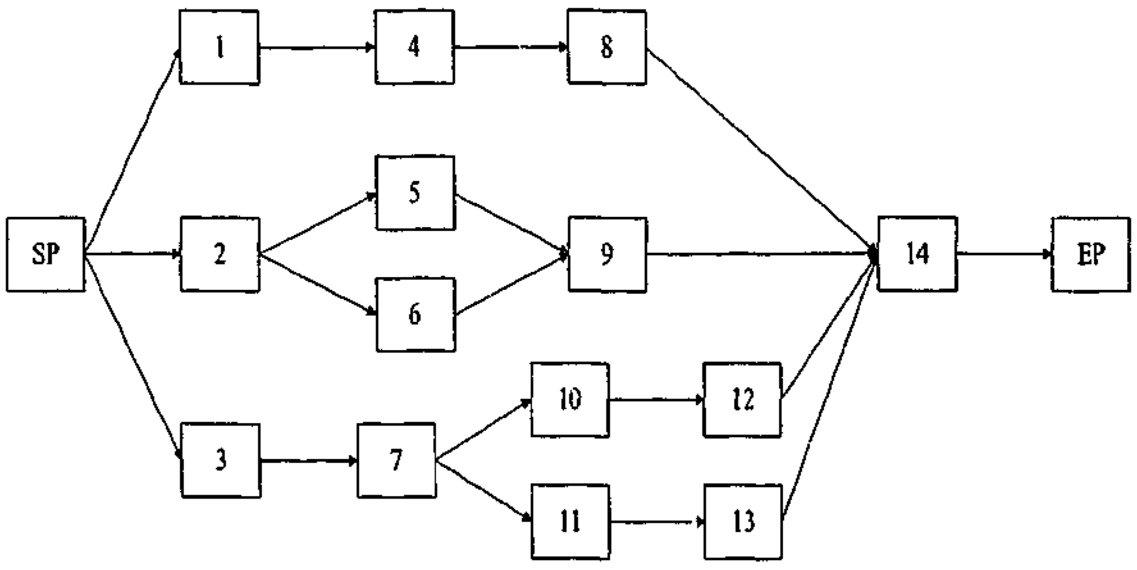
$$\widetilde{L}_{path_i} = \max\{F\widetilde{T}_j \mid j \in C_{path_i}(\widetilde{t}_n)\} + d_{\substack{j^*m \\ m \in M_{j^*}}} + \sum_{j \in J_{path_i} \setminus C_{path_i}(\widetilde{t}_n) \setminus j^*} \sum_{m=1}^{M_j} \frac{\widetilde{d}_{jm}}{M_j} \tag{7.1}$$

Let $N(path)$ be the number of paths in a project, and $(path_i, path_h)$ be a pair of any two paths among $N(path)$. The purpose of this policy is to assign a mode to selected activities at fuzzy scheduled time $\widetilde{t}_n$, so that the activities on each path can be finished with the minimum time in a similar length of each path. The policy for mode assignment can be defined as

$$\min_{(path_i, path_h) \in N(path)} \{ | \min \widetilde{L}_{path_i} - \min \widetilde{L}_{path_h} | \} \tag{7.2}$$

## 7.3 Dredge Overhaul Scheduling

A dredging company in Shanghai faces the busy and demanding tasks of continuously maintaining the required depth of the waterway in Shanghai. The company needs to dispatch dredges regularly to different sites on the waterway. Because of the nature of dredging, the machines and the components of a dredger are often subject to wear and tear. To reduce the chances of sudden breakdown, the dredger needs to be regularly maintained. However, the overhaul time varies depending on the mechanical condition of a dredge. The time possibly spent on an activity or a job has to be assessed by an expert or DM through the examination of the condition of a component or machine. In addition, the same activity or job may be processed with different durations while different work circumstances are exerted. A typical overhaul schedule presented here composes of 14 activities. The precedence relations among these activities are defined by the project network, as shown in Figure 1, where SP and EP are dummy activities representing the beginning and end of the overhaul project.



| | |
|---|---|
| Activity 1: dismantle engines | Activity 2: dismantle pumps |
| Activity 3: clean hopper | Activity 4: clean cylinders |
| Activity 5: clean pumps | Activity 6: clean pipes |
| Activity 7: inspect hydraulic system | Activity 8: assemble engines |
| Activity 9: assemble pumps | Activity 10: inspect doors of hopper |
| Activity 11: examine drags | Activity 12: check up general circuit |
| Activity 13: clean dragheads | Activity 14: test run |

Figure 7.2 The project network of overhaul of a dredge

In the overhaul project, the duration times of activities are often subjectively assessed by an expert or DM using triangular and trapezoidal fuzzy numbers, based on the worn or damaged extent of a component or machine or on current specific local conditions. In addition, activities often have a number of performance modes under different resource conditions in realistic overhaul projects. The expert or DM must estimate a number of durations of an activity performed under different possible resource conditions. Table 7.1 lists real duration times when performed with different resource availabilities. As shown in Table 7.1, a few activities have only one or two executive modes because of their particular characteristics in resource requirements. The duration times of activities 12 and 14 are crisp data because checking up general circuits and testing have very definite processing times and they can be completed in the required times. Due to the nature of the overhaul project, duration times of activities in the project involve both crisp and fuzzy numbers. Crisp numbers have to be converted to fuzzy number during fuzzy number operation. The details of the conversions have been presented in Remark 1 of Section 5.2. The evaluation of fuzzy duration times of activities has also been addressed in Sections 2.2.7 and 4.4.2.

Table 7.1 Fuzzy duration times of activities in different executive modes

| Activity number | Mode 1 Duration ($\tilde{d}_{j1}$) | Mode 2 Duration ($\tilde{d}_{j2}$) | Mode 3 Duration ($\tilde{d}_{j3}$) |
|---|---|---|---|
| 1 | (15, 18, 20, 25) | (19, 21, 25, 29) | (24, 27, 29, 34) |
| 2 | (7, 9, 11, 14) | (10, 12, 15, 20) | (13, 16, 18, 21) |
| 3 | (8, 10, 15) | (10, 13, 16) | (15, 18, 21) |
| 4 | (14, 16, 18, 21) | (17, 19, 21, 25) | (20, 25, 30, 36) |
| 5 | (10, 13, 16, 20) | (13, 15, 18, 22) | (16, 18, 21, 25) |
| 6 | (6, 9, 15) | (9, 12, 18) | (18, 22, 29) |
| 7 | (2, 4, 7, 9) | (5, 8, 10, 15) | (10, 13, 16, 20) |
| 8 | (18, 22, 25, 30) | (24, 27, 31, 35) | (28, 32, 35, 38) |
| 9 | (6, 10, 13, 17) | (10, 15, 20, 25) | (16, 20, 24, 30) |
| 10 | (1, 3, 5) | (5, 8, 12) | (11, 14, 18) |
| 11 | (8, 10, 12, 15) | (10, 13, 16, 19) | (12, 14, 17, 21) |
| 12 | 5 | 8 | Not available |
| 13 | (2, 4, 6) | (4, 6, 9) | Not available |
| 14 | 10 | Not available | Not available |

Three different kinds of renewable resources are required throughout the overhaul project. However, the amount of these resources is limited. In the overhaul project, 2 electricians (*Resource 1*), 8 mechanics (*Resource 2*), and 9 general workers (*Resource 3*) are available in each period. The resource requirements for different executive modes of activities are listed in Table 7.2. The sign of "×" in Table 7.2 indicates that no resources are involved in certain executive modes of activities as these modes are unavailable in the activities.

Table 7.2 Resource requirements of activities performed in different modes

| Activity number | Three kinds of resource requirements | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | Mode 1 | | | Mode 2 | | | Mode 3 | | |
| | $k_{j11}$ | $k_{j12}$ | $k_{j13}$ | $k_{j21}$ | $k_{j22}$ | $k_{j23}$ | $k_{j31}$ | $k_{j32}$ | $k_{j33}$ |
| 1 | 1 | 3 | 2 | 1 | 2 | 1 | 0 | 1 | 1 |
| 2 | 2 | 3 | 3 | 1 | 2 | 2 | 0 | 1 | 2 |
| 3 | 0 | 3 | 4 | 0 | 2 | 3 | 0 | 1 | 2 |
| 4 | 0 | 6 | 8 | 0 | 4 | 7 | 0 | 3 | 5 |
| 5 | 0 | 2 | 4 | 0 | 2 | 2 | 0 | 1 | 1 |
| 6 | 0 | 1 | 2 | 0 | 1 | 1 | 0 | 0 | 1 |
| 7 | 2 | 3 | 4 | 1 | 2 | 2 | 1 | 1 | 1 |
| 8 | 0 | 3 | 4 | 0 | 2 | 2 | 0 | 1 | 1 |
| 9 | 2 | 4 | 5 | 1 | 2 | 3 | 1 | 1 | 2 |
| 10 | 2 | 3 | 3 | 1 | 2 | 2 | 1 | 1 | 1 |
| 11 | 1 | 2 | 2 | 1 | 1 | 2 | 1 | 1 | 1 |
| 12 | 1 | 0 | 2 | 1 | 0 | 1 | × | × | × |
| 13 | 0 | 4 | 5 | 0 | 2 | 2 | × | × | × |
| 14 | 2 | 5 | 5 | × | × | × | × | × | × |

## 7.4 Result Analysis

The fuzzy heuristic approach I developed is implemented in an object-oriented programming language, VB6 with MS Access, in which Data Access Objects (DAO) are used to operate queries, update values and store relations among data tables. After required data are input to the program, priority values of activities are calculated from the priority rules, set in the program. Tables 7.3 lists fuzzy priority values obtained from some fuzzy priority rules out of these 10 rules.

Table 7.3 Data obtained for fuzzy priority rules

| Activity No. | $\widetilde{EST}_j$ | $\widetilde{EFT}_j$ | $\widetilde{LST}_j$ | $\widetilde{LFT}_j$ | $\widetilde{LST}_j - \widetilde{LFT}_j$ | $\tilde{d}_j \sum_{j \in R} k_{jr}$ |
|---|---|---|---|---|---|---|
| 1 | (0,0,0,0) | (15,18,20,25) | (0,0,0,0) | (15,18,20,25) | (0,0,0,0) | (120,144,160,200) |
| 2 | (0,0,0,0) | (10,12,15,20) | (0,9,28,53) | (3,24,40,63) | (0,9,28,53) | (60,72,90,120) |
| 3 | (0,0,0,0) | (8,10,10,15) | (0,27,41,66) | (14,37,51,74) | (0,27,41,66) | (32,40,40,60) |
| 4 | (15,18,20,25) | (35,43,50,61) | (15,18,20,25) | (35,43,50,61) | (0,0,0,0,) | (160,200,240,288) |
| 5 | (10,12,15,20) | (20,25,31,40) | (3,24,40,63) | (23,40,53,73) | (0,9,28,53) | (60,78,96,120) |
| 6 | (10,12,15,20) | (16,21,24,35) | (8,31,44,67) | (23,40,53,73) | (0,16,32,57) | (18,27,27,45) |
| 7 | (8,10,10,15) | (13,18,20,30) | (14,37,51,74) | (29,47,59,79) | (0,27,41,66) | (25,40,50,75) |
| 8 | (35,43,50,61) | (53,65,75,91) | (35,43,50,61) | (53,65,75,91) | (0,0,0,0) | (144,176,200,240) |
| 9 | (20,25,31,40) | (30,40,51,65) | (28,45,64,81) | (53,65,75,91) | (0,14,39,61) | (60,90,120,150) |
| 10 | (13,18,20,30) | (18,26,28,42) | (36,52,62,81) | (48,60,70,86) | (6,32,44,68) | (25,40,40,60) |
| 11 | (13,18,20,30) | (21,28,32,45) | (29,47,59,79) | (44,59,69,87) | (0,27,41,66) | (40,50,60,75) |
| 12 | (18,26,28,42) | (23,31,33,47) | (48,60,70,86) | (53,65,75,91) | (6,32,44,68) | (15,15,15,15) |
| 13 | (21,28,32,45) | (25,34,38,54) | (44,59,69,87) | (53,65,75,91) | (0,27,41,66) | (16,24,24,36) |
| 14 | (53,65,75,91) | (63,75,85,101) | (53,65,75,91) | (63,75,85,101) | (0,0,0,0) | (120,120,120,120) |

Table 7.4 Best fuzzy project completion times under each of 10 different rules

| Heuristic rule | Scheduling sequence | Fuzzy project completion time |
|---|---|---|
| 1. $\widetilde{EST}$ | 1(2), 2(1), 3(3), 7(2), 5(2), 6(1), 4(3), 10(2), 11(2), 12(1), 9(1), 13(2), 8(1), 14(1) | (68, 82, 96, 106) |
| 2. $\widetilde{EFT}$ | 3(1), 2(3), 1(2), 7(2), 6(1), 10(2), 5(3), 11(2), 12(2), 13(1), 9(1), 4(2), 8(1), 14(1) | (76, 94, 107, 145) |
| 3. $\widetilde{LST}$ | 1(1), 4(2), 2(3), 5(1), 3(1), 6(2), 7(1), 8(2), 11(2), 9(1), 10(1), 13(1), 12(1), 14(1) | (76, 94, 105, 133) |
| 4. $\widetilde{LFT}$ | 1(1), 2(1), 3(2), 4(2), 6(2), 5(1), 7(1), 11(1), 10(2), 8(1), 9(2), 13(1), 12(1), 14(1) | (61, 81, 93, 122) |
| 5. *MFSLK* | 1(1), 4(2), 8(1), 2(2), 5(3), 6(1), 9(2), 3(1), 7(2), 11(1), 13(1), 10(2), 12(2), 14(1) | (100, 125, 139, 178) |
| 6. *GFRD* | 1(2), 4(1), 8(3), 2(1), 5(1), 6(1), 9(2), 3(2), 7(3), 11(1), 10(1), 13(1), 12(1), 14(1) | (94, 118, 132, 170) |
| 7. $\widetilde{SPT}$ | 3(1), 2(2), 1(1), 7(2), 6(1), 5(1), 4(3), 10(2), 11(1), 9(2), 8(1), 12(1), 13(2), 14(1) | (64, 78, 89, 111) |
| 8. $\widetilde{LPT}$ | 1(2), 4(1), 8(2), 2(2), 5(1), 6(3), 3(1), 9(1), 7(1), 11(1), 10(2), 13(1), 12(1), 14(1) | (90, 112, 129, 165) |
| 9. *MIS* | 1(1), 3(2), 4(1), 8(2), 2(1), 5(2), 6(1), 9(1), 7(1), 11(2), 10(2), 13(1), 12(1), 14(1) | (87, 107, 121, 151) |
| 10. *LIS* | 2(1), 3(1), 1(1), 7(2), 4(1), 5(2), 6(1), 10(1), 11(1), 9(1), 8(1), 12(1), 13(2), 14(1) | (69, 84, 94, 116) |

As shown in Table 7.3, some activities in the decision set $D(\tilde{t}_n)$ have the same priority values when a specific priority rule is applied. In such cases, the program will automatically assign random values to these activities at the break-even priority values. In fuzzy scheduling, once an activity is selected according to the priority values from the decision set $D(\widetilde{t}_n)$ at scheduled time $\widetilde{t}_n$ of stage $n$, a specific mode is determined for the activity subsequently, based on the mode assignment policy expressed in Formula (7.2). In the overhaul project, 24 different feasible schedules are generated by these 10 priority rules. Table 7.4 lists 10 shortest project completion times of each individual rule from these 24 feasible schedules. How to compare these fuzzy project completion times has been presented in Remark 2 of Section 5.2.

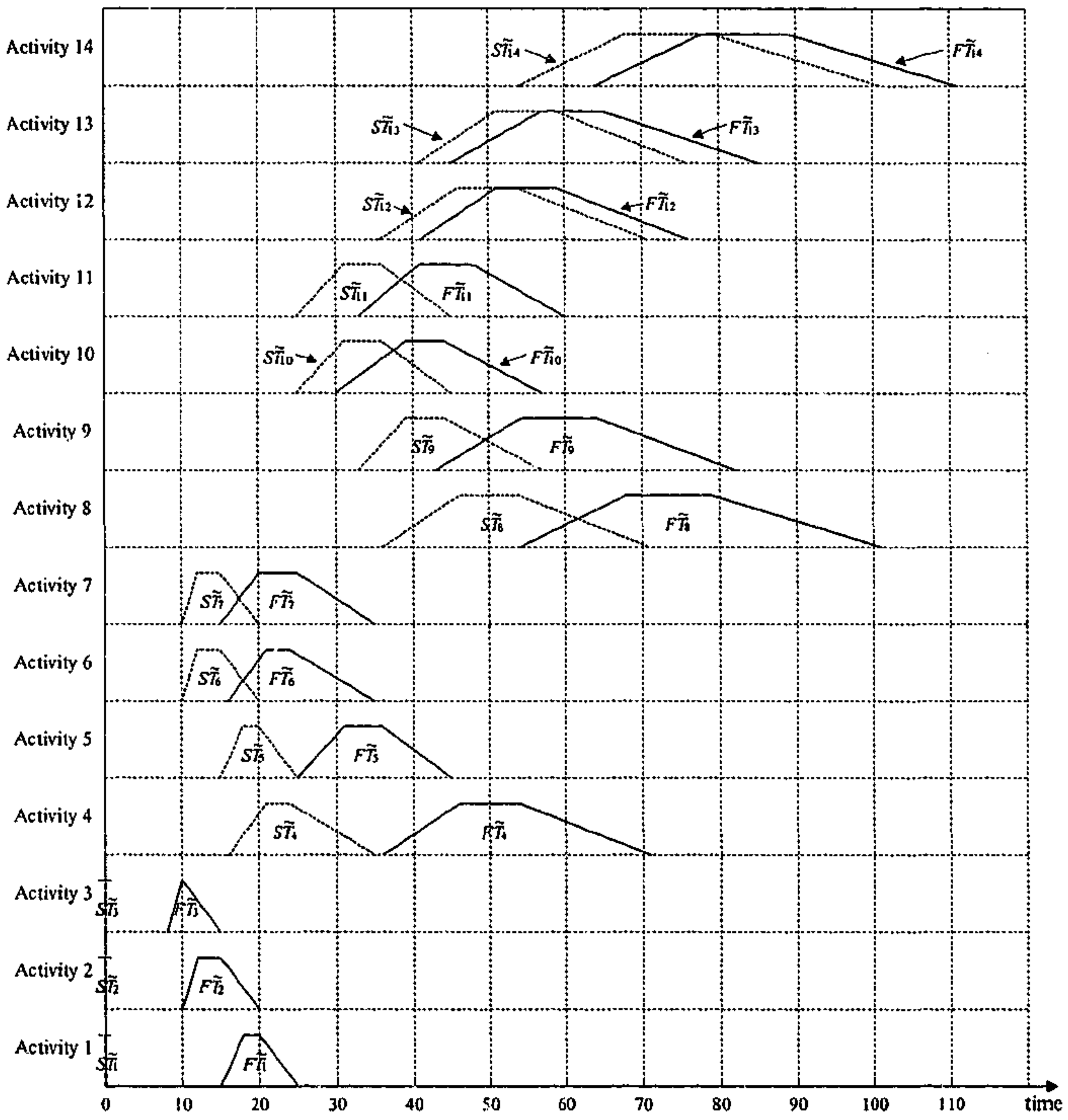


Figure 7.3 The best schedule of the overhaul project in $SP\widetilde{T}$

The scheduling sequence of all activities in the overhaul project are listed in the 2nd column of Table 7.4, where the number in front of parentheses presents the activity number and the following number in parentheses indicates the mode number performed for this activity. In this particular overhaul scheduling, the $ES\tilde{T}$, $LF\tilde{T}$, $SP\tilde{T}$ and $LIS$ rules produce better results than the others, and the fuzzy values of the project completion time, using these four rules, are quite close. But when applying the fuzzy number ranking method, the shortest project completion time is gained by the $SP\tilde{T}$ rule. Figure 7.3 shows each activity start and finish time obtained under the $SP\tilde{T}$ rule. It provides the DM with a reasonable schedule result given the vague information of activity duration times, and with detailed information about every moment of the overhaul performance.

## 7.5 Concluding Remarks

FMMRCPS is complex scheduling due to its NP-hardness and uncertainty. This combinatorial nature makes such scheduling problems difficult to handle if any exact algorithms are to be proposed. Heuristic approaches appear to be attractive since they give reasonably practical solutions in a simple and fast way. However, traditional heuristic approaches suffer from the major shortcoming of being unable to handle fuzziness.

This chapter has presented a heuristic approach that incorporates fuzzy set theory to model the uncertain activity duration times for resolving FMMRCPS problems. To handle multiple mode project scheduling, a set of priority rules are employed for deciding priorities for eligible activities, and at the same time the mode assignment policy is also proposed to provide better modes for selected activities in fuzzy scheduling. This approach is simple and straightforward in practical applications, thus providing a framework for solving FMMRCPS problems involving uncertain activity duration times modelled by fuzzy numbers.

# Chapter 8

# Fuzzy Genetic Algorithms for FMMRCPS

## 8.1 Introduction

Real-world project scheduling problems often have the constraints of both resource limitation and precedence relations. In recent years, exact approaches to the classical single mode RCPS have been developed (Dorndorf *et al* 2000, Brucker *et al* 1998, Mingozzi *et al.* 1998). However, these approaches cannot find an optimal schedule when the scheduling problem is over 50 or 60 activities (Hartmann 1998, Demeulemeester and Herroelen 2002). Genetic algorithms (GA) have been widely applied in many areas since they are robust and effective in solving complex combinatorial optimisation problems such as project scheduling problems. A number of researchers have made efforts in developing GAs for solving realistically-sized RCPS problems (Lee and Kim 1996, Cheng and Gen 1998, Hartmann 1998, Toklu 2002, Hindi *et al.* 2002).

The multiple mode resource-constrained project scheduling (MMRCPS) is a generalisation of the single-mode RCPS. It is much more difficult to solve because MMRCPS is a strong NP-hard problem. Although several researchers have developed effective exact approaches to MMRCPS (Hartmann and Drexl 1998, Sprecher and Drexl 1998, Heilmann 2003), these approaches are unable to solve optimal solutions with more than 20 activities and 2 modes in project scheduling under resource constraints. Some efforts have been made in developing GAs for resolving MMRCPS problems (Mori and Tseng 1997, Özdamar 1999, Hartmann 2001, Alcaraz *et al.* 2003).

A more realistic model of project scheduling should consider not only multiple performance modes, but also the uncertainty caused by lack of precise information on activity duration times, referred to as FMMRCPS, which has been presented in

Section 4.4. FMMRCPS often exists in realistic settings. However, this problem may be not easy to solve because it not only has to tackle complex combinatorial problems in project scheduling, but also has to resolve the fuzziness of activity duration times. Özdamar and Alanya (2001) proposed GA for solving FMMRCPS with a case study of software development. In my research, I develop two fuzzy genetic algorithms, incorporation with fuzzy set theory to find an approximately optimal solution globally for FMMRCPS problems of any realistic sizes efficiently and effectively.

## 8.2 Fuzzy Genetic Algorithms

Genetic algorithm (GA) was first introduced by Holland (1975). It is the intelligent exploitation of a random search that is characterised by a parallel search of the entire space against a conventionally point-by-point search in finding optimal solutions. The parallel search is the way of obtaining a set of possible solutions, termed as *population*. An individual in the population is a string of symbols covering all necessary information about the individual's status, named as *chromosome*, in which each symbol is referred to as a *gene*. The individual chromosomes in the population are evaluated based on the mechanics of natural selection in biological systems, attempting to implement the idea of survival of fit chromosomes and the elimination of weak ones. Today this idea is popularly applied in many areas, particularly in complex combinatorial optimisation problems for searching approximately global optimal solutions within reasonable computational time where exact mathematical approaches are inadequate to handle.

FMMRCPS is a typically complex combinatorial optimisation problem with fuzziness. No exact algorithms are able to cope with such complex characteristics when the problem size increases. Furthermore, fuzzy operations have to take place in scheduling because of fuzzy duration times of activities involved. In my research, GA combined with fuzzy operations is applied to FMMRCPS, referred to as fuzzy genetic algorithm. When fuzzy GA is applied to FMMRCPS, each chromosome represents a possible schedule. Because of the characteristics of FMMRCPS, the chromosome I designed is made up of three sub-chromosomes that I will present in Section 8.3.

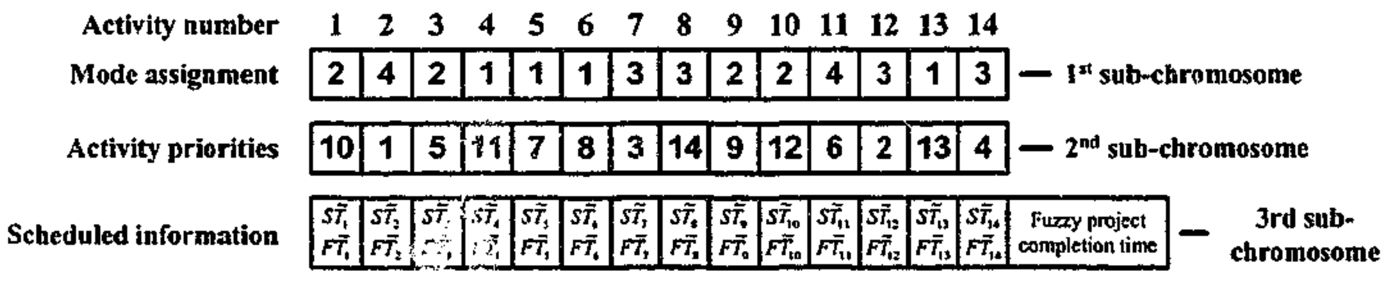
Initially, a certain size of population of chromosomes (schedules) should be generated. Each chromosome evolves through successive generations by employing some genetic operations. During each generation, chromosomes will be evaluated by a specified fuzzy fitness function. A fit chromosome should have a small fuzzy project completion time, in terms of minimising fuzzy project completion time. The fuzzy fitness function is specially designed to meet this criterion to ensure that fitter chromosomes (schedules with smaller project completion times) always have higher survival probabilities. Through a number of generations, the surviving fittest chromosome (the best schedule) should be found eventually, representing the globally optimal or near globally optimal solution to an FMMRCPS problem.

## 8.3 Explicit Representation of FMMRCPS

In developing a fuzzy GA, a chromosome should represent a solution to the problem. To reflect the problem-specific nature of FMMRCPS, a chromosome has to be carefully designed, to operate on chromosomes easily, using the principle of GA without any distortion. To avoid the procedure of decoding a chromosome that is often required in GA, the chromosomes I designed in fuzzy GA can directly reflect all necessary scheduling information without any interpretation for a solution. As characteristics of FMMRCPS, the chromosomes I designed contain three elements: (a) mode assignment, (b) activity priority, and (c) details of fuzzy start and finish times of each activity as well as the fuzzy project completion time. Let $\Phi_{\lambda}^{\gamma}$ be a chromosome $\gamma$, representing a possible schedule in the λth generation. The 1st and 2nd rows respectively describe a mode $m_j$ and a priority value $\omega_j$ assigned to each activity *j* in chromosome $\gamma$. The 3rd row indicates the fuzzy start $\widetilde{ST}_j$ and finish $\widetilde{FT}_j$ times of each activity *j*, together with the last column as the trailing information on fuzzy project completion time $\widetilde{PCT}$ for chromosome $\gamma$. Therefore, a chromosome comprising three sub-chromosomes can be expressed mathematically as

$$\Phi_{\lambda}^{\gamma} = \left| \begin{array}{cccc} m_1 & \ldots & m_J & \\ \omega_1 & \ldots & \omega_J & \\ \widetilde{ST}_1, \widetilde{FT}_1 & \ldots & \widetilde{ST}_J, \widetilde{FT}_J & \widetilde{PCT} \end{array} \right| \tag{8.1}$$

To visually describe a chromosome for representing a possible schedule to FMMRCPS, Figure 8.1 depicts an example of a chromosome containing three sub-chromosomes. All the information about an activity is indicated by the corresponding location boxes of three sub-chromosomes vertically under the activity number. The 1st and 2nd sub-chromosomes represent mode assignment and the priority value for each activity in a project respectively. The 3rd sub-chromosome stores the fuzzy start and finish times of each activity with the trailing information on fuzzy project completion time. As shown in Figure 8.1, the location of a gene in each sub-chromosome, except for the last box of the 3rd sub-chromosome, corresponds to the activity number. The numbers in the location box of the 1st and 2nd sub-chromosomes represent the mode selected and priority value assigned respectively. In the 3rd sub-chromosome, each location box places both the start and finish times of its corresponding activity and the last extra box in the 3rd sub-chromosome is referred to as the trailing information and contains fuzzy project completion time, treated as fitness value in fuzzy GA for FMMRCPS. The example shown in Figure 8.1 is a possible scheduled solution of a project having 14 activities. Location 9 corresponds to activity 9, to which mode 2 is assigned in the 1st sub-chromosome of mode assignment, and the priority value is given as 12 in the second sub-chromosome of activity priorities. The fuzzy start $S\widetilde{T}_9$ and finish $F\widetilde{T}_9$ times of activity 9 appears in column 9 of the 3rd sub-chromosome.



Activity number 1 2 3 4 5 6 7 8 9 10 11 12 13 14
Mode assignment 2 4 2 1 1 1 3 3 2 2 4 3 1 3
1st sub-chromosome
Activity priorities 10 1 5 11 7 8 3 14 9 12 6 2 13 4
2nd sub-chromosome
Scheduled information
ST1 FT1 … ST14 FT14
Fuzzy project completion time
3rd sub-chromosome

Figure 8.1 Solution presentation by a chromosome with 3 sub-chromosomes

## 8.4 Mode Assignment Operations

As fuzzy GA deals with FMMRCPS, the designed chromosome should have a capacity of manipulating mode changes in chromosomes during offspring generations. For mode operation, the system allows the user to choose one out of two categories of either assigning modes independently or mode assignment based on search strategies. The option of assigning modes independently is that the mode changes only rely on the user preset and is not concerned with the trends of solution changes during offspring generation, whereas mode assignment concerned with search strategies is based on the current conditions in generated solutions from the recent past to the present.

For assigning modes independently, a different range of selections for mode assignment can be chosen simultaneously for the 1st sub-chromosome of mode assignment. These options include the following:

- Assigning mode unchanged for *certain* or *random* times;
- Assigning mode changed for *certain* or *random* times;
- Swapping mode between any two locations for *certain* or *random* times;
- Changing mode in random locations for *certain* or *random* times;
- Choosing the shortest feasible mode in random locations for *certain* or *random* times.

The shortest feasible mode in the last option is that the mode selected has as shorter a duration time as possible according to the currently available resources.

For the other category of mode assignment based on search strategies, three or less options can be chosen out of six options for mode operation in the 1st sub-chromosome. These options show the following:

- Mode changes for a specific activity in *certain* or *random* times;
- Followed by mode changes for two specific activities in *certain* or *random* times if solutions are not improved in the number of times preset by the user;

- Followed by mode changes in random locations up to the entire locations of the sub-chromosome for *certain* or *random* times.

One or two specific activities in the first and second bullet-headed outline above indicates the activity/activities for mode change were either selected randomly by the system or preselected before the running the fuzzy GAs.

As presented above for mode assignment, there are a number of mode operational options. Mode assigned to an activity may not be changed for a number of times, or may be changed randomly, or locations are swapped, or the shortest feasible modes are selected in a number of locations. These mode operations can be flexibly combined prior to the running of the fuzzy GAs or can be determined by the search strategies. Once mode operations are specified, the system will automatically assign different modes to one or more activities during the running of the fuzzy GAs.

## 8.5 Activity Priority Operations

To generate the next generation, genetic operations are not only applied to mode assignment, but also required to have some distinctive ways of operating on the 2nd sub-chromosome for activity priorities at the same time. Three genetic operations are used to operate on the 2nd sub-chromosome in the fuzzy GAs, including (a) mutation, (b) crossover, and (c) neighbourhood swaps.

### 8.5.1 Mutation

The mutation operation is used to alter only two genes from the 2nd sub-chromosome of a parent while generating that for a child. That is, two locations along the 2nd sub-chromosome will be selected randomly, and their priority values on the two locations are swapped to generate the 2nd sub-chromosome for a child. Figure 8.2 shows an example of how the mutation is operated. In the example, two locations of activities 4 and 9 are chosen randomly and their values in the two locations (i.e. 11 and 9) are interchanged so as to create the 2nd sub-chromosome for an offspring.

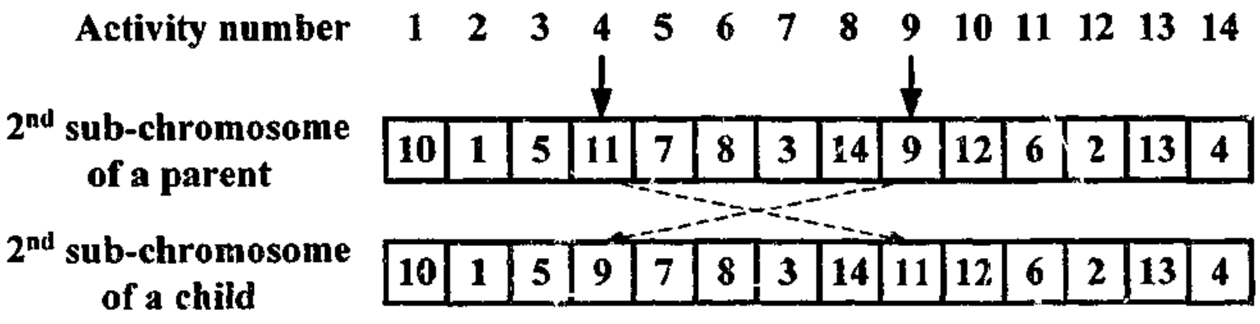


Figure 8.2 Mutation operation

As shown in Figure 8.2, it is noticed that the generated 2nd sub-chromosome of the offspring has the similar genetic structure to that of the parent which differs in two genes only. The purpose of applying this operation is to introduce some chromosomes with small variations into the population. This operation is used to find a good schedule in a local search area, when the trend of generated schedules has been improved in the recent searching. When this operation is applied, a mode in each location of the 1st sub-chromosome is seldom changed because dramatic changes in the genetic structure of the 1st sub-chromosome will lose the meaning of searching a good solution locally.

### 8.5.2 Crossover

The crossover operation combines the features of the 2nd sub-chromosomes of two parents to generate that of an offspring. The nature of the crossover can be viewed as a permutation to form the 2nd sub-chromosome of a chid. In this case, the 2nd sub-chromosome of the chid is not similar to that of either parent. The 2nd sub-chromosome of a child generated, takes some genes from that of parent 1 at random, and fills the empty locations with genes from that of parent 2, by a left to right shift.

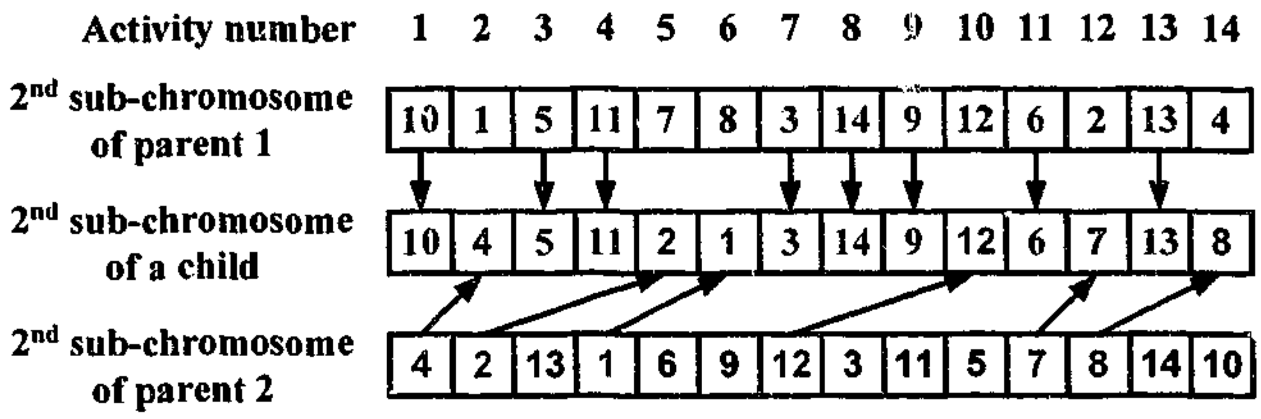

Activity number 1 2 3 4 5 6 7 8 9 10 11 12 13 14
2nd sub-chromosome of parent 1
10 1 5 11 7 8 3 14 9 12 6 2 13 4
2nd sub-chromosome of a child
10 4 5 11 2 1 3 14 9 12 6 7 13 8
2nd sub-chromosome of parent 2
4 2 13 1 6 9 12 3 11 5 7 8 14 10

Figure 8.3 Crossover operation

An example of crossover operation is shown in Figure 8.3, where 8 genes representing the priority values of 10, 5, 11, 3, 14, 9, 6 and 13 randomly selected from the 2nd sub-chromosome of parent 1 are placed at the same locations as that of a child, and 7 empty locations of the child are filled in the priority values of 4, 2, 1, 12, 7 and 8 from that of parent 2 from left to right in sequence. These values that have not been taken from that of parent 1 during the operation, are taken from that of parent 2. As shown in Figure 8.3, the 2nd sub-chromosome generated for the child is different from either parent. In terms of project scheduling, the priority values to each activity are changed greatly. This operation is often applied to situations where different search spaces would like to be explored in order to avoid trapping in local optima. In this case, the mode assignment for the 1st sub-chromosome is often changed diversely at the same time during operation.

The crossover is one of the most important genetic operations. This operation combines the characteristics of 2nd sub-chromosomes of both parents into one 2nd sub-chromosome of an offspring. The design of the crossover operation for FMMRCPS should always have this specifically genetic feature in generating offspring because poorly designed crossover may become a sort of mutation. The crossover operation I designed in Figure 8.3 reflects the genetic structure of the new sub-chromosome by combining both genes from two parents, thus making the new sub-chromosome of a child is significantly different from its parents.

## 8.5.3 Neighbourhood Swaps

Neighbourhood swaps can be viewed as a serial of mutations by pair-wise interchanges to generate a set of the 2nd sub-chromosomes of offspring based on the original one of a parent. Each pair-wise mutation generates one sub-chromosome of an offspring. This process is repeated to generate a number of 2nd sub-chromosomes of children, until it reaches the original location of the key gene. The key gene is randomly selected by the system.

As an example, Figure 8.4 describes the operations of neighbourhood swaps. A key gene with the value of 10 in the location corresponding to activity 10 is chosen by the system. Then the key gene swaps its location with the 2nd location. From the 2nd location, the key gene moves to the 3rd, 4th locations, and so on successively, and simultaneously the gene on the location taken by the key gene is moved to the initial location of the key gene. The operations are continued until the key gene is positioned immediately adjacent to its initial starting position.

| Activity number | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | | ↓ Key gene | | | | |
| 2nd sub-chromosome of a parent | 10 | 1 | 5 | 11 | 7 | 8 | 3 | 14 | 9 | 12 | 6 | 2 | 13 | 4 |
| 2nd sub-chromosome of child 1 | 10 | 12 | 5 | 11 | 7 | 8 | 3 | 14 | 9 | 1 | 6 | 2 | 13 | 4 |
| 2nd sub-chromosome of child 2 | 10 | 1 | 12 | 11 | 7 | 8 | 3 | 14 | 9 | 5 | 6 | 2 | 13 | 4 |
| 2nd sub-chromosome of child 3 | 10 | 1 | 5 | 12 | 7 | 8 | 3 | 14 | 9 | 11 | 6 | 2 | 13 | 4 |
| 2nd sub-chromosome of child 4 | 10 | 1 | 5 | 11 | 12 | 8 | 3 | 14 | 9 | 7 | 6 | 2 | 13 | 4 |
| 2nd sub-chromosome of child 5 | 10 | 1 | 5 | 11 | 7 | 12 | 3 | 14 | 9 | 8 | 6 | 2 | 13 | 4 |
| 2nd sub-chromosome of child 6 | 10 | 1 | 5 | 11 | 7 | 8 | 12 | 14 | 9 | 3 | 6 | 2 | 13 | 4 |
| 2nd sub-chromosome of child 7 | 10 | 1 | 5 | 11 | 7 | 8 | 3 | 12 | 9 | 14 | 6 | 2 | 13 | 4 |
| 2nd sub-chromosome of child 8 | 10 | 1 | 5 | 11 | 7 | 8 | 3 | 14 | 12 | 9 | 6 | 2 | 13 | 4 |

Figure 8.4 Neighbourhood swaps

Neighbourhood swaps are used to improve schedules to FMMRCPS by searching the neighbourhood from an initial solution. The idea is that this process may generate a good schedule by making both small changes in the 2nd sub-chromosome whilst making no or small changes in the 1st sub-chromosome of mode assignment.

## 8.6 Search Strategies

In fuzzy GAs, intensive and extensive search strategies are used in turn to guide search in the evolution process depending on the current condition. Hill-climbing is an example of the intensive strategy that explores the best solution of possible improvement by ignoring the exploration of the other search spaces. Random search is an example of the extensive strategy that explores diverse search spaces by ignoring the space that may be a promising area.

The intensive search strategy is used to search intensively in the specific local space when the improved solution is found after generating a number of offspring. The mutation and neighbourhood swaps are usually employed to the 2nd sub-chromosome to support this strategy with no and small changes in the genetic structure of the 1st sub-chromosome. However, if an improved solution cannot be found through a number of operations, the extensive strategy will be followed to explore other search spaces. There are two steps in employing the extensive strategy. First, the extensive search in one dimension change in either the 1st sub-chromosome of mode assignment or the 2nd sub-chromosome of activity priorities. Second, the extensive search in two dimension changes in both the 1st and 2nd sub-chromosomes.

In the extensive search, sometimes the space explored needs to be not far away from the original search space in order to widen the exploring space from the original area. This strategy is used in situations where improved solutions cannot be found after a number of searches in a specific area by mutation and particularly neighbourhood swap operations. Crossover operations for the 2nd sub-chromosome or dramatic changes in mode at random or by presetting in the system, for the 1st sub-chromosome, are conducted until the criterion set by the user in the system is met.

The other way of the extensive search is to jump out from the current search space while no improved solutions can be found in that space or after a widening search has been carried out. In this case, sometimes it may be a good idea to explore the space far away from the current one in the hope for finding promising areas for good solutions.

## 8.7 Fitness Evaluation

In generating each generation, all chromosomes are evaluated against a certain measure of fitness. The fitter chromosomes are selected in the artificial version of the naturally surviving phenomenon referred to as the survival of the fittest and the elimination of weakness. In this phenomenon, all chromosomes from the entire population in a generation are calculated against a fitness function so that some fitter chromosomes can be chosen as parents for generating the next generation.

The principle of GA is that the fitter chromosome should have a higher value of the fitness function in proportional selection. Under the current context, the objective is to minimise the fuzzy project completion time, while GA seeks a higher value for fitness. As a result, the objective of the project completion time cannot be applied directly as a means of evaluating fitness for chromosomes. A specific fitness function needs to be developed to determine the chances of survival based on the status of the fuzzy project completion time of individual chromosomes. That is, the shorter project completion time should have a higher value presented in the specifically designed fitness function.

The fitness function should accommodate the situation where fuzzy duration times are involved in FMMRCPS. To simplify fuzzy arithmetic, all fuzzy project completion times in the trailing information of the 3rd sub-chromosome of individual chromosomes in a whole generation, are converted into ranking indices. The calculation of fuzzy ranking indices has been presented in Sections 2.5 and 5.2. Let $\tilde{f}_{\lambda(\max)}$ and $\tilde{f}_{\lambda(\min)}$ be the maximum and minimum fuzzy project completion times respectively in the λth generation, and $\widetilde{f}_{\lambda}^{\gamma}$ represent the objective function of the fuzzy project completion time for chromosome $\gamma$ in the generation. The ranking indices of $\widetilde{f}_{\lambda(\max)}$, $\widetilde{f}_{\lambda(\min)}$ and $\widetilde{f}_{\lambda}^{\gamma}$ are denoted as $R(\widetilde{f}_{\lambda(\max)})$, $R(\widetilde{f}_{\lambda(\min)})$ and $R(\widetilde{f}_{\lambda}^{\gamma})$ respectively. $v_{\lambda}^{\gamma}$ represents the value of fitness for chromosome $\gamma$ in the λth generation. The fitness function for chromosome $\gamma$ in the λth generation measured by the fuzzy project completion times can be expressed as:

$$v_{\lambda}^{\gamma} = \frac{R(\widetilde{f}_{\lambda(\max)}) - R(\widetilde{f}_{\lambda}^{\gamma}) + \alpha}{R(\widetilde{f}_{\lambda(\max)}) - R(\widetilde{f}_{\lambda(\min)}) + \alpha} \tag{8.2}$$

where $\alpha$ is a parameter, and $\alpha$ is chosen as a positive real number within the open interval (0, 1). There are two purposes for setting up $\alpha$. First, it prevents the denominator of formula (8.2) from being zero. Second, it makes an adjustment in some way to naturalise the random selection.

Formula (8.2) implicitly reflects the status of fuzzy project completion times of individual chromosomes. Individual chromosomes are ordered depending on their fitness, obtained by Formula (8.2), so that shorter fuzzy project completion times of chromosomes have a higher probability of being selected.

## 8.8 Fuzzy GA and Fuzzy GA with Tabu Mechanism

In my research, two different approaches: (a) pure fuzzy GA and (b) fuzzy GA with tabu mechanism, referred to as hybrid fuzzy GA, are developed. The experiment has been conducted to examine the performance between them.

### 8.8.1 Parameter Setting and Combination of Genetic Operations

Before running fuzzy GAs, both parameters and genetic operations must be determined for either pure fuzzy GA or hybrid fuzzy GA. Proper parameter settings and appropriate combinations of genetic operations are crucial to the success of both pure and hybrid fuzzy GAs. The parameter setting and choice of good genetic operations are usually based on computational experiments, and there are no general rules for the setting and choice. These parameters and the combination of genetic operations are described as follows:

- *Population size*:

  How many individual chromosomes (possible solutions) required are usually determined on the basis of activity numbers, mode numbers and resources types, that are involved in a particular FMMRCPS. For instance, by experiment, the appropriate population size containing 60-80 chromosomes may be adequate for a project having 20 activities, 2 modes, 3 resource types. A larger population may give a larger number of diverse solutions in one generation, but it requires much more computational time.

- *The number of fit chromosomes chosen as parents*:

  This parameter determines how many fit chromosomes may act as parents for generating offspring in the next generation. The experiment shows that the proper number of fit chromosomes chosen may be in the range of 10% - 25% of the population.

- *The number of generations*:

  The number of generations chosen for the pure or hybrid fuzzy GA is based on the size of an FMMRCPS problem and the number of modes for activities. By experiment, the adequate number of generations may be around 50 when the size of the project scheduling is the same as the example presented in the population size above.

- *Tabu size*:

  The tabu size is decided when only hybrid fuzzy GA is applied. The size of tabu can be fixed or selected at random. Through experiment, the adequate size of tabu may be within the range of one to three times of the square root of the number of activities of a project. The function of the tabu mechanism will be explained in Section 8.8.5.

- *Assigning modes independently* or *based on search strategies:*

  Either of these two options is required to be chosen before running either pure or hybrid fuzzy GA. In each option, a variety of combinations in mode assignment can be made depending on different requirements for running fuzzy GAs for a specific project. The details of the combination of mode assignment have been presented in Section 8.4.

- *Operations for activity priorities*:

  The operations on the 2nd sub-chromosome of activity priorities have four different options: (a) specification of *the number* of mutation, followed by *the number* of neighbourhood swap and finally *the number* of crossover operations, (b) *random times* in the sequence of mutation, neighbourhood swap, and crossover operations respectively, (c) any operations chosen randomly, and (d) only one operation selected in mutation, or crossover or neighbourhood swap operations for the whole approach.

### 8.8.2 Initial Population

In the beginning of either pure or hybrid fuzzy GAs, initial chromosomes (possible schedules) are generated repeatedly until the population size set in the parameter setting is reached. To manipulate individual chromosomes effectively in terms of offspring generation, each individual chromosome is made up of three sub-chromosomes. The number of genes designed in each sub-chromosome equals the number of activities in a project, and the location of each gene corresponds to an activity number in the project. A complete chromosome, representing a possible schedule is created by the following three phases.

*Phase 1*: For the 1st sub-chromosome of mode assignment, the mode of activity $j$ is randomly assigned to a location corresponding to that activity from its available modes ($m_j \in M_j$) until each location in the sub-chromosome is allocated by a mode from the available modes of each activity.

*Phase 2*: For the 2nd sub-chromosome of activity priorities, an integer for the priority value of activity $j$ is randomly selected, and subsequently allocated to a location that represents the activity until all locations of the sub-chromosome are allocated a by priority value. To simplify the procedure of assigning a priority value to each activity in my fuzzy GAs, an integer is selected randomly from the range of 1 to the number of activities ($\omega_j \in \{1, 2, \ldots, J\}$) rather than the values obtained by applying priority rules that many researchers have previously applied. This avoids the same values occurring in activity priorities whilst employing priority rules. Here it is assumed that a smaller value of priority for activities always has a higher priority in fuzzy GAs.

*Phase 3*: Once the 1st and 2nd sub-chromosomes are created, the 3rd sub-chromosome will be generated by retaining the shortest project completion time among two schedules obtained through fuzzy forward and backward scheduling.

In these three sub-chromosomes, all genes on the 1st and 2nd sub-chromosomes are used to calculate the start and finish times of each activity and the objective value for fuzzy scheduling whilst the fuzzy forward or backward scheduling is conducted. In addition, the 1st and 2nd sub-chromosomes of the fitter chromosomes can be manipulated for generating offspring if these chromosomes are selected as parents.

The 3rd sub-chromosome only carries relevant information about a schedule and is never involved in any operations.

### 8.8.3 Next Generation Construction

As stated in Section 8.8.2, all initial solutions in the first population are generated randomly. To avoid generating the same chromosomes initially, a mechanism is built into the system to ensure that all chromosomes generated in the generation are different from each other.

In the construction of the next generation, a specified number of fit chromosomes are selected in the generation, based on the fitness function given in Formula (8.2). To improve the performance of fuzzy GAs whilst generating the next generation, it is important to create a number of chromosomes with a small variance, in an attempt to find a good solution in a specific area. After this procedure, crossover operations may be applied in order to prevent from premature convergence or being trapped in local optima when generating offspring.

### 8.8.4 Pure Fuzzy GA

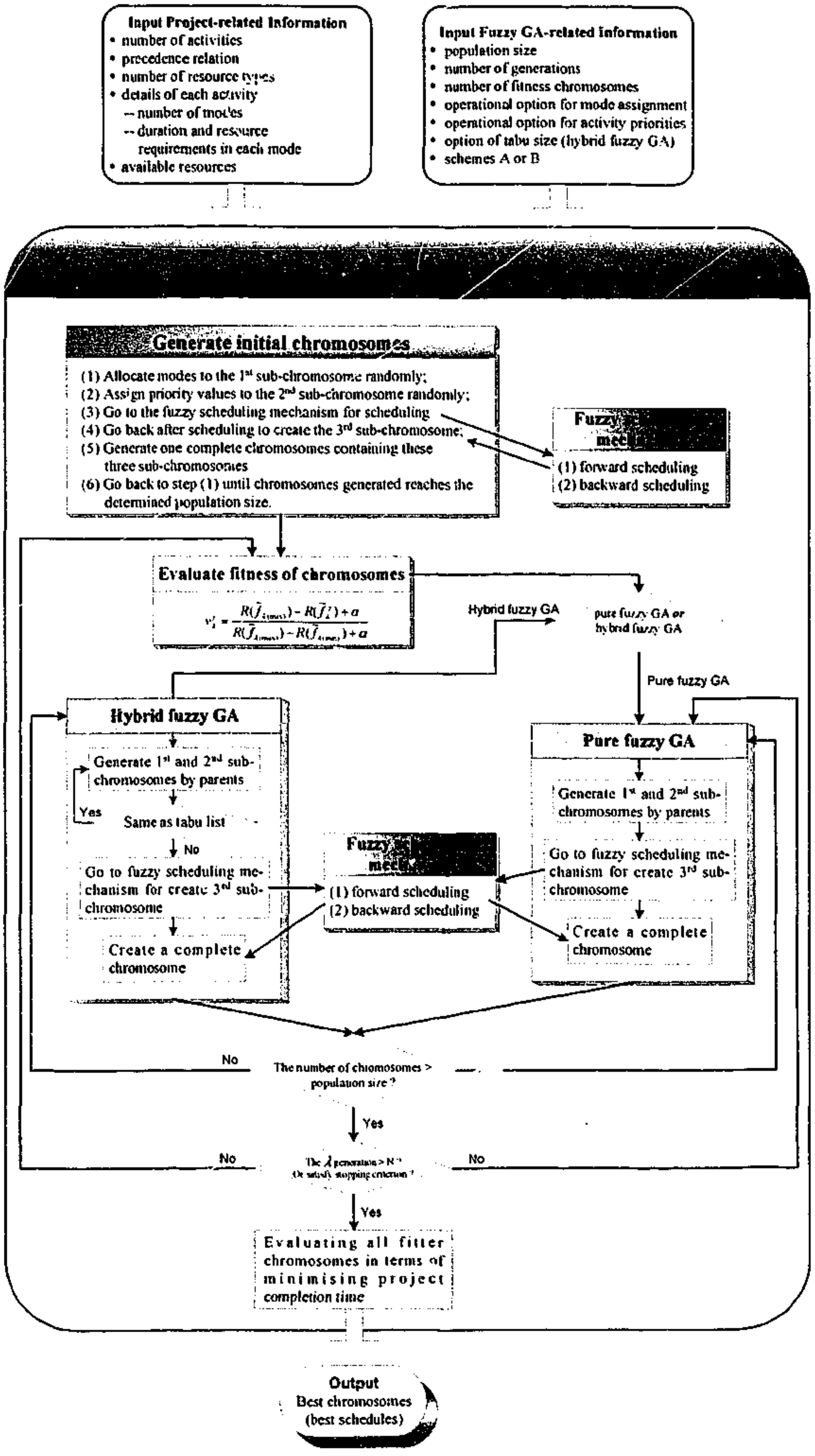
Figure 8.5 presents the structure of fuzzy GAs. In this subsection, a pure fuzzy GA developed for FMMRCPS is carried out by the following procedure.

Before running the pure fuzzy GA, information related to both the project and the fuzzy GA are required to be input. Project-related information includes the number of activities for a project and the precedence relations amongst activities, as well as the number of resources used in a project and resource availabilities. The number of modes, the duration time and resource requirements for each mode of activity $j$ can be either assigned by the system randomly or specified by the user. For the fuzzy GA-related information, the population size, the number of generations, the number of fitter chromosomes, and the operational options for both mode assignment and activity priorities need to be determined. The determination of these parameters has been presented in Sections 8.4, 8.5, 8.6 and 8.8.1. Schemes A and B are also required beforehand. The details of schemes A and B have been specified in Definitions 6 and 7 of Section 5.2. Schemes A and B can be used either separately or together for a schedule, once operations on both the 1st and 2nd sub-chromosomes have been conducted.

Once all required information has been input, initial solutions are generated in the 1st generation. In generating initial solutions, the system first assigns both the mode and priority value randomly to each location of gene in the 1st and 2nd sub-chromosomes, then a schedule is produced through fuzzy forward and backward scheduling, using schemes A or B or both, and the information on fuzzy start and finish times of activities as well as the project completion time of the schedule, is stored in the 3rd sub-chromosomes.

When all initial chromosomes (solutions) have been created, all the chromosomes are evaluated using the fitness function in Formula (8.2). A number of fitter chromosomes act as parents to generate offspring for the next generation using genetic operations on the 1st and 2nd chromosomes of parents. How genetic operations are performed, is preset by an operational option.

If the generations generated reach the preset number of generations *N*, or the stopping criterion is met, no more generation is required. The stopping criterion set in my fuzzy GA is the useful mechanism that terminates the fuzzy GA when an improved solution cannot be found within a specified number of times of search, thus avoiding unnecessarily continuing the process of the fuzzy GA. Finally, the fittest chromosomes will be selected among the fitter chromosomes in the last generation, as the output of the solution for FMMRCPS in terms of minimising the project completion time.


Input Project-related Information
• number of activities
• precedence relation
• number of resource types
• details of each activity
-- number of modes
-- duration and resource requirements in each mode
• available resources
Input Fuzzy GA-related Information
• population size
• number of generations
• number of fitness chromosomes
• operational option for mode assignment
• operational option for activity priorities
• option of tabu size (hybrid fuzzy GA)
• schemes A or B
Generate initial chromosomes
(1) Allocate modes to the 1st sub-chromosome randomly;
(2) Assign priority values to the 2nd sub-chromosome randomly;
(3) Go to the fuzzy scheduling mechanism for scheduling
(4) Go back after scheduling to create the 3rd sub-chromosome;
(5) Generate one complete chromosomes containing these three sub-chromosomes
(6) Go back to step (1) until chromosomes generated reaches the determined population size.
Fuzzy mec
(1) forward scheduling
(2) backward scheduling
Evaluate fitness of chromosomes
Hybrid fuzzy GA
pure fuzzy GA or hybrid fuzzy GA
Pure fuzzy GA
Hybrid fuzzy GA
Generate 1st and 2nd sub-chromosomes by parents
Yes
Same as tabu list
No
Go to fuzzy scheduling mechanism for create 3rd sub-chromosome
Create a complete chromosome
Fuzzy mech
(1) forward scheduling
(2) backward scheduling
Pure fuzzy GA
Generate 1st and 2nd sub-chromosomes by parents
Go to fuzzy scheduling mechanism for create 3rd sub-chromosome
Create a complete chromosome
No
The number of chiomosomes > population size ?
Yes
No
The λ generation > N ?
Or satisfy stopping criterion ?
No
Yes
Evaluating all fitter chromosomes in terms of minimising project completion time
Output
Best chromosomes
(best schedules)

$$v'_i = \frac{R(\tilde{J}_{i(max)}) - R(\tilde{J}'_i) + a}{R(\tilde{J}_{i(max)}) - R(\tilde{J}_{i(min)}) + a}$$

Figure 8.5 Structure of the Fuzzy GAs for FMMRCPS

### 8.8.5 Hybrid Fuzzy GA

The procedure of hybrid fuzzy GA is similar to that of pure fuzzy GA except that hybrid fuzzy GA incorporates tabu mechanism. The tabu search was proposed by Glover (1989, 1990), and originally extended the steepest descent search approach for solving optimisation problems. One of the main functions of tabu is the use of adaptive memory to guide the search behaviour that is the hallmark of tabu search.

The adaptive memory feature of tabu records information on search history that can be exploited in the search process. In my research, a short-term memory of tabu, referred to as *tabu list*, is adopted in the hybrid fuzzy GA. Using tabu can directly exclude the search alternatives classified as *forbidden* that have been visited recently, thus avoiding visiting the same solutions (or generating the same chromosomes) more than once. The size of tabu list can be set randomly by the system or specified by the user. However, the size of tabu has to be carefully decided, since a large tabu list may take too much time in evaluating the current search against every previous search stored. Experiments have demonstrated that a good size of tabu list may be around the population size divided by 10.

## 8.9 Experimental Analysis in Fuzzy GAs

To examine the performance of pure and hybrid fuzzy GAs, four projects with different sizes are conducted. To avoid human bias during data selection, the system randomly assigns fuzzy duration times and resource requirements to each mode of activities, and the number of modes for each activity can be varied, ranging from 1 to 5. Four types of renewable resources are involved in these projects.

Table 8.1 shows that the four projects have 50, 100, 150 and 200 activities respectively. In the experiment, the population sizes for these four projects are each set to be 200, and the number of generations is chosen as 250.

For hybrid fuzzy GA, the tabu size varies, based on the status of recent searches. If, during the generation of chromosomes, the same chromosomes have not appeared for a specified number of times in the tabu list (10 times in this experiment), the size of tabu will be reduced by 1. Otherwise, the size will return gradually to the original size set by the system.

Table 8.1 Results of pure and hybrid fuzzy GAs for four projects

| | 50 activities | Pure fuzzy GA | Hybrid fuzzy GA |
|---|---|---|---|
| | Best solution | (59, 77, 84, 101) | (56, 72, 81, 99) |
| Project 1 | Worst solution | (78, 92, 107, 131) | (66, 91, 105, 114) |
| | Av. deviations | 19.4 | 17.9 |
| | Max. deviations | 25.25 | 19.81 |
| | | | |
| | 100 activities | | |
| | Best solution | (89, 102, 123, 142) | (79, 93, 113, 132) |
| Project 2 | Worst solution | (101, 137, 160, 183) | (98, 121, 149, 177) |
| | Av. deviations | 21.5 | 18.2 |
| | Max. deviations | 45.98 | 41.63 |
| | | | |
| | 150 activities | | |
| | Best solution | (122, 148, 171, 193) | (118, 142, 167, 183) |
| Project 3 | Worst solution | (151, 174, 199, 235) | (142, 171, 182, 229) |
| | Av. deviations | 22.6 | 20.81 |
| | Max. deviations | 56.12 | 55.89 |
| | | | |
| | 200 activities | | |
| | Best solution | (162, 193, 214, 250) | (142, 187, 198, 239) |
| Project 4 | Worst solution | (194, 239, 265, 304) | (186, 227, 258, 292) |
| | Av. deviations | 23.9 | 22.65 |
| | Max. deviations | 89.84 | 75.32 |

To analyse the relative performances of pure and hybrid fuzzy GAs, each project set has been run 100 times. After running the fuzzy GAs in the system, the worse and best schedules in terms of fuzzy project completion times are given in Table 8.1. Furthermore, the average and maximum deviations were calculated for the two approaches. These deviations were consistently smaller in hybrid fuzzy GA than in pure fuzzy GA. As a result, the schedules produced from hybrid fuzzy GA are better and more encouraging than those of pure fuzzy GA

This experimental result can be explained by the fact that the tabu mechanism prevents chromosomes generated in the current generation from being the same as those previously generated. As such, hybrid fuzzy GA produces more diverse chromosomes in the vast search space to obtain effectively a good approximate globally optimal solution for MMRCPS.

## 8.10 Concluding Remarks

FMMRCPS is the most generalised project scheduling that usually appears in practical applications, because projects often lack precise information in regard to the activity duration time. Similar activities may have never been performed or performed infrequently. There may be insufficient information on activity duration times. Fuzzy set theory is an effective way of handling the fuzziness of the activity duration time. In addition, FMMRCPS is most difficult and complex scheduling. No exact algorithm is viable in solving such complex optimisation problems. fuzzy GAs seem to provide an effective means of handling such problems.

In this chapter, both pure and hybrid fuzzy GAs have been presented. Both approaches can deal with any practical FMMRCPS problems of large sizes with uncertain information. They are simple, straightforward and can be easily implemented for practical applications. They are effective in computational effort and robust in solving complex MMRCPS problems.

To examine the performance of these two approaches, an experiment on four projects of different sizes has been conducted. Through the calculation of the average and maximum deviations of these two approaches, there is a strong indication that fuzzy GA-tabu outperforms fuzzy GA alone in terms of obtaining a good approximate optimal solution for FMMRCPS.

# Chapter 9

# Fuzzy Simulated Annealing for FMMRCPS

## 9.1 Introduction

FMMRCPS is a generalised case of the classical resource-constrained project scheduling under situations where activities may be performed in one of several executive modes and where activity duration times are considered fuzzy. This generalised model has a great practical significance in representing the different ways of performing activities in a project under fuzzy activity duration times. The problem description of FMMRCPS has been presented in Chapter 4. However, such a problem is probably one of the most difficult scheduling problems to solve, because it not only resolves two issues of mode assignment and the scheduling sequence, but also handles fuzzy activity duration times caused by imprecise information. The literature survey shows that research publications dealing with FMMRCPS are very scant.

Researchers have recently made great efforts in developing a variety of approaches such as exact, heuristic and metaheuristic approaches in an attempt to resolve multiple mode resource-constrained project scheduling problems without taking fuzziness into consideration. A number of exact approaches to multiple mode project scheduling have been developed mainly based on enumeration schemes and branch-and-bound procedures (Sprecher and Drexl 1998, Heilmann 2003, Brucker and Knust 2003). The computational experiment on these approaches has demonstrated that such exact approaches may only solve problems to optimality where there are up to 30 activities (Bouleimen and Lecocq 2003). Heuristic approaches applying single or multiple-pass or sampling with priority rules have also been proposed for solving project scheduling with multiple executive modes (Boctor 1994, Slowinski *et al.* 1994). Heuristic approaches may always produce feasible solutions, but may not guarantee optimal or near optimal solutions. Metaheuristic approaches may provide an efficient and effective way for solving multi-mode project scheduling of practical sizes in obtaining a global or nearly global optimal solution. The concept of GA has been applied to project scheduling with multiple modes, and different approaches have been developed (Özdamar 1999, Hartmann 2001).

A number of approaches based on simulated annealing (SA) have also been developed for resolving multi-mode project scheduling (Jozefowska *et al.* 2001, 2002, Hapke *et al.* 1998, Pan and Yeh 2003d, Bouleimen and Lecocq 2003). These SA approaches, developed for multi-mode project scheduling, do not deal with fuzzy activity duration times. In practical settings, FMMRCPS is one of the most generalised models for solving project scheduling. Developing a variety of effective and efficient approaches to such a model has implications for favourably resolving complex project scheduling in realistic settings. Although several researchers have recently taken fuzziness into consideration in solving multi-mode project scheduling (Hapke *et al.* 1999, Özdamar and Alanya 2001, Pan and Yeh 2003c), research publications on this issue are few and more research developments are needed. This chapter will present two versions of simulated annealing approaches to FMMRCPS, in search of global or approximately global optimality under fuzzy activity duration times, with a view to minimising the fuzzy project completion time.

## 9.2 Fuzzy Simulated Annealing

Simulated annealing (SA) is a heuristic optimisation technique for tackling complex combinatorial problems where a large number of variables are involved. The metaphor of SA comes from metallurgy and is based on the thermal process for obtaining a low energy level while a metal is cooling gradually from molten to solid state. The mathematical approach for annealing was developed by Metropolis *et al.* (1953). This cooling process, referred to as *annealing* by Metropolis' algorithm, asserts that, if the melted metal cools down too quickly from a very high temperature, the desired properties, such as metal strength and magnetism will be lost or degraded. To enhance or retain these properties, the high energy level in the melted state of the metal must be gradually cooled down to its minimum energy level whilst reaching a solid state. However, if the annealing is very slow, the properties still remain the same, and the time spent on annealing will be wasted. To reach the state of the minimum energy level, temperature control is important in the annealing process.

Since Kirkpatrick *et al.* (1983) adopted the idea of SA in physical chemistry to solve large combinatorial problems for obtaining an approximate global optimisation, SA algorithms have been applied to a broad field of combinatorial optimisation such as scheduling, assignment, and network flows (Rutenbar 1989). The resource-constrained project scheduling problem is a typical optimisation problem of this kind. For such a combinatorial problem, SA has proven to be a powerful stochastic search algorithm for applications where there is little prior knowledge available about the problem (Pannetier, 1990; Yao, 1995 and Lee *et al.*1996).

SA proposed for solving complex combinational problems assumes the availability of deterministic data. However, the problem to be tackled here is multi-mode project scheduling under situations where the activity duration time is fuzzy. This has been discussed in Section 4.4, and defined as FMMRCPS. In solving FMMRCPS, two versions of SA approaches incorporating the use of fuzzy arithmetic are developed, referred to as fuzzy SA.

Before presenting the two versions of fuzzy SA approaches to FMMRCPS, the solution representation, Metropolis' criterion, neighbourhood generation, and key parameters will be introduced in the following individual sections since these are important components for processing fuzzy SA approaches.

## 9.3 Solution Representation

The solution representation is an important component in fuzzy SA, and differs with different natures of problems. The designed solution representation should not only represent the nature of a specific problem, but also suit neighbourhood operation in fuzzy SA. Because of the complex characteristics of FMMRCPS as well as the requirement to minimise the fuzzy project completion time, the solution presentation should be designed to reflect these features. In addition, the solution presentation should be easily manipulated for neighbour generation without any distortion during the process of fuzzy SA.

The solution presentation designed here contains four elements: (a) a list of mode assignments of activities for each stage of partial schedules, (b) a list of sets of priority values to the activities in each partial schedule, (c) a list of all stages of partial schedules, and (d) the objective function of the project completion time. The partial schedule, as defined in Definition 5 of Section 5,2, is a schedule where only part of activities are scheduled at a particular scheduled time.

The first element in the solution presentation is the mode assignment list of partial schedules in all stages of scheduling. It is the list of sets of specific modes assigned for activities on partial schedules. In addition, the mode selected for an activity should be within the available modes of the activity. Let $mode_{list}$ be a list of sets of modes assigned to activities on partial schedules, and let $PS_n$ denote a partial schedule in stage $n$ ($n$ = 1, 2,..., $N$). The mode assignment list for partial schedules in each stage can be mathematically defined as

$$mode_{list} = \sum_{n=1}^{N} \{j \in PS_n \mid \{m_j\},\} \quad m_j = 1,2,...,M_j; j = 1,2,...,J \tag{9.1}$$

The second element is a list of sets of priority values assigned to activities in each partial schedule. To simply assigning priority values to activities rather than employing priority rules, a priority value randomly assigns a different integer to each activity for generating the initial solution at the beginning of fuzzy SA. That is, if a project contains 100 activities, a different integer from 1 to 100 will be assigned to each activity. This element is used to manipulate priority values to certain or all activities when the operation on activity priorities is required. Let $priority_{list}$ be a list of activity priorities in each partial schedule and $v_j$ be a priority value assigned to activity $j$. The list of sets of activity priorities in every partial schedule of the whole schedule can be represented as

$$priority_{list} = \sum_{n=1}^{N} \{j \in PS_n \mid \{v_j\},\} \quad j = 1,2,...,J \tag{9.2}$$

Let $PST_{list}$ be a collection of all partial schedule times in a complete schedule. The complete schedule is a schedule where all activities in a project are scheduled, as defined in Definition 5 of Section 5.2. Let $PST_n$ denote the partial schedule time in stage $n$ of scheduling. The set of all partial schedule times can be expressed as

$$PST_{list} = \sum_{n=1}^{N} \{PS\widetilde{T}_n,\} \tag{9.3}$$

The third element of the solution presentation records all details of partial schedules in each stage of scheduling. Let $PS_{list}$ be a list of sets of partial schedules for every stage from 1 to $N$, and let $A(\widetilde{t}_n)$ and $C(\widetilde{t}_n)$ denote the active set and complete set at fuzzy partial schedule time $\widetilde{t}_n$ of stage $n$ in scheduling respectively. The active set, $A(\widetilde{t}_n)$ is the set for placing those activities that are being scheduled and have not been finished at the fuzzy partial schedule time $\widetilde{t}_n$ of stage $n$ whereas the complete set is the set where the activities have been completed at the fuzzy partial schedule time $\widetilde{t}_n$ of stage $n$. The details of the active set and complete set have been defined in Definitions 2 and 3 of Section 5.2. The list of all partial schedules in every stage of scheduling can be represented as

$$PS_{list} = \sum_{n=1}^{N} \{(A(\widetilde{t}_n), C(\widetilde{t}_n)) \mid j \in A(\widetilde{t}_n) \cup C(\widetilde{t}_n)\} \quad n = 1,2,\ldots,N;\ \widetilde{t}_n = PS\widetilde{T}_n \tag{9.4}$$

The last element of the solution presentation represents the objective function of fuzzy project scheduling, expressed as $\widetilde{f}(i)$. The objective function of project scheduling here is the fuzzy project completion time that can be represented as the partial schedule time $PS\widetilde{T}_N$ at the last stage $N$, plus the longest fuzzy finish time of activity $j$ among those activities in the complete set $C(\widetilde{t}_N)$ for the last stage $N$ of scheduling. The objective function $\widetilde{f}(i)$ of fuzzy project completion time can be described as

$$f(i) = PS\widetilde{T}_N + \max\{F\widetilde{T}_j \mid j \in C(\widetilde{t}_N)\} \tag{9.5}$$

The above defined solution presentation has a number of merits. The solution representation can initially indicate the fuzzy project completion time as the objective function of scheduling. It also reflects all necessary information required by FMMRCPS, including both the mode assignment and activity priorities, as well as the sequence of activities of partial schedules for every fuzzy scheduled time. Furthermore, the solution representation can be manipulated efficiently and effectively in fuzzy SA to generate a neighbouring solution. Therefore, the solution representation designed here meets all requirements of both fuzzy SA and FMMRCPS.

## 9.4 Metropolis' Criterion

Metropolis' criterion (1953) is a core part of simulated annealing. This criterion claims that the new neighboring solution, generated from the current one, is definitely accepted if the new neighboring solution is better than the current one. However, some bad neighboring solutions may be occasionally accepted under the acceptance probability of Metropolis' criterion, in order to avoid being trapped into local optima. Let $f(i)$ be the objective function for the current solution, $f(i+1)$ be the objective function of neighboring solution, and $T$ be the current temperature. The acceptance probability $P_{mc}$ of Metropolis' criterion for the worse neighboring solution, generated from the current one, is expressed as

$$P_{mc} = e^{-[f(i+1)-f(i)]/T} \tag{9.5}$$

As shown in Formula (9.5), the influence of the acceptance probability is the variance of two values of the objective function between the new neighboring and current solutions, and the measure of the current temperature. The acceptance probability of Metropolis' criterion indicates that a smaller degradation of a new neighboring solution may have a higher acceptance probability. In addition, a higher value of temperature $T$ gives a greater acceptance probability. However, the acceptance probability of Metropolis' criterion can only be applied to crisp numbers.

FMMRCPS is a multi-mode project scheduling mode involving fuzzy activity duration times. To apply the acceptance probability of Metropolis' criterion, fuzzy values (the fuzzy project completion time) have to be defuzzified. Let $\tilde{f}(i+1)$ be a new fuzzy neighboring solution generated from the current fuzzy solution $\tilde{f}(i)$. The values of the fuzzy objective function in the current $\tilde{f}(i)$ and neighborhood $\tilde{f}(i+1)$ solutions are converted into ranking indices $R(\tilde{f}(i))$ and $R(\tilde{f}(i+1))$ respectively. The calculation of ranking indices has been presented in both Remark 2 and Example 5.5 of Section 5.2. To evaluate the new fuzzy neighboring and current fuzzy solutions, the rules of fuzzy ranking are applied. The rules of fuzzy ranking are presented in Section 2.5 and Examples 5.4 and 5.5. Let *MC(T)* denote Metropolis' criterion at the current temperature *T* for the acceptance probability of a new solution in fuzzy SA for solving FMMRCPS problems. *MC(T)* can be expressed as

$$MC(T)=\begin{cases}1 & \text{if } R(\tilde{f}(i+1))\leq R(\tilde{f}(i)),\\ e^{-[R(\tilde{f}(i+1))-R(\tilde{f}(i))]/T} & \text{if } R(\tilde{f}(i+1)>R(\tilde{f}(i))\end{cases} \tag{9.6}$$

Metropolis' criterion in Formula (9.6) visually indicates that a new fuzzy solution $\tilde{f}(i+1)$ is always accepted, if it is better than the current fuzzy one $\tilde{f}(i)$, whilst some "uphill moved" solutions may also be accepted based on the acceptance probability of Metropolis' criterion, $e^{-[R(\tilde{f}(i+1))-R(\tilde{f}(i))]/T}$. The acceptance of a bad solution will be decreased if this solution is very poor, or if the temperature *T* is very low. This is because fuzzy SA will converge to a near globally optimal solution when the acceptance probability of Metropolis' criterion reaches 1, under the condition of a very small value of temperature *T*.

## 9.5 Neighbourhood Generation

To generate neighbouring solutions, first of all the system requires picking up a partial schedule time $\tilde{t}_n$ as *a selected fuzzy time point*, from the set of partial schedule times of the current solution, as expressed in Formula (9.2). Once the system selects the partial schedule time $\tilde{t}_n$, a series of neighbouring solutions are generated through perturbations. This is accomplished by mode assignment, or activity priority operations, or a combination of both operations. These operations can be summarised as follows:

(1) Operation on mode assignment:

- Change modes for the activities from the selected fuzzy time point $\tilde{t}_n$;
- Change modes randomly for a number of activities in the selected fuzzy time point $\tilde{t}_n$;
- Randomly change modes in two positions;
- Change a mode in one activity only;
- Change modes for all activities from the selected fuzzy time point $\tilde{t}_n$.

(2) Operation on activity priorities in the decision sets subsequent to the selected fuzzy time point $\tilde{t}_n$:

- Randomly change priority order for all activities in a decision set;
- Randomly change priorities for some activities in a decision set;
- Randomly change priority for only one activity in a decision set;
- Priorities are unchanged in a decision set.

As stated in the previous section, the solution representation in fuzzy SA has been specifically designed for easily generating a series of feasible neighbouring solutions from the current solution in fuzzy SA. More importantly, the reschedule of all activities in all stages of partial schedules is avoided by the two versions of fuzzy SA approaches to be presented in Section 9.7, thus saving unnecessary computational time.

## 9.6 Key Parameters of Fuzzy SA

In constructing an annealing process, parameter setting may have a significant influence on the performance of fuzzy SA approaches. That is, the values of key parameters should be appropriately determined so that there is sufficient time to perform fuzzy SA approaches for achieving the globally approximate optimisation. These key parameters include the following:

- The initial value of temperature setting -- $T_0$

As suggested by Kirkpatrick *et al.* (1983), the initial value of temperature should be set appropriately large so as to make the initial probability of accepting transitions to be close to 1. Wong and Liu (1986) proposed a simple formula to determine the initial value of temperature through experiment, suggesting that the initial value should be the mean of the difference of two values of the objective function between a new neighbouring solution and the current solution in transitions for only "downhill" moves divided by natural logarithm of an initial probability. Let $M_d$ and $M_u$ be the number of downhill and uphill moves respectively in an experiment. Let $\Delta fitness_i$ be the difference of the two values of objective functions in any pair containing a new neighbouring solution and the current solution while the new neighbouring solution moves downward. To facilitate applying the fuzzy objective function, ranking indices ($R(\cdot)$) are used for calculating the average $\overline{\Delta fitness}$

$$\overline{\Delta fitness} = \frac{\sum_{i=1}^{M_d} R(\Delta fitness_i)}{M_d} \tag{9.7}$$

To ensure the initial temperature is sufficiently large, let the probability $P_0$ of accepting uphill moves be set very high (say $P_0$ $\approx$0.999) in the initial stage of fuzzy SA. The initial temperature for annealing process can be set as

$$T_0 = \frac{\overline{\Delta fitness}}{\ln(P_0)} \tag{9.8}$$

There is also another way of setting the initial value of temperature that is concerned with the proportion between the solutions accepted and all solutions generated in $T_0$ as well as the number of the solutions towards both uphill and downhill respectively, proposed by Aarts and Korst (1990). Let $\Delta unfitness_i$ be the difference of two values of the objective function between the new neighbouring and current solutions while the new neighbouring solution moves uphill. Let $\overline{\Delta unfitness}$ be the average $\Delta unfitness_i$ for all uphill moves in all transitions. To allow the fuzzy objective function to involve the determination of the initial temperature, let $\overline{\Delta unfitness}$ be expressed as

$$\overline{\Delta unfitness} = \frac{\sum_{i=1}^{M_u} R(\Delta unfitness_i)}{M_u} \tag{9.9}$$

Let $\chi_0$ be the ratio between the solutions accepted and all solutions generated in the initial temperature $T_0$. The initial temperature $T_0$ can be determined as

$$T_0 = \frac{\overline{\Delta unfitness}}{\ln(\frac{M_u}{M_u \times \chi - M_d \times (1-\chi)})} \tag{9.10}$$

According to my experiment with fuzzy SA for FMMRCPS, the initial temperature may be set to be the number of activities of a project multiplied by factorial of the maximum number of modes used in an activity. The initial temperature can be mathematically expressed as

$$T_0 = J \times \max\{M_j \mid j = 1, 2, ..., J\}! \tag{9.11}$$

- Cooling ratio – $\alpha$

$\alpha$ is a cooling ratio used to decline the temperature gradually from the initial value while the cooling procedure is processed. Usually the value of $\alpha$ is chosen as $0.8 \le \alpha < 1$. Alternatively, it can be determined by Formula (9.2), suggested by Potts and Van Wassenhove (1991).

$$\alpha = (\frac{T_L}{T_0})^{1/(epochs-1)} \tag{9.12}$$

where *epochs* is the number of times of neighbourhood generation in a temperature level for annealing, and $T_L$ is the last temperature expected. As the experiment is conducted for FMMRCPS, a value of $\alpha$ between 0.8 and 0.9 is appropriate. In the system developed for two versions of fuzzy SA, the value of $\alpha$ can be specified by the user or selected randomly by the system.

- Epoch length (the number of solutions generated in a temperature level) – $L_{markov}$

In every temperature level, there is an epoch where the neighborhood search iterates an appropriate number of times, in order to allow exploring the search space as much as possible, referred to as *epoch length* or *the Markov chain length*, $L_{markov}$. At the beginning of the cooling process, the proper length of $L_{markov}$ may be set as the number of activities in a project because it seems, in my experiment, to provide an overall better performance of fuzzy SA. To be the flexible purpose, one of several options can be selected in choosing the length of $L_{markov}$ at the beginning of fuzzy SA. A number of values of $L_{markov}$ in options are given by the system, using different proportions to the number of activities and modes involved in a project.

- Annealing ratio – $\beta$

$\beta$ is an annealing ratio that is used to change the previous length of $L_{markov}$ at the current temperature level. Applying $\beta$ controls the length of annealing at every temperature level. Most implementations choose a fixed length of $L_{markov}$ at every temperature level in solving combinatorial optimisation problems. To be robust in appropriate search for good solutions at different temperature levels, the length of $L_{markov}$ is set to be different, depending on the search status in the previous temperature level. The annealing ratio usually is selected as $0 < \beta \leq 1$ or $1 < \beta < 2$, specified by the user or determined by the system on the basis of the situation of search.

- The threshold to the number of unimproved solutions during search in the $L_{markov}$ at a temperature level – $\delta$

  $\delta$ is a threshold to the number of unimproved solutions obtained through search in the length of $L_{markov}$ at a temperature level, predetermined by the user or the system. $\delta$ is used to trace whether the current number of unimproved solutions at the temperature level exceeds this preset threshold. It is used to examine the search status at the current temperature level so as to adjust the length of $L_{markov}$ through changing the annealing ratio $\beta$ in the next cooling temperature level. The value of $\delta$ may be set to the number of activities divided by an integer number, and the integer chosen depends on the complexity of an FMMRCPS problem.

- The threshold to the number of iterations of continuous search without the best solution being found – $\epsilon$

  $\epsilon$ is a threshold to the number of search iterations without the best solution being found. $\epsilon$ may be a useful parameter to decide whether fuzzy SA should terminate if the best solution cannot be found through exploring different areas of search space in a specified time, and it can be treated as one of the stopping criteria for terminating fuzzy SA. This additional parameter is optional in the system, and whether the parameter is applied, depending on the requirements of the performance of fuzzy SA. The value of $\epsilon$ may be set as being equal to or greater than the length of $L_{markov}$ at the initial temperature.

- The threshold for the maximum iterations allowed in fuzzy SA – *MaxTime*

  *Maxtime* is the threshold that allows for the maximum iterations of search in the whole fuzzy SA, and it is automatically preset by the system. It is considered one of the stopping criteria in fuzzy SA. Once the initial value of temperature $T_0$, cooling ratio $\alpha$, and the length of $L_{markov}$ at the initial temperature level are determined, the *Maxtime* can be generated automatically by the system.

The determination of the appropriate values for key parameters has been suggested above. Experiment has shown that different values chosen in the parameters may greatly impact on the quality of optimal solutions for FMMRCPS problems. Therefore, parameter setting is an important initial step in achieving good performance of fuzzy SA.

## 9.7 Two Versions of Fuzzy SA Approaches to FMMRCPS

To resolve FMMRCPS in the fuzzy environment, I develop two versions of fuzzy SA approaches in my PhD studies, One is fuzzy SA alone, called *pure fuzzy SA approach*, and the other is fuzzy SA combined with tabu mechanism, referred to as *hybrid fuzzy SA approach*. Both versions of fuzzy SA to FMMRCPS are an iterative search through priority value and mode changes to activities in order to obtain a global or an approximately global optimal solution, through the annealing process.

Before explaining the process of fuzzy SA approaches, new notations are introduced as follows:

- $CurS_{\text{forward}}$ = the current solution obtained by fuzzy forward scheduling;
- $CurS_{\text{backward}}$ = the current solution obtained by fuzzy backward scheduling;
- $NewS_{\text{forward}}$ = a new solution generated from the current solution that is obtained by fuzzy forward scheduling;
- $NewS_{\text{backward}}$ = a new solution generated from the current solution that is obtained by fuzzy backward scheduling;

- *BestS* = The best solution found so far in any iteration;
- $\Delta$ $= \begin{cases} R(NewS_{\text{forward}}) - R(CurS_{\text{forward}}) & \text{if fuzzy forward scheduling is applied} \\ R(NewS_{\text{backward}}) - R(CurS_{\text{backward}}) & \text{if fuzzy backward scheduling is applied} \end{cases}$

As solutions (project completion times) generated are fuzzy numbers, to compare a fuzzy new neighboring solution and a fuzzy current solution, the ranking method that has been presented in Section 2.5 and Remark 2 of Section 5.2, is applied. $\Delta$ represents the difference of ranking indices of a new neighboring solution and the current solution by either in fuzzy forward or backward scheduling, indicating how a new generated solution is worse or better than the current solutions.

Figure 9.1 presents the logic flow of the two versions of fuzzy SA. Both fuzzy SA approaches consist of two main parts. One is the perturbation algorithm in which the main purpose is to alter priority values and modes of activities through perturbation operations, which has been mentioned in Section 5.1. The other is the fuzzy scheduling mechanism, used to schedule activities after the perturbation algorithm has been applied. The details of the fuzzy scheduling mechanism have been presented in Sections 5.3 and 5.4. This section explains how these two parts work together in either fuzzy SA approach to FMMRCPS.

## 9.7.1 Pure Fuzzy SA

Before processing pure fuzzy SA, the information on both the project and the fuzzy SA approach are required to be input. The determination of appropriate values for the parameters has been discussed in Section 9.6. Once information on both the project and the parameters has been input, two initial solutions (schedules) are generated through the fuzzy forward and backward scheduling, when the system randomly assigns both a mode and a priority value to each activity of a project, as shown in Figure 9.1.

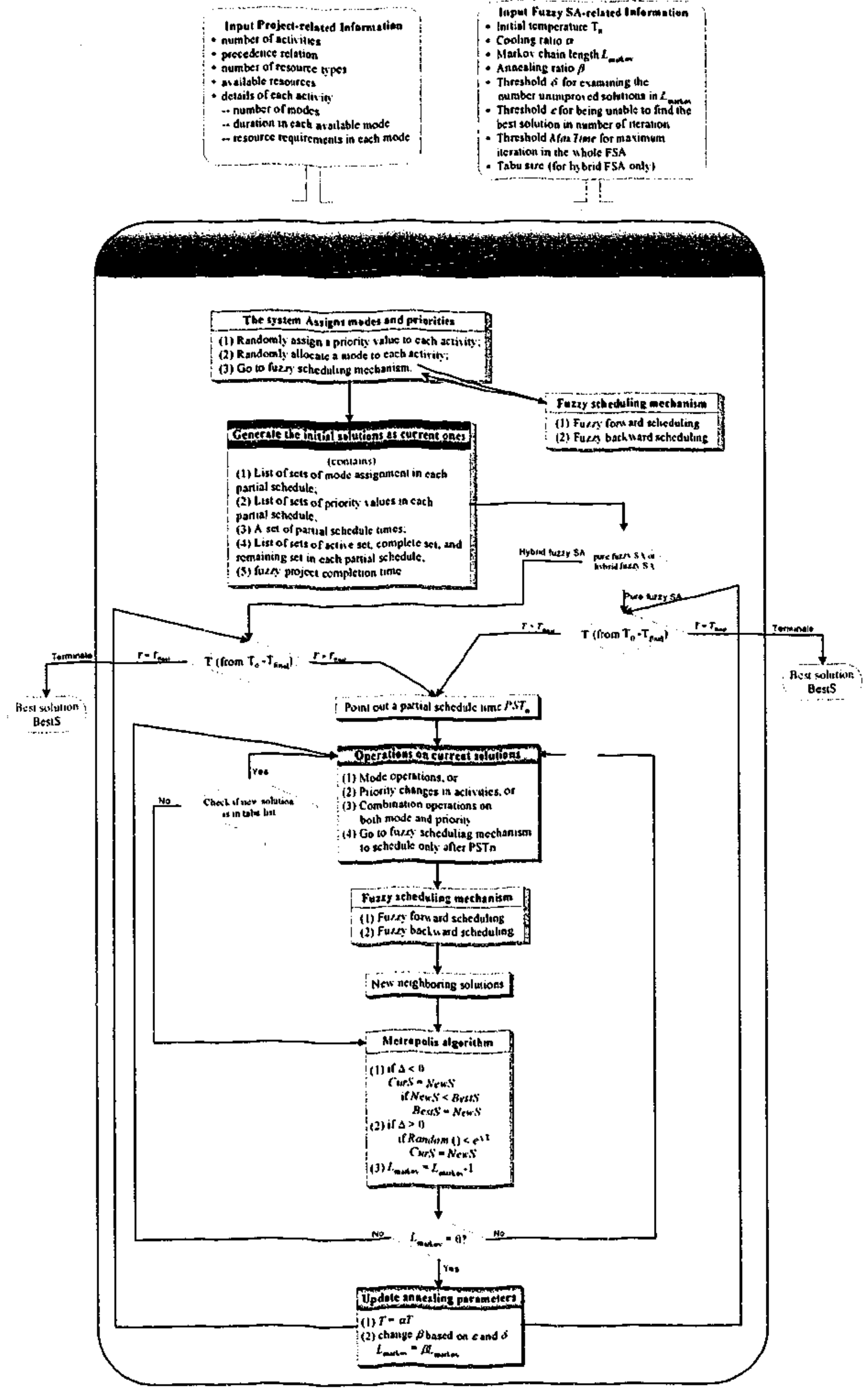

Input Project-related Information
• number of activities
• precedence relation
• number of resource types
• available resources
• details of each activity
-- number of modes
-- duration in each available mode
-- resource requirements in each mode
Input Fuzzy SA-related Information
• Initial temperature $T_0$
• Cooling ratio $\alpha$
• Markov chain length $L_{markov}$
• Annealing ratio $\beta$
• Threshold $\delta$ for examining the number unimproved solutions in $L_{markov}$
• Threshold $\varepsilon$ for being unable to find the best solution in number of iteration
• Threshold *Max Time* for maximum iteration in the whole FSA
• Tabu size (for hybrid FSA only)
The system Assigns modes and priorities
(1) Randomly assign a priority value to each activity;
(2) Randomly allocate a mode to each activity;
(3) Go to fuzzy scheduling mechanism.
Fuzzy scheduling mechanism
(1) Fuzzy forward scheduling
(2) Fuzzy backward scheduling
Generate the initial solutions as current ones
(contains)
(1) List of sets of mode assignment in each partial schedule;
(2) List of sets of priority values in each partial schedule,
(3) A set of partial schedule times;
(4) List of sets of active set, complete set, and remaining set in each partial schedule,
(5) fuzzy project completion time
Hybrid fuzzy SA
pure fuzzy SA or hybrid fuzzy SA
Pure fuzzy SA
$T > T_{final}$
T (from $T_0$ - $T_{final}$)
$T = T_{final}$
Terminate
Best solution BestS
Terminate
$T = T_{final}$
T (from $T_0$ - $T_{final}$)
$T > T_{final}$
Best solution BestS
Point out a partial schedule time $PST_n$
Operations on current solutions
(1) Mode operations, or
(2) Priority changes in activities, or
(3) Combination operations on both mode and priority
(4) Go to fuzzy scheduling mechanism to schedule only after PSTn
Yes
No
Check if new solution is in tabu list
Fuzzy scheduling mechanism
(1) Fuzzy forward scheduling
(2) Fuzzy backward scheduling
New neighboring solutions
Metropolis algorithm
(1) if $\Delta < 0$
$CurS = NewS$
if $NewS < BestS$
$BestS = NewS$
(2) if $\Delta > 0$
if $Random() < e^{\Delta T}$
$CurS = NewS$
(3) $L_{markov} = L_{markov} - 1$
No
$L_{markov} = 0$?
No
Yes
Update annealing parameters
(1) $T = \alpha T$
(2) change $\beta$ based on $\varepsilon$ and $\delta$
$L_{markov} = \beta L_{markov}$

Figure 9.1 Logic flow of two versions of fuzzy SA

The solution presentation in Section 9.3 has been specially designed for easily manipulating neighborhood generation using fuzzy SA, and the current solution obtained by fuzzy forward scheduling can only apply fuzzy forward scheduling to generate neighborhoods whereas the current solution gained through fuzzy backward scheduling can only be allowed to use fuzzy backward scheduling to produce neighborhoods. Therefore, the two initial solutions obtained through both fuzzy forward and backward scheduling become two current solutions, $CurS_{\text{forward}}$ and $CurS_{\text{backward}}$ at the beginning of fuzzy SA. The better solution (in terms of minimising the fuzzy project completion time) is taken from these two initial solutions, and is temporarily placed in the variable, *BestS* of the system.

In the section of Figure 9.1 labelled "generate the initial solutions as current ones", the initial solutions should contain the following information:

(1) a list of sets of modes allocated to the activities in each partial schedule;
(2) a list of sets of priority values assigned to the activities in each partial schedule;
(3) a set of partial schedule times in each stage of a whole schedule;
(4) a list of sets of an active set, a complete set and a remaining set of each partial schedule;
(5) fuzzy project completion time.

All the information must be stored in the system as it is required for a series of neighboring generation.

As shown in Figure 9.1, the initial value $T_0$ of temperature should be set appropriately high in the beginning of fuzzy SA in order to allow enough time for the annealing process. The neighbouring solution can be obtained through adequate perturbation from the initial solution. In my fuzzy SA, the system randomly picks up a partial schedule time from each of two initial solutions as a perturbing fuzzy time point in the two current solutions respectively. To perturb the current solutions, several operations may be made on these two current solutions as shown in Figure 9.1: (1) mode operation in various ways on activities, or (2) only priority value changes in different activities, or (3) operation on both mode and priority values simultaneously. Once the operation has been made on the current solutions, two new solutions are generated in parallel by only scheduling some activities from the fuzzy time point using fuzzy forward and backward scheduling.

Metropolis algorithm in Figure 9.1 indicates the way by which the algorithm accepts the new solution as the current solution. When the difference $\Delta$ between the ranking indices of a new neighbouring solution and the current solution is negative, the new solution moves downhill in terms of minimising the fuzzy project completion time. This improved solution, of course, is accepted as the current solution. The new solution will also supersede the position of the best solution *BestS* if the ranking index of the new solution is smaller than that of the current best solution stored in memory. However, sometimes a new neighbourhood moves uphill for a worse solution (where $\Delta > 0$), and the worse solution may also be accepted by Metropolis' criterion $e^{\Delta/T}$. If the criterion is greater than the random number between 0 and 1 selected by the system, the criterion is satisfied and the new solution becomes the current solution. This is permitted in order to explore more different search areas, and to avoid being trapped in local optima. When the cooling temperature is high, more low-quality solutions will be accepted in early stages. This allows uphill moves to explore or dig more search space. As the temperature is decreased, the probability of accepting lower quality solutions will be reduced and the search becomes more focused on better ones.

Once the two new neighboring solutions have been processed in the annealing procedure after they have been obtained by fuzzy forward and backward scheduling respectively, Markov chain length, $L_{markov}$ is reduced by one. Such a procedure will be repeated for the number of iterations specified by $L_{markov}$ until its length becomes zero. Thus, the Metropolis algorithm is attempting to generate and inspect a number of neighboring solutions in $L_{markov}$ at the same temperature $T$ in order to dig for a better solution. If $L_{markov} = 0$, some annealing parameters are required to be updated, as shown in the last procedure box of Figure 9.1 in order to get ready for the next new temperature level. The fuzzy SA will be terminated if one of the stopping criteria has been satisfied or the temperature level has reached the lowest temperature $T_{final}$. When fuzzy SA is terminated, the solution stored in BestS of the system becomes the optimal or approximately optimal solution to FMMRCPS.

### 9.7.2 Hybrid Fuzzy SA

To prevent the reproduction of the same solutions generated from recent searches, tabu mechanism is incorporated into fuzzy SA. Tabu for short-term memory, referred to as *tabu list*, is employed in the hybrid fuzzy SA. The tabu list keeps track of solution attributes for representing previous searches in order to avoid cycling in the same search space. The attributes of recently visited solutions are labelled as *tabu active* and solutions containing these attributes become tabu. The memory available for holding the set of previous solutions is called the tabu size. The size of tabu chosen may affect the performance of the hybrid fuzzy SA. A small size of tabu may not be sufficient to contain information on the attributes of recent searches, whereas a large size of tabu requires a large amount of computational time to examine whether each new solution has any attributes rendering it forbidden.

Apart from the tabu mechanism, hybrid fuzzy SA is similar to pure fuzzy SA. Fuzzy SA incorporating a short-term memory function can forbid transitions in solutions if the attributes are already contained in the solutions generated. Therefore, the merit of hybrid fuzzy SA is to ensure that the search procedure does not revisit solutions previously generated. In addition, the tabu mechanism expands the search for better neighboring solutions although sometimes worse solutions may be generated. Each time a new neighboring solution is generated that is not forbidden, it moves to the top of tabu list and the bottom solution in the tabu list is dropped off. The size of tabu can be selected either fixed or changeable in the system. In the case of changeable tabu size, the size is variable, depending on the status of the current search. If new neighboring solutions are generated that have not recently appeared in the tabu list, the size of tabu will be reduced as the current search is in a definitely different search area. If newly generated neighboring solutions have appeared frequently in the tabu list, the size of tabu needs to be increased as the search may be moving towards a previous search area.

## 9.8 Computation Comparison

To evaluate the two versions of fuzzy SA experimentally, five different sizes of projects with 50, 100, 150, 200, and 250 activities are randomly generated by the system, and activities in each project have at most three modes. Three renewable resource types are consumed in the five projects. The cooling ratio $\alpha$ for temperature changes, and the annealing ratio $\beta$ for $L_{markov}$ are both set to 0.8. The initial value of temperature is set to 1,200. The initial value of Markov chain length is chosen to be 200. For this experiment, *MaxTime* and $\delta$ are set to 3,500 and 150 respectively for all the project sizes.

Each project has been run 100 times to determine how far the average solution differs from the best solution, thus verifying the deviation from the best solution. In order to apply the average deviation from the best solution, all results are converted into ranking indices because of fuzzy project completion times. Table 9.1 lists the best and worst solutions, and the average deviation for each project size.

Table 9.1 Experiment in two versions of fuzzy SA

| Project size | Pure fuzzy SA | Hybrid fuzzy SA | Average deviations from the best one | |
|---|---|---|---|---|
| | Best solution | Best solution | Pure fuzzy SA | Hybrid fuzzy SA |
| 50 | (68, 85, 96, 105) | (77, 90, 105,112) | 18.90 | 17.75 |
| 100 | (90, 110, 130, 160) | (100, 120, 139, 170) | 19.24 | 18.25 |
| 150 | (112, 132, 145, 145) | (102, 129, 134, 139) | 20.35 | 19.28 |
| 200 | (122, 152, 222, 243) | (134, 169, 230, 263) | 22.50 | 21.40 |
| 250 | (146, 189, 225, 254) | (139, 179, 210, 249) | 24.55 | 23.42 |

As noticed in Table 9.1, the best solutions obtained from hybrid fuzzy SA are better than those obtained from pure fuzzy SA. This is because hybrid fuzzy SA forbids the search to visit the areas that have been visited recently in order to explore other areas for searching better solutions.

Table 9.1 also lists the deviations from the best solution both for pure and hybrid fuzzy SA. The deviations range from 19 to 25 in pure fuzzy SA and from 18 to 23 in hybrid fuzzy SA for the projects with 50, 100 and 150 activities. However, the deviation gaps becomes large when sizes of the projects have 200 and 250 activities in both pure and hybrid fuzzy SA. The reason is that the values of key parameters have been set the same for all sizes of the five projects in the experiment. The experiment indicates that the existing values for the key parameters seems suitable for the project with 150 activities whereas the solutions are not much improved for the projects with 50 and 100 activities by the current parameter setting. The values set for the parameters, particularly for the temperature and Markov chain length, are required to be appropriately large to have adequate time to conduct the annealing process for the search of good solutions. The experiment indicates that the key parameter setting is essential to the overall performance of either pure or hybrid fuzzy SA.

## 9.9 Concluding Remarks

FMMRCPS with fuzzy activity duration times is a difficult combinatorial optimisation problem, and it becomes more complex as the project size increases. As such, no exact algorithms can provide efficient solutions. Metaheuristic approaches incorporating fuzzy set theory are an effective way of obtaining an approximate globally optimal solution for FMMRCPS under the fuzzy environment.

To develop an efficient and effective metaheuristic approach, two versions of fuzzy SA approaches to FMMRCPS have been developed. The solution presentation is an important element for the efficient and effective performance of fuzzy SA. The solution presentation I have designed always guarantees the generation of feasible neighbouring solutions. In addition, this solution presentation is perturbed easily for generating neighbourhoods, and it saves computational time in producing a new neighbouring solution from the current one as only partially scheduling is required.

The underlying concept of both pure and hybrid fuzzy SA presented in this chapter for solving FMMRCPS problems under fuzziness is simple and straightforward. Both approaches are suitable for solving practical FMMRCPS applications. As indicated in the experiment conducted, the key parameter setting has a great influence on the result of the fuzzy project completion time. As such, in practical applications the key parameters in fuzzy SA should be carefully set to obtain a good solution for minimising the fuzzy project completion time.

# Chapter 10

# System Development for FMMRCPS

## 10.1 Introduction

Developing a system for handling FMMRCPS is an important work as part of my PhD research where four metaheuristic approaches are built in the system: (a) fuzzy GA alone, (b) fuzzy GA with tabu, (c) fuzzy SA alone, and (d) fuzzy SA with tabu. For the development of individual approaches to FMMRCPS, the system is an indispensable means of analysing and testing these approaches to help improve these approaches. In addition, different values of parameters for individual approaches greatly impact on their quality of performance. The system enables the conduct of extensive experiments on parameter settings under different conditions, making general recommendations about choosing appropriate values of parameters prior to implementing individual approaches. Furthermore, the system allows much insight to be gained into the characteristics of individual approaches by the way of experiments. This helped me get an idea about developing two hybrid approaches, fuzzy GA with tabu and fuzzy SA with tabu as they could perform better than an approach alone. Without developing the system, it may be difficult to achieve good approaches for solving FMMRCPS efficiently and effectively.

Project scheduling is widely used in organisations and industry. The system for FMMRCPS should not be limited to use on a standalone computer, but importantly should be shared by different departments across an organisation on the network base. For this reason, I chose a new generation of object-oriented programming language, VB.NET, because applications written in this language are able to be run on any computers and operation systems. Applications written in VB.NET can be distributed by servers and shared by multiple users across the network. In addition, the system written in VB.NET is much more secure than previous versions (Cornell and Morrison

2002). These reasons motivated me to use VB.NET for developing the system. I will now briefly explain the structures and functions of the system.

## 10.2 System Design

In my PhD studies, the system for FMMRCPS is not only used to implement the metaheuristic approaches for solving FMMRCPS, but more importantly, assists in improving individual approaches during their development through experiments and examination from sample projects in the system. For the purpose of facilitating the development of these individual approaches, some parts of the system for carrying out the tasks of data input, experiment, and result summaries have to be completed in design before developing individual metaheuristic approaches. In addition, the parts related to the approaches are often required to be modified in order to achieve improvements during their development. Moreover, the common parts of the system, such as fuzzy forward and backward scheduling, fuzzy arithmetic, and fuzzy ranking, are frequently shared by a number of developed approaches. To meet these concerns, the methodology of modular design is a suitable way of treating these parts as independent tasks in system development. Each module has its own unique task or function. The advantage of applying this methodology in system development is that any modification on a particular module will not affect other modules in the system. The other benefit of using modular design is that some modules are reusable because common tasks that are treated as modules, are used a number of times in individual developed approaches.

The system development for implementing FMMRCPS is a very complex programming process involving a number of complicated procedures for processing activities with precedence relationships and resource constraints through different stages. To facilitate logical understanding and maintenance of the system, the entire structure of the system uses traditional structure programming techniques along with the modular concepts for each individual task (Budgen 2003).

During the system development, each module has been verified for correctness by application to the sample data before the modules were assembled together. Thus, the system will reliably implement any of the four metaheuristic approaches in handling FMMRCPS. The system for FMMRCPS is implemented using four stages as follows:

(a) the input of data for a project manually or for data to be generated by the system; in the following stage;

(b) the input of values of parameters for the selected approach;

(c) implementing the selected approach;

(d) result output as either a single result or the results of statistical analyses.

## 10.3 Structure of the System

Figure 10.1 shows the overall structure of the system. The system is decomposed into five main parts: (a) interfaces for the user input, (b) individual main structures for each approach built into the system, (c) perturbation algorithms for manipulating activity priorities and modes, (d) fuzzy scheduling mechanism, and (e) output interfaces.

During the system development, most situations that might occur in project scheduling are taken into consideration. In terms of activity duration times, the system provides several flexible options to handle either fuzzy or crisp times, or times containing both. The system also provides choices for dealing with either single mode RCPS or multiple mode RCPS. To set up project instances, one, or a set of projects, can be generated manually by the user or generated automatically by the system. In addition, the system has the facilities for statistical analysis for evaluating the approaches that are developed.

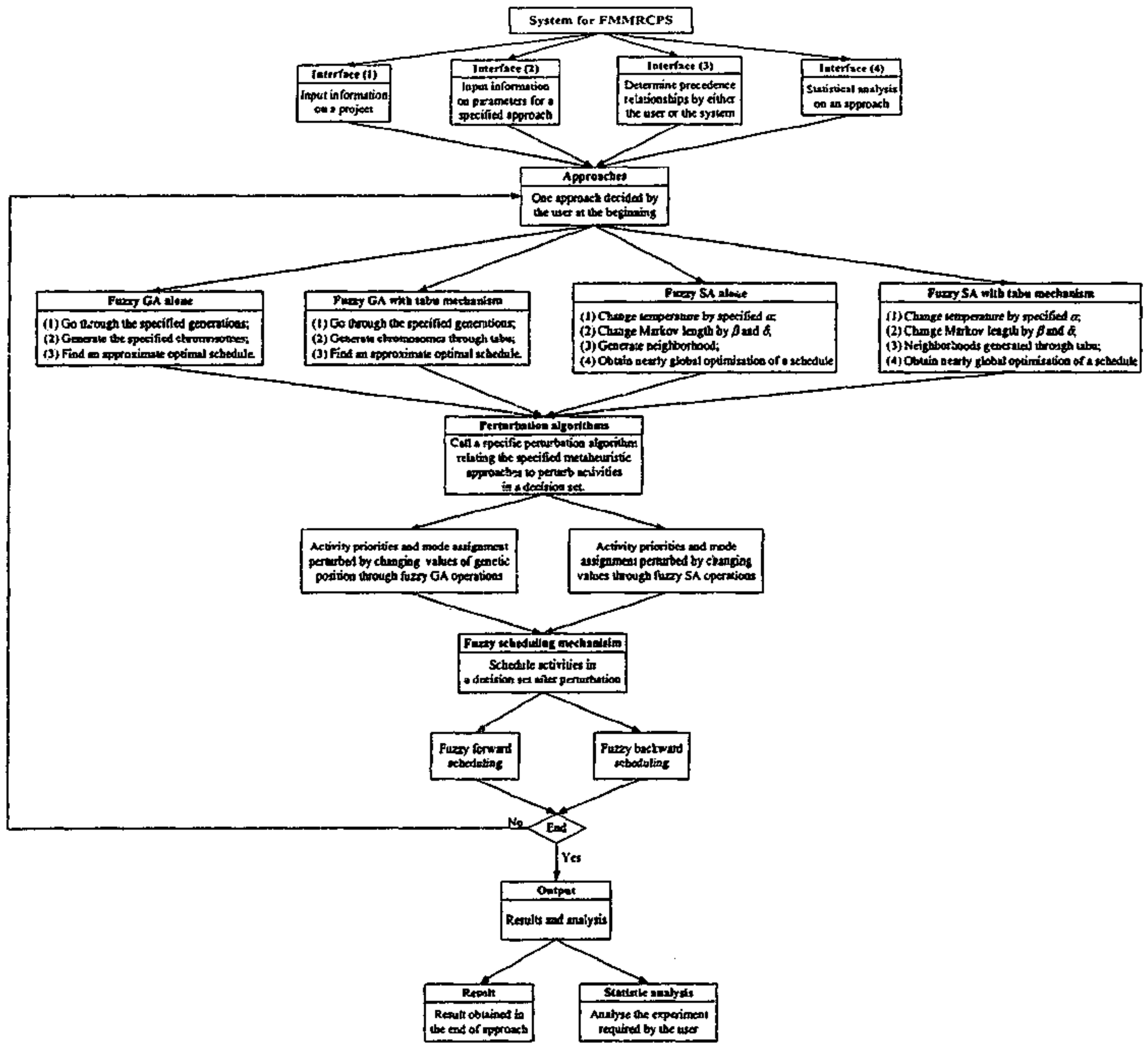
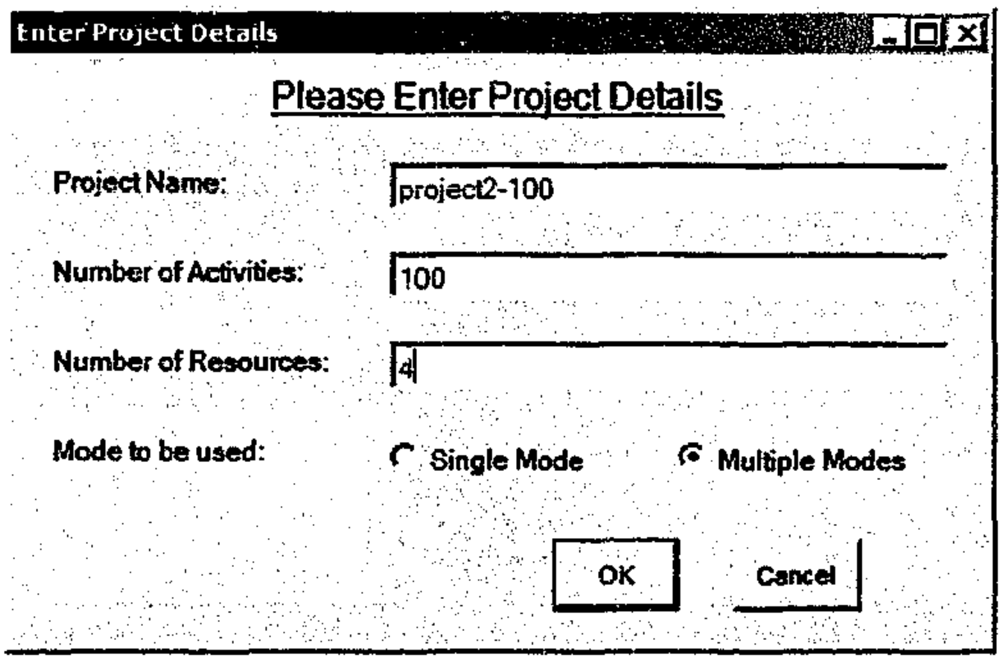

System for FMMRCPS
Interface (1)
Input information on a project
Interface (2)
Input information on parameters for a specified approach
Interface (3)
Determine precedence relationships by either the user or the system
Interface (4)
Statistical analysis on an approach
Approaches
One approach decided by the user at the beginning
Fuzzy GA alone
(1) Go through the specified generations;
(2) Generate the specified chromosomes;
(3) Find an approximate optimal schedule.
Fuzzy GA with tabu mechanism
(1) Go through the specified generations;
(2) Generate chromosomes through tabu;
(3) Find an approximate optimal schedule.
Fuzzy SA alone
(1) Change temperature by specified α;
(2) Change Markov length by β and δ;
(3) Generate neighborhood;
(4) Obtain nearly global optimization of a schedule
Fuzzy SA with tabu mechanism
(1) Change temperature by specified α;
(2) Change Markov length by β and δ;
(3) Neighborhoods generated through tabu;
(4) Obtain nearly global optimization of a schedule
Perturbation algorithms
Call a specific perturbation algorithm relating the specified metaheuristic approaches to perturb activities in a decision set.
Activity priorities and mode assignment perturbed by changing values of genetic position through fuzzy GA operations
Activity priorities and mode assignment perturbed by changing values through fuzzy SA operations
Fuzzy scheduling mechanisim
Schedule activities in a decision set after perturbation
Fuzzy forward scheduling
Fuzzy backward scheduling
No
End
Yes
Output
Results and analysis
Result
Result obtained in the end of approach
Statistic analysis
Analyse the experiment required by the user

Figure 10.1 The system integrating four metaheuristic approaches to FMMRCPS

### 10.3.1 User Interfaces for Data Input

In the system, there are three interfaces for data input: (a) data on activities of a project, (b) parameters on individual approaches, and (c) experimental requirements.

Before running any approaches built into the system, information about a project is required to be input. The size of the project, and whether a single mode RCPS or multiple mode RCPS need to be determined as shown in Figure 10.2. For single mode project scheduling, mode assignment is not required in any individual approaches. However, where multiple modes are chosen, the number of modes needs to be decided for individual activities by either the user or by the system. To avoid tedious data input, the system can assign the number of modes to activities, as shown in Figure 10.3. Once the number of modes is assigned to an activity, the activity will be both highlighted and ticked automatically by the system until all activities are assigned to the number of modes.


Enter Project Details

Please Enter Project Details

Project Name: project2-100

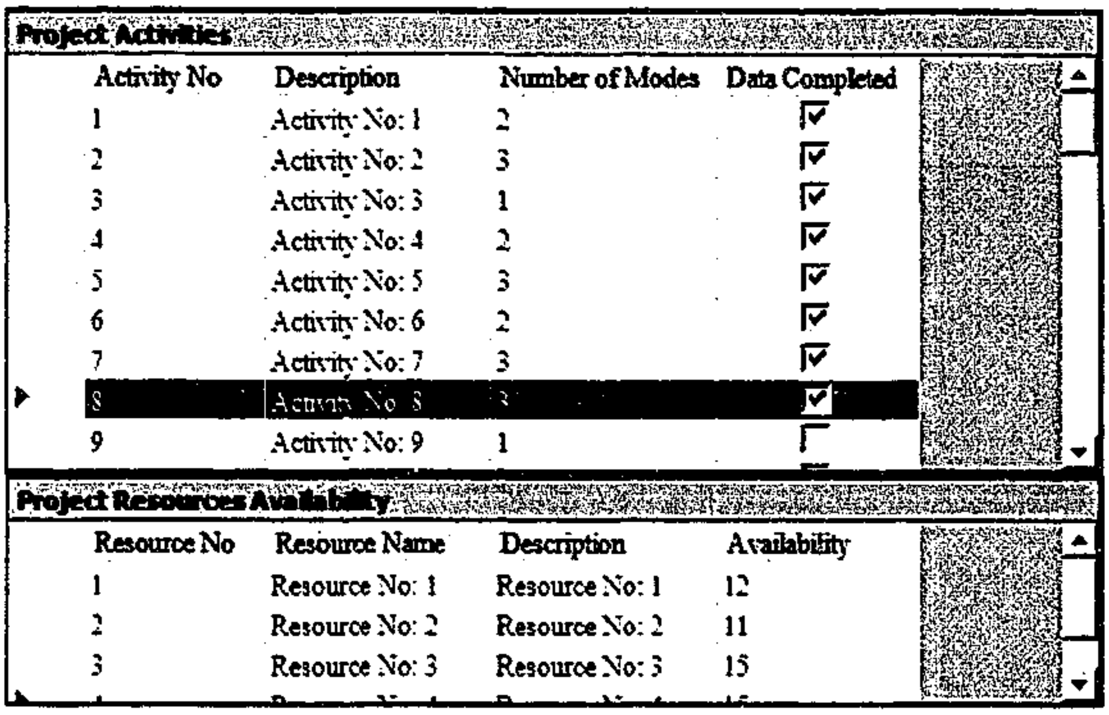
Number of Activities: 100

Number of Resources: 4

Mode to be used: Single Mode [✓ Multiple Modes]

Figure 10.2 Interface for determining the size of the project and mode options

Scheduling activities in a project is the most difficult part in the programming. Whether activities are eligible to be scheduled relies on their precedence relationship as well as on resource availabilities. To set up their precedence relationships straightforwardly and effectively during scheduling, a tree structure, along with the data sets of Boolean type, is applied. The precedence relationships can be input manually by the user in the manner shown in Figure 10.4. To protect the logical correctness of their relationships, the system has a protection mechanism to ensure that the activity number must always be greater than that of its predecessors.

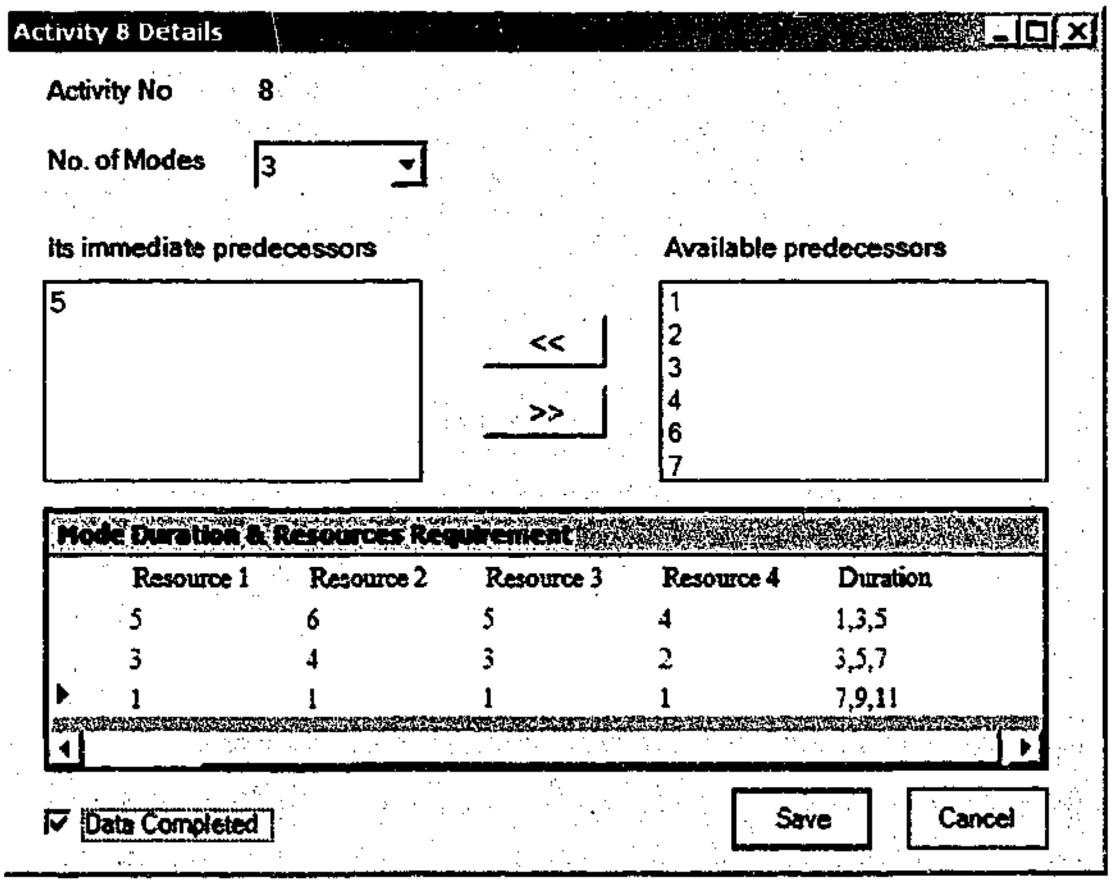

Project Activities

| Activity No | Description | Number of Modes | Data Completed |
|---|---|---|---|
| 1 | Activity No: 1 | 2 | ✓ |
| 2 | Activity No: 2 | 3 | ✓ |
| 3 | Activity No: 3 | 1 | ✓ |
| 4 | Activity No: 4 | 2 | ✓ |
| 5 | Activity No: 5 | 3 | ✓ |
| 6 | Activity No: 6 | 2 | ✓ |
| 7 | Activity No: 7 | 3 | ✓ |
| 8 | Activity No: 8 | 3 | ✓ |
| 9 | Activity No: 9 | 1 | |

Project Resources Availability

| Resource No | Resource Name | Description | Availability |
|---|---|---|---|
| 1 | Resource No: 1 | Resource No: 1 | 12 |
| 2 | Resource No: 2 | Resource No: 2 | 11 |
| 3 | Resource No: 3 | Resource No: 3 | 15 |

Figure 10.3 The system assigns the numbers mode to activities


Activity 8 Details

Activity No 8

No. of Modes 3

Its immediate predecessors: 5

Available predecessors: 1, 2, 3, 4, 6, 7

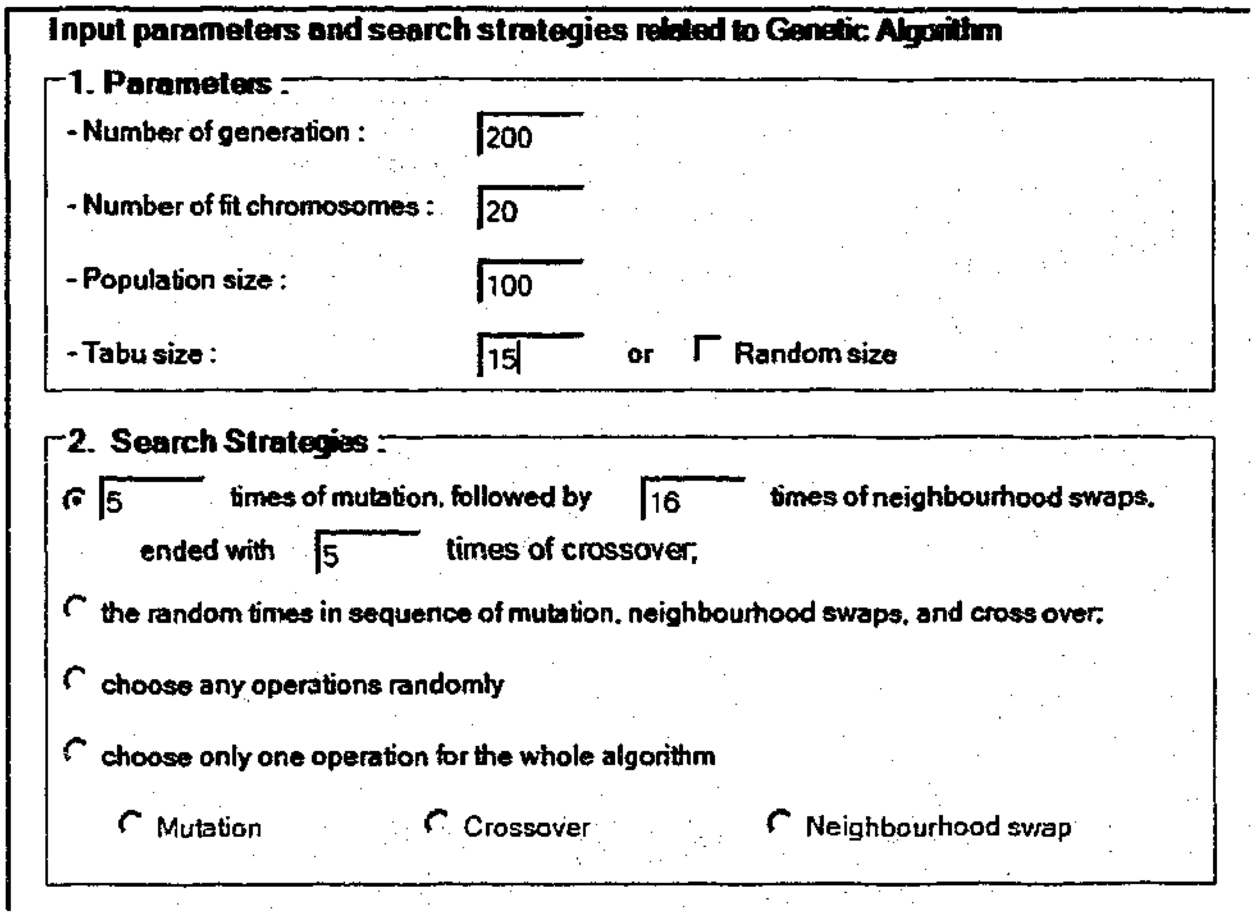
Mode Duration & Resources Requirement

| Resource 1 | Resource 2 | Resource 3 | Resource 4 | Duration |
|---|---|---|---|---|
| 5 | 6 | 5 | 4 | 1,3,5 |
| 3 | 4 | 3 | 2 | 3,5,7 |
| 1 | 1 | 1 | 1 | 7,9,11 |

☑ Data Completed

Figure 10.4 Activity details in each mode and its precedence relationships

The system provides four metaheuristic approaches. Once an approach is chosen, for example, fuzzy GA with tabu, the values of parameters and search strategies on activity priorities are required to be determined as shown in Figure 10.5. Different values given in the approach will impact on the performance of the approach. These will be exhaustively explained through intensive experiments in Chapter 11.


**Input parameters and search strategies related to Genetic Algorithm**

**1. Parameters :**

- Number of generation : 200
- Number of fit chromosomes : 20
- Population size : 100
- Tabu size : 15 or [ ] Random size

**2. Search Strategies :**

[✓ 5 times of mutation, followed by 16 times of neighbourhood swaps, ended with 5 times of crossover;]

( ) the random times in sequence of mutation, neighbourhood swaps, and cross over;

( ) choose any operations randomly

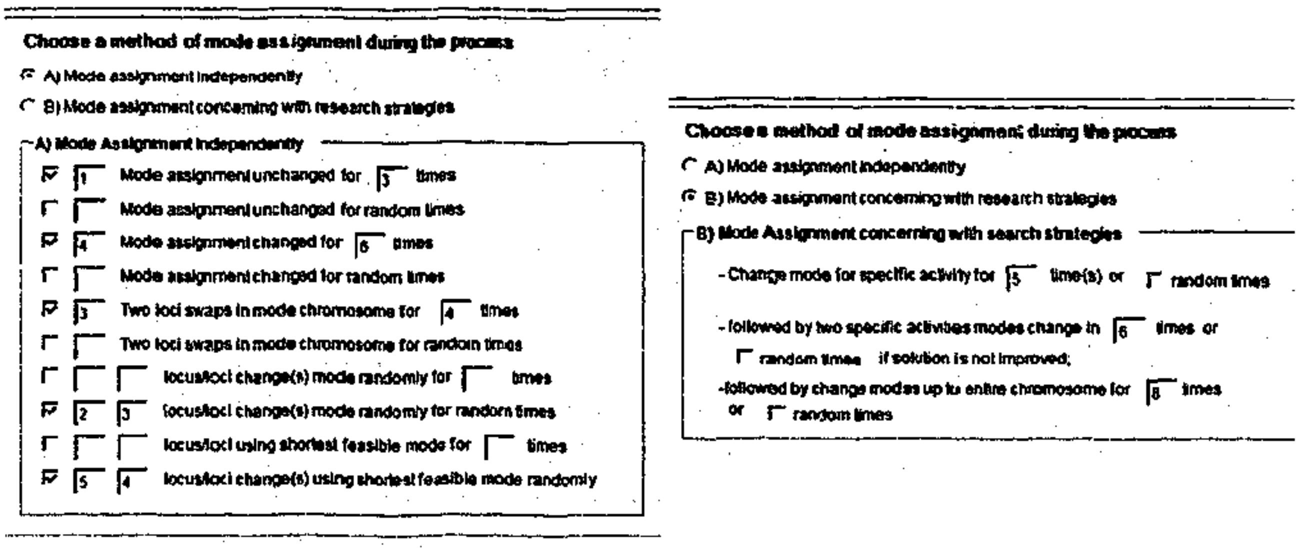
( ) choose only one operation for the whole algorithm

( ) Mutation ( ) Crossover ( ) Neighbourhood swap

Figure 10.5 Interface for input values of parameters of GA with tabu approach

To deal with multiple modes in fuzzy GA with tabu, genetic operations on mode assignment are required. To broaden operations on mode assignment, there are two main options for manipulating mode assignment. Option A, referred to as "mode assignment independent" in Figure 10.6, has nothing to do with the current search status of the approach, and the user can choose one or several selections arbitrarily. Option B, referred to as "mode assignment concerned with search strategies", is a manner where several specifically designed operations are made on mode assignment for specific sequences based on the search status, and the user is only required to specify the number of times applied to each operation, as shown in Figure 10.6.

For conducting experiments for evaluating an approach, the approach may need to run a number of times, and the best solution can be obtained for each run. The different types of deviations such as average, maximum and standard deviations can be calculated by the system. The optimum solution is selected from a set of best solutions that are obtained from each run. The average CPU time is the average time spent in implementing an approach. The system provides several options for experimental analysis, shown as in Figure 10.7.



**Choose a method of mode assignment during the process**

[✓ A) Mode assignment independently]

B) Mode assignment concerning with research strategies

**A) Mode Assignment Independently**

- ✓ 1 Mode assignment unchanged for 3 times
- ☐ Mode assignment unchanged for random times
- ✓ 4 Mode assignment changed for 6 times
- ☐ Mode assignment changed for random times
- ✓ 3 Two loci swaps in mode chromosome for 4 times
- ☐ Two loci swaps in mode chromosome for random times
- ☐ locus/loci change(s) mode randomly for times
- ✓ 2 3 locus/loci change(s) mode randomly for random times
- ☐ locus/loci using shortest feasible mode for times
- ✓ 5 4 locus/loci change(s) using shortest feasible mode randomly

**Choose a method of mode assignment during the process**

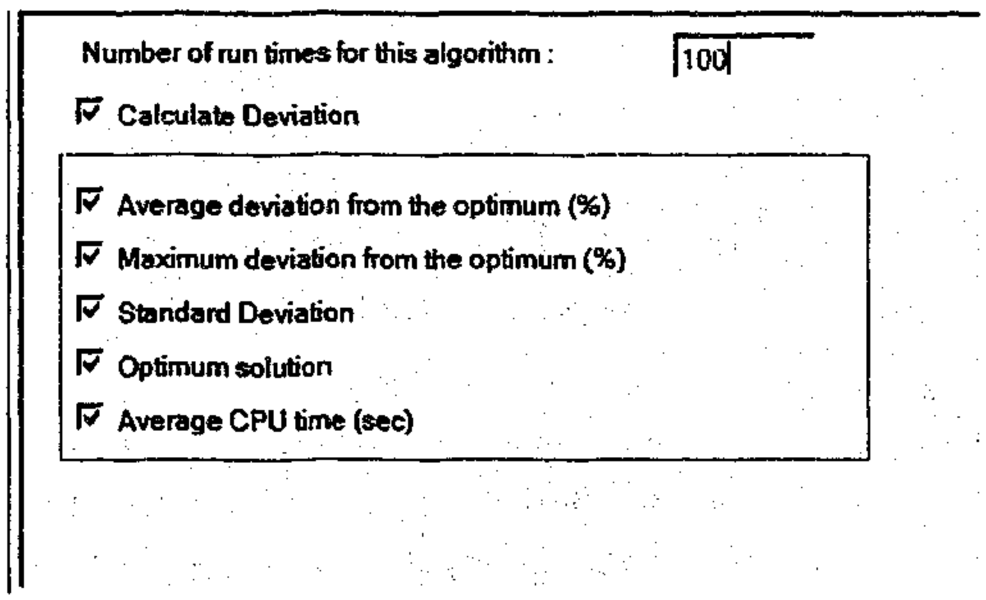
A) Mode assignment independently

[✓ B) Mode assignment concerning with research strategies]

**B) Mode Assignment concerning with search strategies**

- Change mode for specific activity for 5 time(s) or ☐ random times
- followed by two specific activities modes change in 6 times or ☐ random times if solution is not improved;
- followed by change modes up to entire chromosome for 8 times or ☐ random times

Figure 10.6 Operations on mode assignment in fuzzy GA with tabu



Number of run times for this algorithm : 100

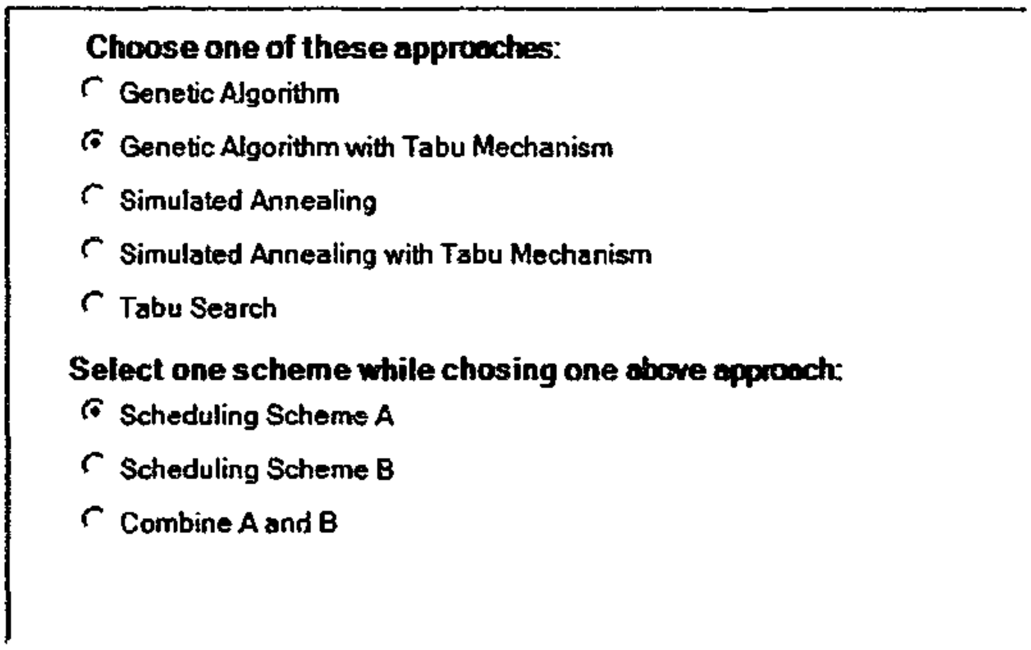
✓ Calculate Deviation

- ✓ Average deviation from the optimum (%)
- ✓ Maximum deviation from the optimum (%)
- ✓ Standard Deviation
- ✓ Optimum solution
- ✓ Average CPU time (sec)

Figure 10.7 Options of experimental analysis

## 10.3.2 Main Structure of Individual Metaheuristic Approaches

As mentioned in Section 10.1, four metaheuristic approaches are built into the system. Any of these approaches can be chosen at a time. Figure 10.8 is an example where the choice is GA with tabu. Any selected approach is incorporated into either scheduling scheme A or B or a combination of both. Scheme A is defined as strict scheduling. That is, other eligible activities are not allowed to be scheduled if resources are insufficient for one selected eligible activity. The details of Scheme A are presented in Definition 6 of Chapter 5. Scheme B is nonstrict scheduling where the eligible activity of the next highest priority can be scheduled if resources are insufficient for the first highest priority of the selected activity. The explanation of Scheme B is in Definition 7 of Chapter 5. Figure 10.8 shows the GA with tabu incorporating Scheme A.


Choose one of these approaches:
- ○ Genetic Algorithm
- [✓ Genetic Algorithm with Tabu Mechanism]
- ○ Simulated Annealing
- ○ Simulated Annealing with Tabu Mechanism
- ○ Tabu Search

Select one scheme while chosing one above approach:
- [✓ Scheduling Scheme A]
- ○ Scheduling Scheme B
- ○ Combine A and B

Figure 10.8 Approach selection

In the system, the four approach modules contain the main structure of the corresponding approach. In the modules of both fuzzy GA and fuzzy GA with tabu, the outer loop controls the required number of generations and the inner loop manages the specified population size in each generation. In the inner loop, to generate chromosomes, the module of the perturbation algorithm is called first. This module functions to change both activity priorities and modes in their genetic positions on their two sub-chromosomes. The details of chromosome design for FMMRCPS are presented in Section 8.3. Once these two sub-chromosomes are generated, the module of the scheduling mechanism will be called to get relevant scheduling information for the third sub-chromosome. This process is repeated until the outer loop reaches the final generation. In addition to fuzzy GA requirements, fuzzy GA with tabu has the extra facility of preventing the generation of chromosomes that have been recently generated. This mechanism is in the main structure of fuzzy GA with tabu.

In the other two approach modules of fuzzy SA and fuzzy SA with tabu, the outer loop controls the temperature changes under the cooling ratio $\alpha$. The inner loop undertakes neighbourhood generations in the specified Markov length for the annealing process at the same temperature. Whilst generating neighbourhoods from the current solution, the perturbation algorithm module is called to change values of activity priorities and modes, followed by calling the fuzzy scheduling mechanisms module. This module is to generate neighbouring solutions. In the simulated annealing process, the Markov length is often changeable depending on the status of neighbourhood generations in the previous temperature level. The whole process is terminated when the preset stopping criteria are satisfied or the outer loop attains a sufficiently lower temperature. Fuzzy SA with tabu differs from fuzzy SA in only one aspect. That is, fuzzy SA with tabu has a mechanism to forbid generating neighbouring solutions from recently generated solutions.

### 10.3.3 Perturbation Algorithms

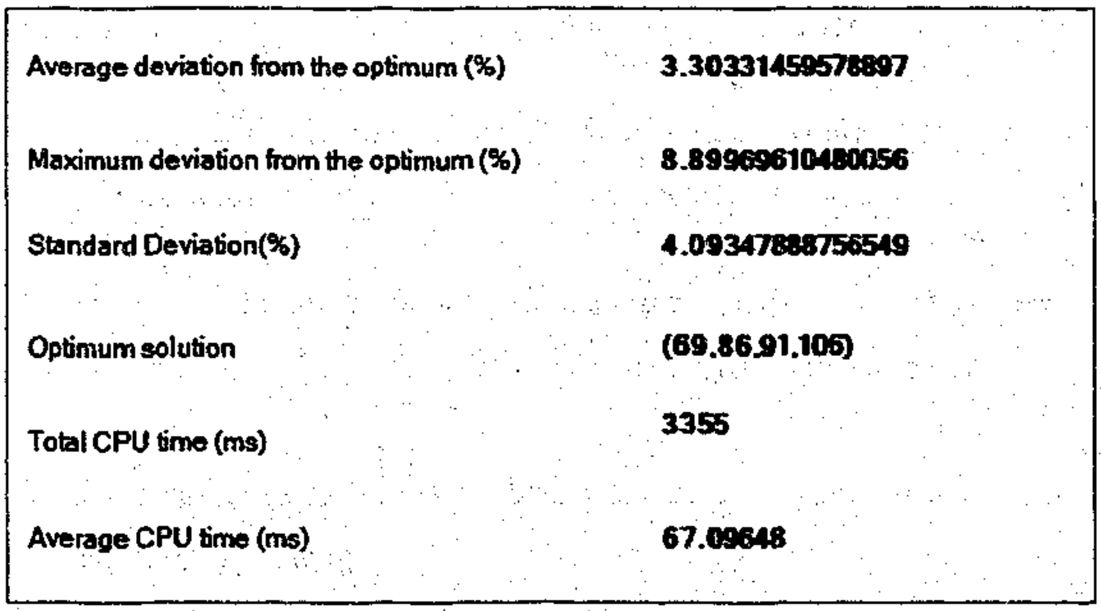
In perturbation algorithms, there are two perturbation modules: (a) dealing with fuzzy GA and fuzzy GA with tabu, and (b) dealing with fuzzy SA and fuzzy SA with tabu, as shown in Figure 10.1. In the module for perturbation on activities of fuzzy GA based approaches, operations of mutation, crossover and neighbourhood swaps are applied to chromosomes in order to generate new chromosomes. In the other module of perturbation on activities for fuzzy SA based approaches, activity priorities and modes can be changed randomly or specified by the user.

## 10.3.4Fuzzy Scheduling Mechanism

The fuzzy scheduling mechanism contains two sub-modules: (a) fuzzy forward scheduling module and (b) fuzzy backward scheduling module. When fuzzy forward or backward scheduling is conducted, four sets are used, mainly: (a) decision set, (b) active set, (c) complete set, and (d) remaining set. The details of these sets are referred to Section 5.2. Activities in these sets are often changed from each scheduled time. To identify activities in different sets at different scheduled times, data bound control is applied to each set at each scheduled time. Once perturbations are applied to both activity priorities and modes, activities will be scheduled by either fuzzy forward or backward scheduling.

## 10.3.5Output Interfaces

The system has two output interfaces: (a) single result, and (b) result with statistical analysis. For the single result, the output shows only the best schedule and CPU time for implementing an approach. The second output module is often used in the system. As the number of options in statistical analysis is selected in Figure 10.7, all required information is displayed after running this approach 100 times, as shown in Figure 10.9.



| | |
|---|---|
| Average deviation from the optimum (%) | 3.30331459578897 |
| Maximum deviation from the optimum (%) | 8.89969610480056 |
| Standard Deviation(%) | 4.09347888756549 |
| Optimum solution | (69.86.91.106) |
| Total CPU time (ms) | 3355 |
| Average CPU time (ms) | 67.09648 |

Figure 10.9 Output interface for statistic analysis

## 10.4 Salient Features of the System

The salient features of the system developed for implementing FMMRCPS can be summarised below:

- The system provides a user friendly interface;
- The system can handle both crisp and fuzzy activity duration times;
- The system can deal with both classical project scheduling with single mode and multi-mode project scheduling;
- The system can generate project information either manually or automatically;
- The system provides several metaheuristic approaches for solving FMMRCPS;
- The system allows intensive experiments to be conducted for evaluating individual approaches or for determining the values of parameters of individual approaches;
- The system can provide a crisp report about the details of experimental results and the relevant information such as CPU time for individual runs;
- The system not only can be run on standalone computers, but also can be distributed through networks and used by multiple users;
- The system is compact and allows common modules to be shared, because of the application of the modular concept during system development;
- The system structure is well organised and is easy to maintain with understandable coding;
- The system provides a number of open options for adding other approaches.

## 10.5 Concluding Remarks

The system development for handling FMMRCPS has helped me develop solid metaheuristic approaches through evaluation analyses provided by the system. The development of the system has facilitated the development of good approaches in my PhD research.

The system developed is a complete system with user friendly interfaces from data input to result output, and with the detailed statistical analysis for individual FMMRCPS approaches. The system is reliable because each module has been tested logically and technically from the sample project data, and the whole system has also been tested by several project instances with either crisp or fuzzy data or the combination of both. This system can solve any practically-sized FMMRCPS problem under most possible situations.

# Chapter 11

# Experimental Studies for Metaheuristic Approaches

## 11.1 Introduction

In this chapter, experimental studies are carried out to evaluate the four metaheuristic approaches I have developed: (a) fuzzy GA, (b) fuzzy GA with tabu, (c) fuzzy SA, and (d) fuzzy SA with tabu. These experiments have two major purposes: (a) to determine appropriate values for key parameters for providing better performance in individual approaches, and (b) to use appropriate values of key parameters obtained in experiments for evaluating these four approaches with a comparative analysis. All experiments are performed on a Pentium IV personal computer with 1.59 GHz clock-pulse and 256 MB of RAM, using the FMMRCPS system I have developed in VB.NET. Five multi-mode projects are generated by the system with sizes of 25, 50, 100, 150 and 200 activities respectively, which are used for experimental analysis of the approaches. The network diagrams representing precedence relationships of activities are listed in Appendix A.

To evaluate the performance of individual approaches in view of their randomised nature, several types of deviations from the optimum solution are applied to the experiments whilst a comparative analysis is carried out. The optimum solution considered here is the shortest fuzzy project completion time selected from the number of best solutions obtained after repeatedly running an approach for a number of times. The deviations include the following:

(a) Average deviation from the optimum solution (AD):

The average deviation is the average difference between the optimum solution $(\tilde{PC}_{optimum})$ and the best solution, $(\tilde{PC}_i)$ obtained for an individual run after running $N$ times for an approach. As the solutions are represented as fuzzy project completion times, they are difficult to compare directly. Fuzzy ranking indices are applied in calculating the deviation. Details of fuzzy ranking indices have been presented in Remarks 2 of Section 5.2. Let $R(\tilde{PC}_{optimum})$ and $R(\tilde{PC}_i)$ be denoted as the fuzzy ranking indices of the optimum solution and an individual solution of a single run $i$ $(1 \le i \le N)$ of the approach respectively. The average deviation can be calculated as

$$AD = \frac{\sum_{i=1}^{N} | R(\tilde{PC}_i) - R(\tilde{PC}_{optimum}) |}{N} \tag{11.1}$$

The average deviation is an intuitive measure, reflecting how far the average of individual solutions is from the optimum solution in terms of minimising fuzzy project completion times.

(b) Maximum deviation from the optimum solution (MD):

The maximum deviation is the worse solution obtained after completing $N$ runs of an approach, and can be defined as

$$MD = \max_{i \in N} | R(\tilde{PC}_i) - R(\tilde{PC}_{optimum}) | \tag{11.2}$$

The maximum deviation is used to measure how the worse performance can occur at random while implementing the approach.

(c) Standard deviation from the optimum solution (SD):

The standard deviation is the measure of a probability distribution of variance between the optimum solution and the actually best solution obtained by running an approach $N$ times, and can be calculated as

$$SD = \sqrt{\frac{\sum_{i=1}^{N}[R(P\tilde{C}_i) - R(P\tilde{C}_{optimum})]^2}{N-1}} \tag{11.3}$$

The standard deviation is important in statistics, providing a useful basis for interpreting the variability of individuals from solution sets in probability.

## 11.2 Key Parameters in Fuzzy GA Based Approaches

In this section, two key parameters, the number of generations and the tabu size, are examined to determine their appropriate values for the five sample projects generated by the system. If the values of two parameters are too large, enormous computational time is required in implementing fuzzy GA based approaches. In addition, solutions may not necessarily be improved. This section addresses these two parameters in the subsequent two subsections. The experiment is first conducted to evaluate the appropriate number of generations. It then examines how different sizes of tabu affect the implementation of fuzzy GA with tabu, while using appropriate values obtained from the experiment.

### 11.2.1 Appropriate Values for the Generation Parameter

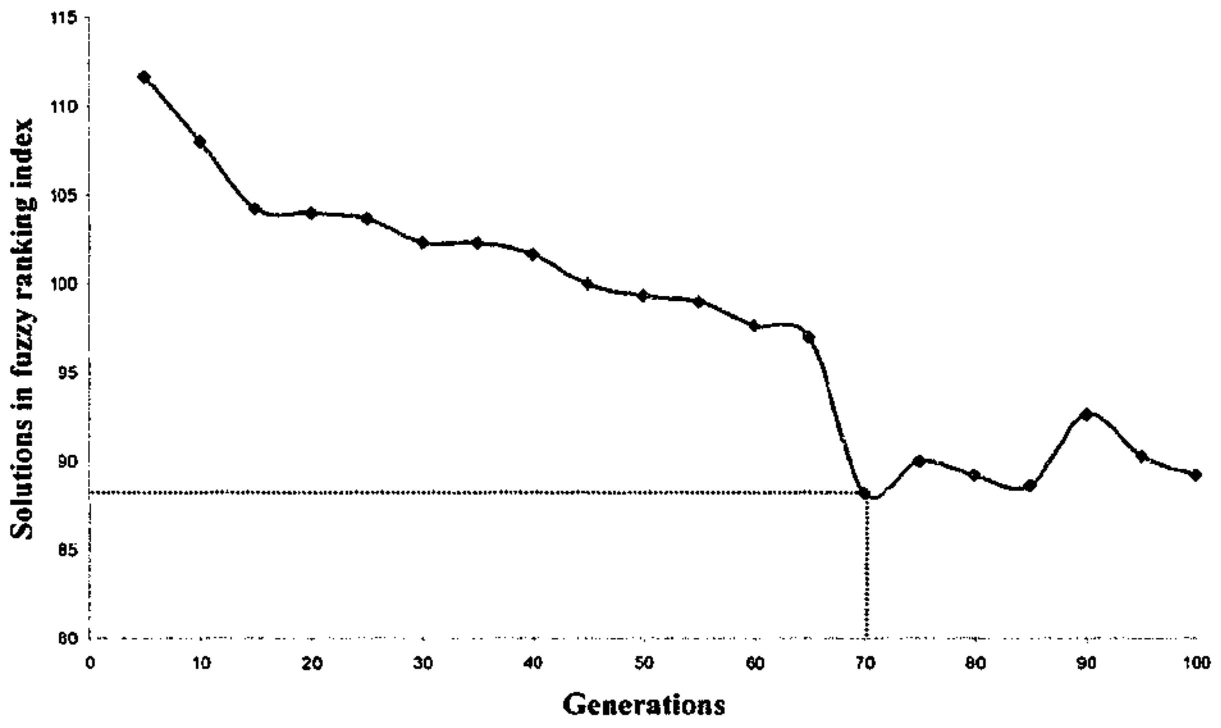
Appropriate values for different project instances are difficult to set directly because of the complexity of the projects itself, including such issues as the project size, the number of modes and resources involved. The value of the generation parameter is effectively a stopping criterion to determine when a fuzzy GA based approach is terminated. A small set value leads poor results in project scheduling while implementing the approach. Improperly large values may not improve the quality of performance and may unnecessarily increase the processing time.

In the experiments conducted, sizes of project instances are set at 25, 50, 100, 150 and 200 activities respectively where all three modes and four kinds of resources are involved. The experimental results of these project instances will be explained individually. To get the best result in project scheduling, it is important to conduct experiments to determine the best value of the generation parameter for different sizes of project instances, prior to implementing a fuzzy GA based approach in the system. This is because it may be difficult to give a direct clue to an appropriate value setting for individual instances in general.

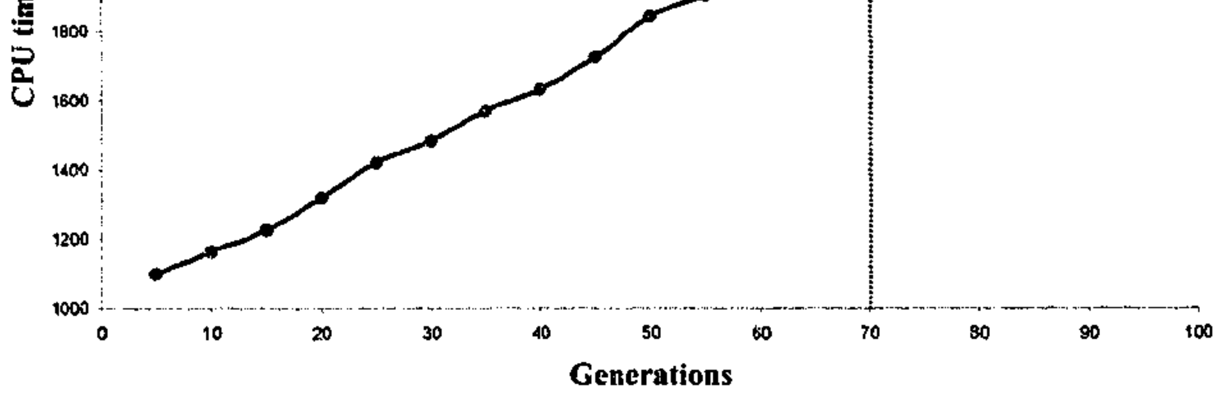
For the project with 25 activities, the population size is set to 70 and the fit chromosomes parameter for each generation is set at 20. Details of the population size setting and the proper number of fit chromosomes to be selected, have been presented in Section 8.8.1. If the population size is set at a slightly different level, even under the same instance, it will not influence the performance of the approach greatly because the value of the generation parameter can be re-estimated experimentally once the population size has been determined. However, based on the large number of experiments carried out in my PhD studies, the population size of this instance can be properly chosen as 70, and the appropriate number of fit chromosomes is set at 20.

The experimental result of this instance is given in Figure 11.1(a). When the number of generations is set to less than 65, the approach has a poor performance for this instance. A better performance result is achieved, when the number of generations is set within the range of 70 to 80. The processing time for the system to implement the approach in this instance requires only between 2.1 and 2.3 seconds as given in Figure 11.1(b).

For the project instance with 50 activities, the population size is set to 70 with 20 fit chromosomes being selected in each generation. As clearly shown in Figure 11.2(a), the performance is very poor with the fuzzy GA approach if the number of generations is smaller than 90. The appropriate value of the generation parameter for the approach to achieve a better performance should range from 100 to 120, for which the processing time required ranges from 5.2 to 5.8 seconds as shown in Figure 11.2(b). In the later experiments for determining both the appropriate size of tabu and the comparative analysis for the four metaheuristic approaches, the value of the generation parameter will be set as 110 for the project size of 50 activities.

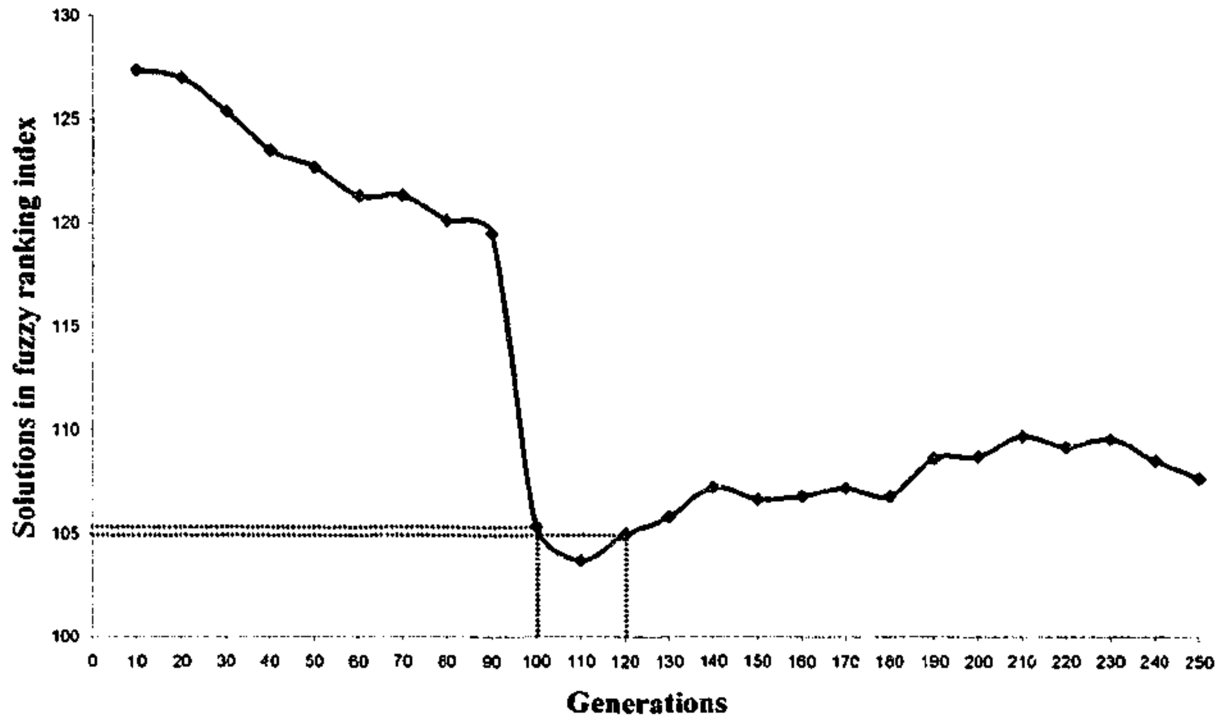


(a) The number of generations and its corresponding solutions

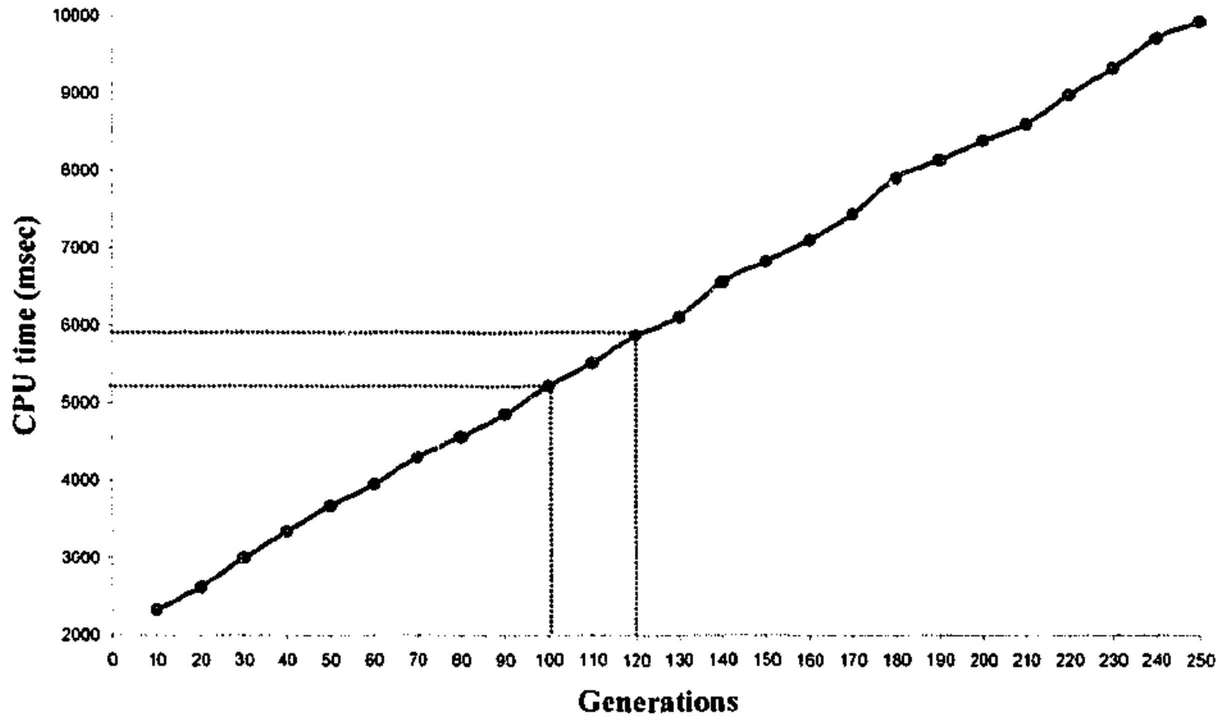


(b) The number of generations with their corresponding CPU time required

Figure 11.1 Experiment with the generation parameter for a project of 25 activities



(a) The number of generations and its corresponding solutions



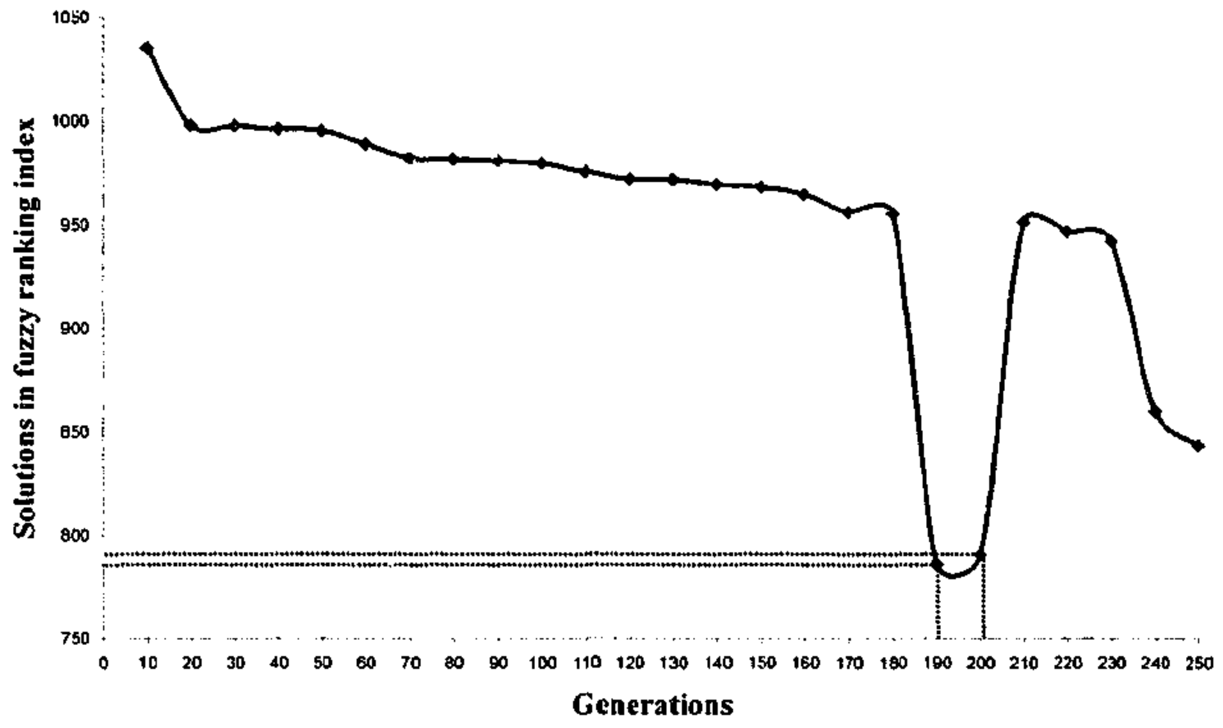
(b) The number of generations with their corresponding CPU time required

Figure 11.2 Experiment with the generation parameter for a project of 50 activities

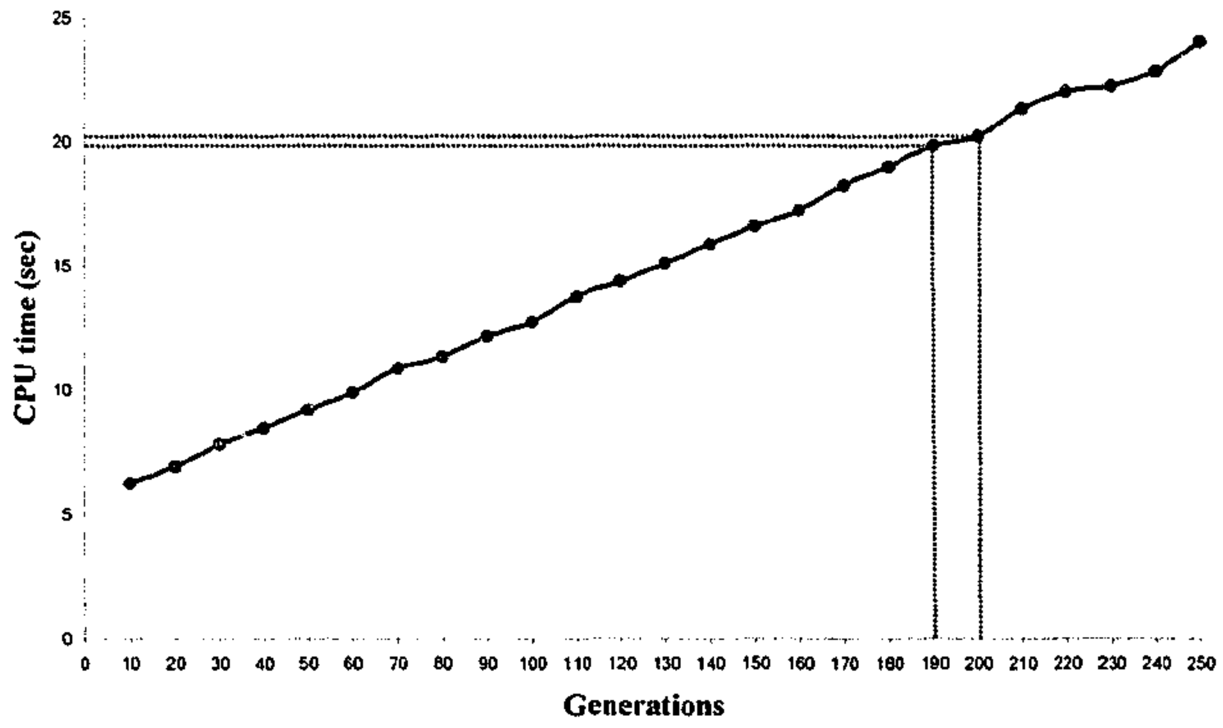
For the project size of 100 activities, the population size is set to 80 and the fit chromosomes being selected from an entire population are set to 20. Figure 11.3 shows the experimental results of the solution quality when changing values of the generation parameter using the fuzzy GA approach. As shown in Figure 11.3(a), the solutions obtained are poor in terms of minimising the fuzzy project completion time when the number of generations is set to less than 190. When the number of generations reaches to the range of 190 and 200, two lowest points in solution are gained whereas the solutions get worse or are not much improved when the number of generations is further increased, as clearly shown in Figure 11.3(a). Therefore, the appropriate value of the generation parameter is between 190 and 200, and the time required for implementing the approach is about 20 seconds as shown in Figure 11.3(b).

Figure 11.4 shows the experimental results for both the solutions and the CPU time using the fuzzy GA approach under different numbers of generations, where the population size is 150 with 25 fit chromosomes being selected from each generation. Clearly, the quality of the best solutions, obtained by implementing the fuzzy GA approach, is very poor when the number of generations is smaller than 160. As shown in Figure 11.4(a), the appropriate value for the generation parameter ranges from 320 to 350 for solving the project sizes of 150 activities with the three executive modes and four different kinds of resources. The processing time for implementing the approach for such project instances is approximately 45 to 48 seconds as shown in Figure 11.4(b).

Figure 11.5 shows the performance of the fuzzy GA approach under different value settings of the generation parameter, with the project size of 200 activities, in terms of the quality solutions and the corresponding processing times of implementing the fuzzy GA, where the population size is set to 150, with 30 fit chromosomes being selected from each generation. As shown in Figure 11.5(a), the quality of solutions obtained is very poor if the fuzzy GA approach is implemented with the number of generations being less than 170. The appropriate values of the generation parameter for the optimal performance range from 390 to 420, where the computational time required for implementing the approach is between 84 and 98 seconds as shown in Figure 11.5(b). The value of the generation parameter will be chosen as 420 for evaluating both the tabu size and fuzzy GA based approaches with project sizes of 200 activities.

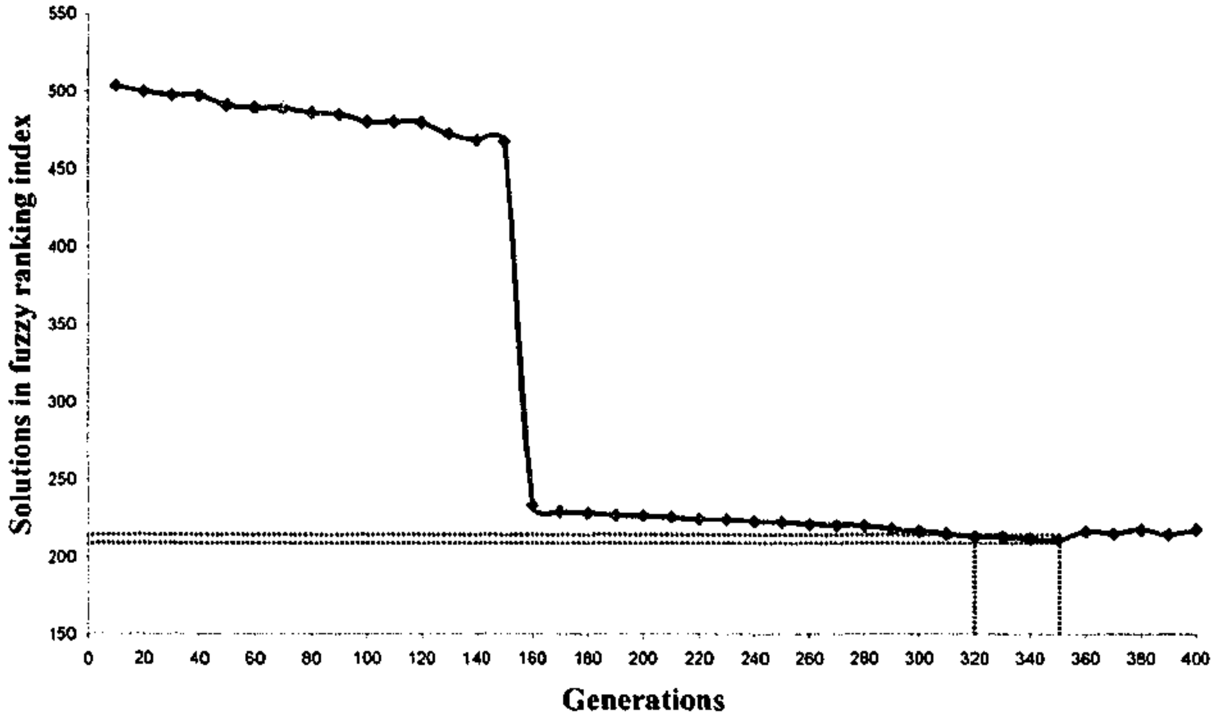


a) The number of generations and its corresponding solutions

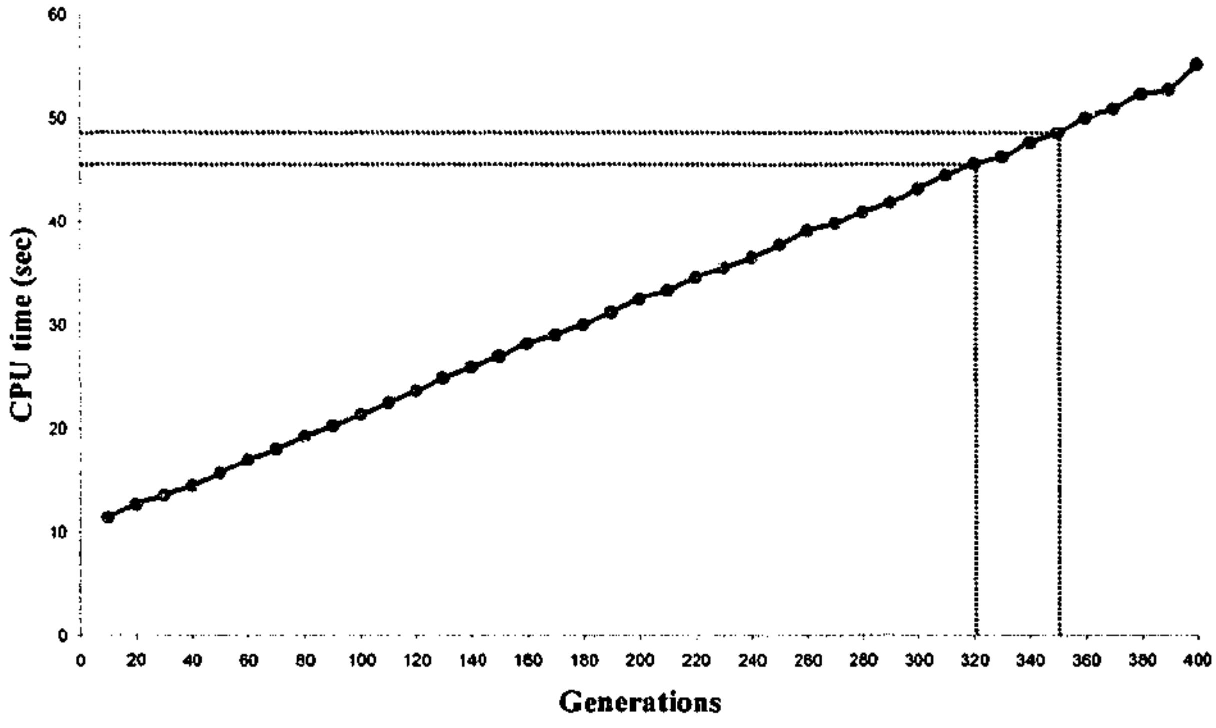


b) The number of generations with their corresponding CPU time required

Figure 11.3 Experiment with the generation parameter for a project of 100 activities

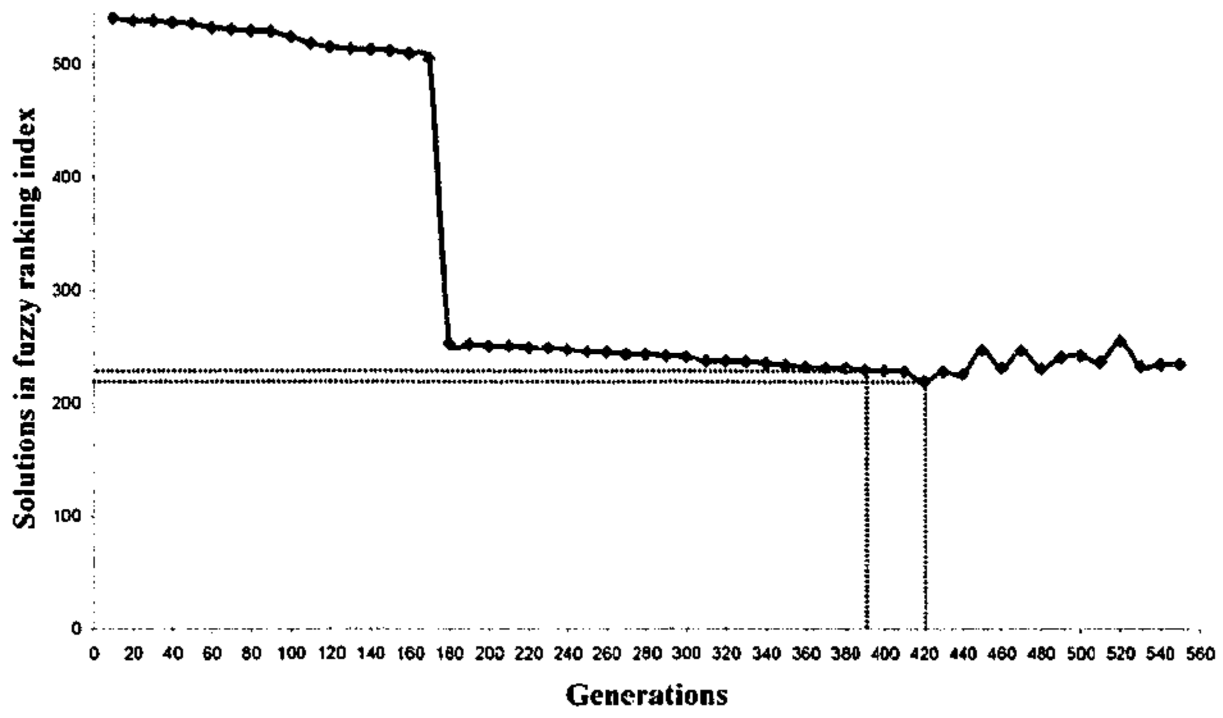


a) The number of generations and its corresponding solutions

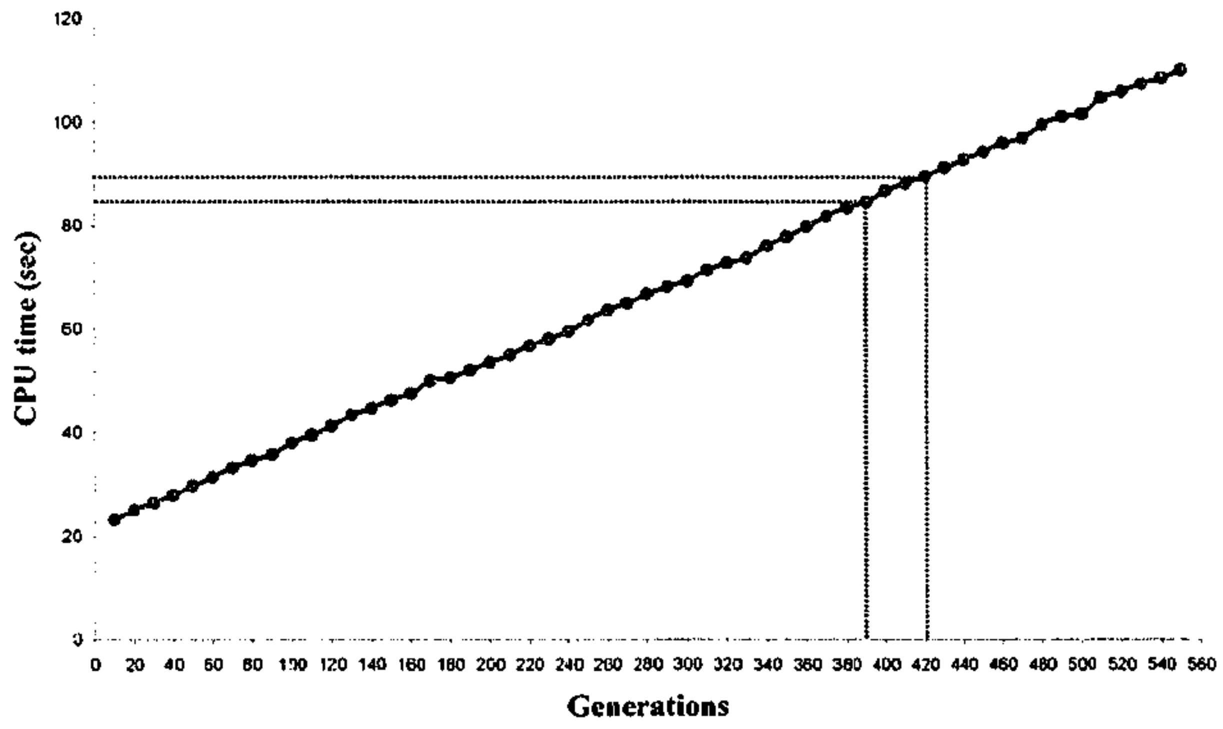


b) The number of generations with their corresponding CPU time required

Figure 11.4 Experiment with the generation parameter for a project of 150 activities



a) The number of generations and its corresponding solutions



b) The number of generations with their corresponding CPU time required

Figure 11.5 Experiment with the generation parameter for a project of 200 activities

As shown in Figures 11.1-11.5, the solution curves of these five individual projects are not all similar. This is because these project instances have different network structures. It is noticed that some solutions (points on the curves) within certain generation value range, drop dramatically while evaluating the generation parameter in Figures 11.1-11.3. However, the trends of the solutions in these five instances are all the same. As such solutions are gradually improved at a changeable gradient when the values of the generation parameter increase from the beginning to a certain value. Despite individual differences in the curves, each curve has a common feature that a special turning point or range can be detected at which the best global solutions are to be found.

These experiments also demonstrate that the computational time required is no more than one and a half minutes, for up to 200 activities of any project instance with three executive modes and four types of resources when the fuzzy GA approach is applied. This indicates that the fuzzy GA approach is computationally efficient for solving project scheduling problems.

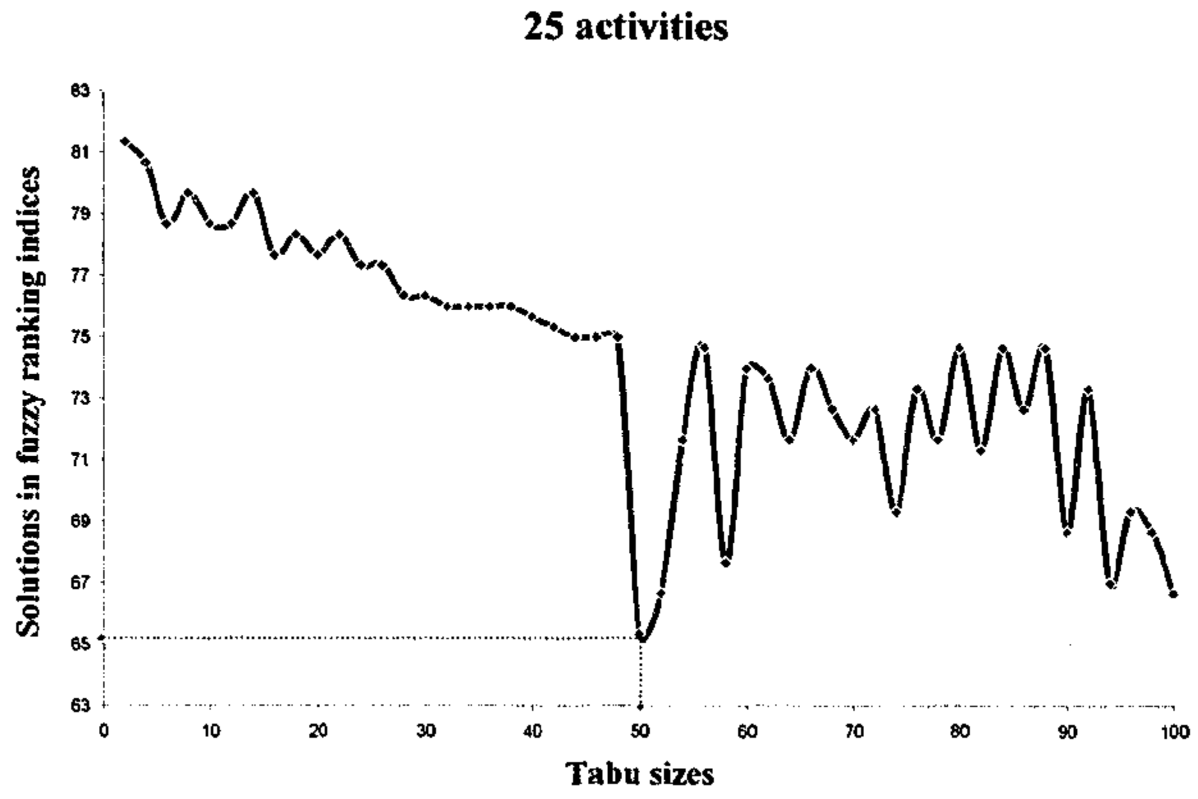
The appropriate values of the generation parameter obtained by experiments for different sizes of projects will be used to evaluate the proper tabu sizes and to conduct a comparative analysis for the four metaheuristic approaches developed in Subsection 11.2.2 and Section 11.4.

### 11.2.2 Appropriate Tabu Sizes

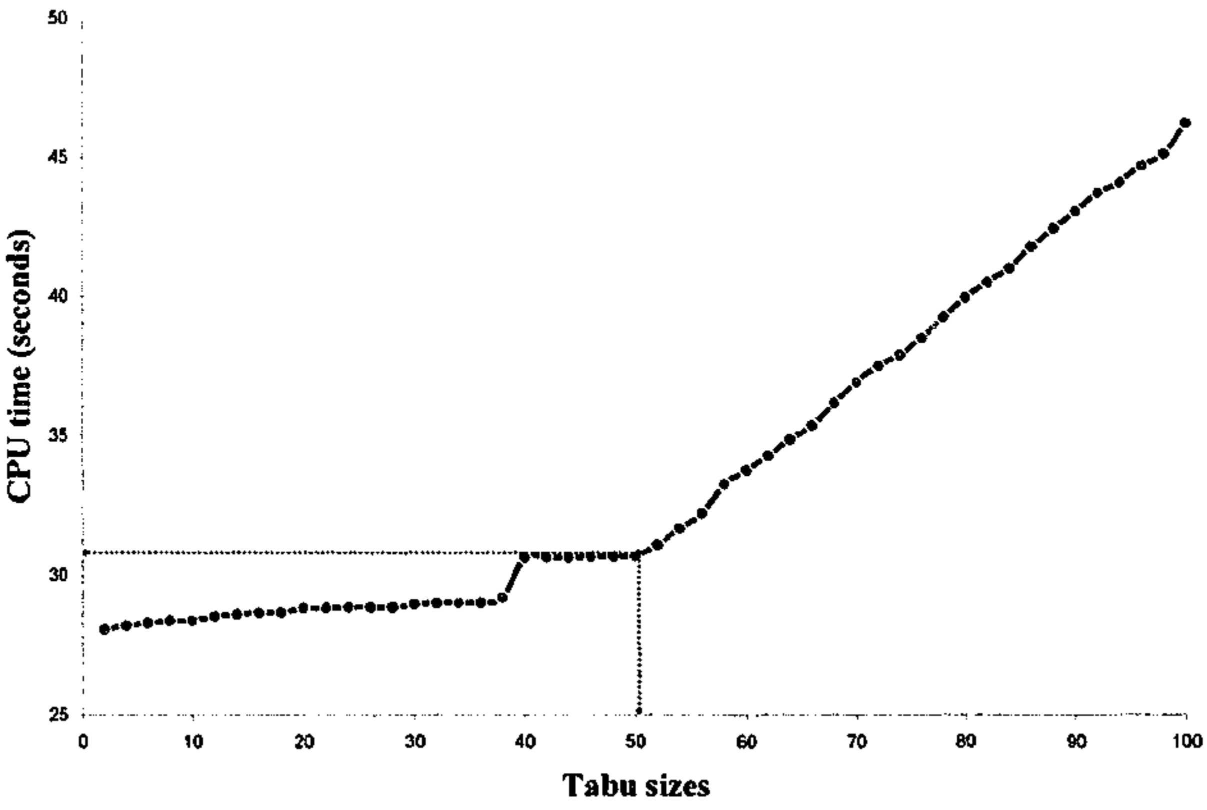
The tabu mechanism is an important component in the fuzzy GA with tabu approach developed. This approach can avoid producing chromosomes that have been generated recently. This phenomenon may occur in fuzzy GA. Therefore, the incorporation of the tabu mechanism in fuzzy GA is an effective way of bringing out more various chromosomes than by the use of fuzzy GA alone, improving the search process for effectively finding an approximate globally optimal solution in terms of minimising the fuzzy project completion time. However, the size selected for the tabu mechanism affects the computational time required for implementing the approach. In addition, an improperly large size of tabu may not get better solutions. The aim of the experiments conducted is to examine which sizes of tabu are appropriate to the five individual project sizes when the fuzzy GA with tabu approach is applied.

Figure 11.6 shows both the solutions in terms of minimising the fuzzy project completion time and the corresponding CPU time for experiments conducted with different sizes of the tabu mechanism when the fuzzy GA with tabu approach is applied to a project instance with 25 activities, where the generation parameter is set to 70, a value obtained from the experiment reported in Subsection 11.2.1 for the best solution quality. The other parameter settings are the same as used in the experiment reported of Subsection 11.2.1. As shown in Figure 11.6(a), the appropriate size of the tabu mechanism for obtaining the best solution is 50 in terms of minimising the fuzzy project completion time. The solutions are not much improved when the tabu size is set to a value greater than 50. Figure 11.6(b) indicates that the tabu size of 50 is a critical point. That is, the computational time is significantly increased when the tabu size is over 50. Therefore, the appropriate size of tabu for projects with 25 activities is around 50. At this tabu size, the computational time required is about 30 seconds.

For a project size with 50 activities, the generation parameter is set to 110. This value was obtained for the best performance of the fuzzy GA approach with this size of project in the experiment of Subsection 11.2.1. The values of the other parameters are set to the values reported in Subsection 11.2.1. Figure 11.7 shows the relationship between the solution quality and the CPU time when the tabu size changes. Clearly, the tabu size of 50 is the lowest point in the curve of solution changes as shown in Figure 11.7(a). However, when the tabu size is further increased, solutions are not further improved, and the CPU time required is significantly increased, as shown in Figure 11.7(b). The computational time is about 97 seconds when the fuzzy GA with tabu approach is applied to a project size of 50 activities.

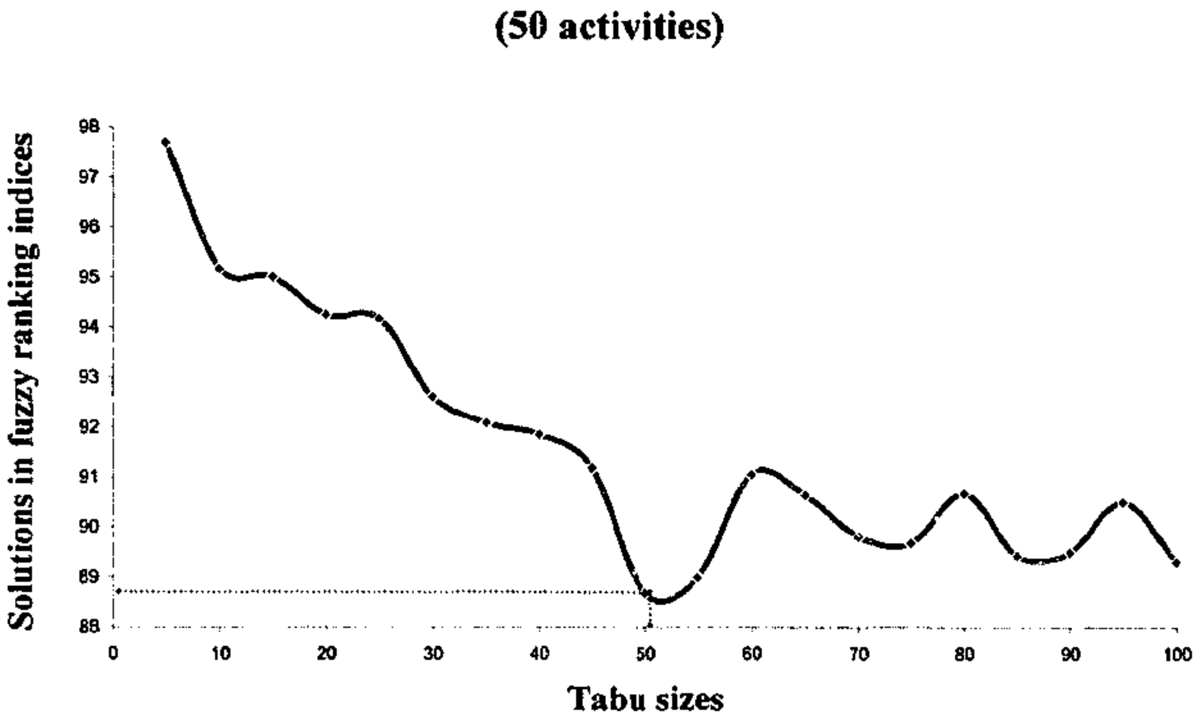


(a) Solution changes with various tabu sizes

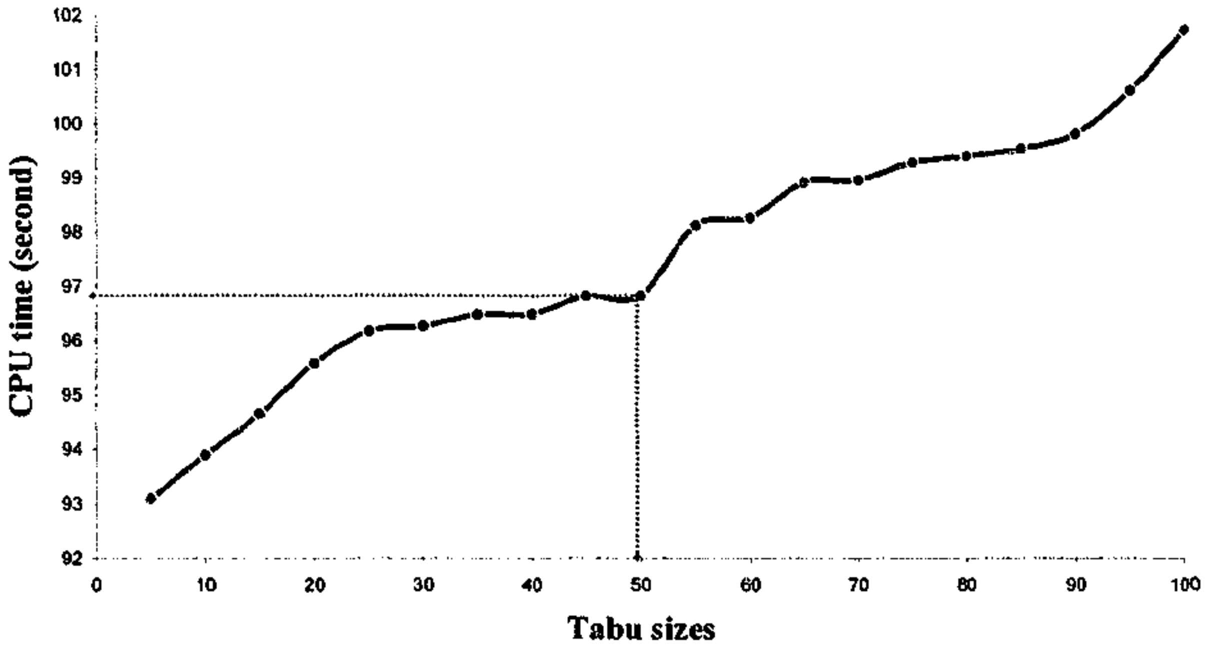


(b) CPU time required for various tabu sizes

Figure 11.6 The evaluation of tabu sizes in a 25-activity project



(a) Solutions with various tabu sizes



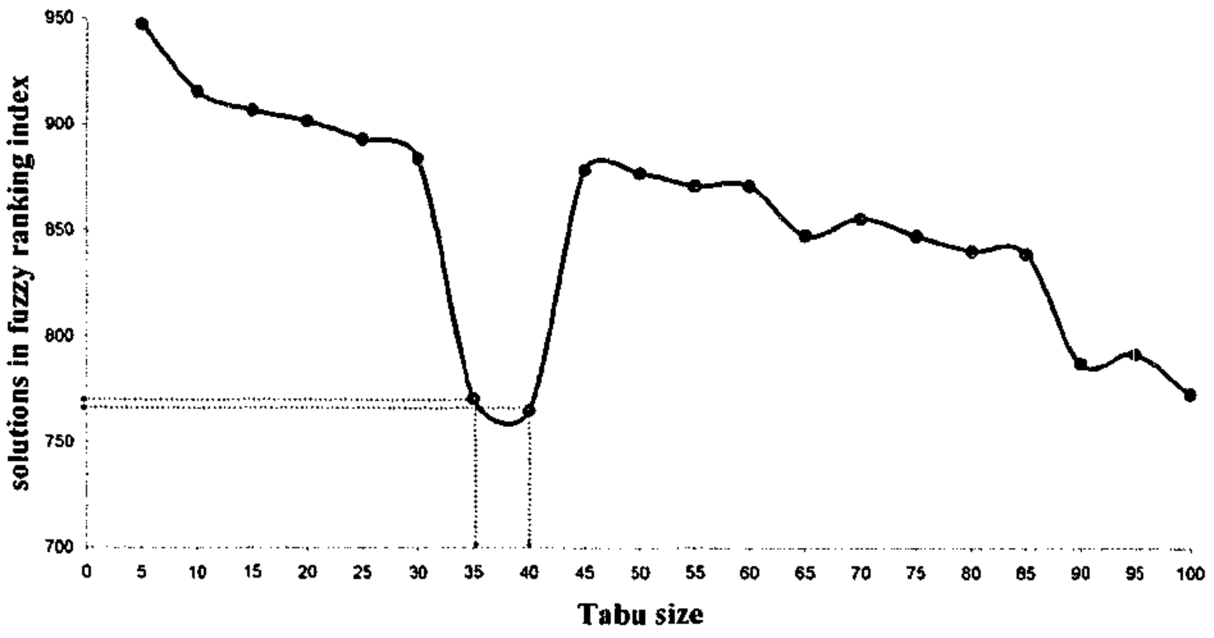
(b) The relations between the CPU time and tabu sizes

Figure 11.7 The experiment to evaluate tabu sizes in a project with 50 activities

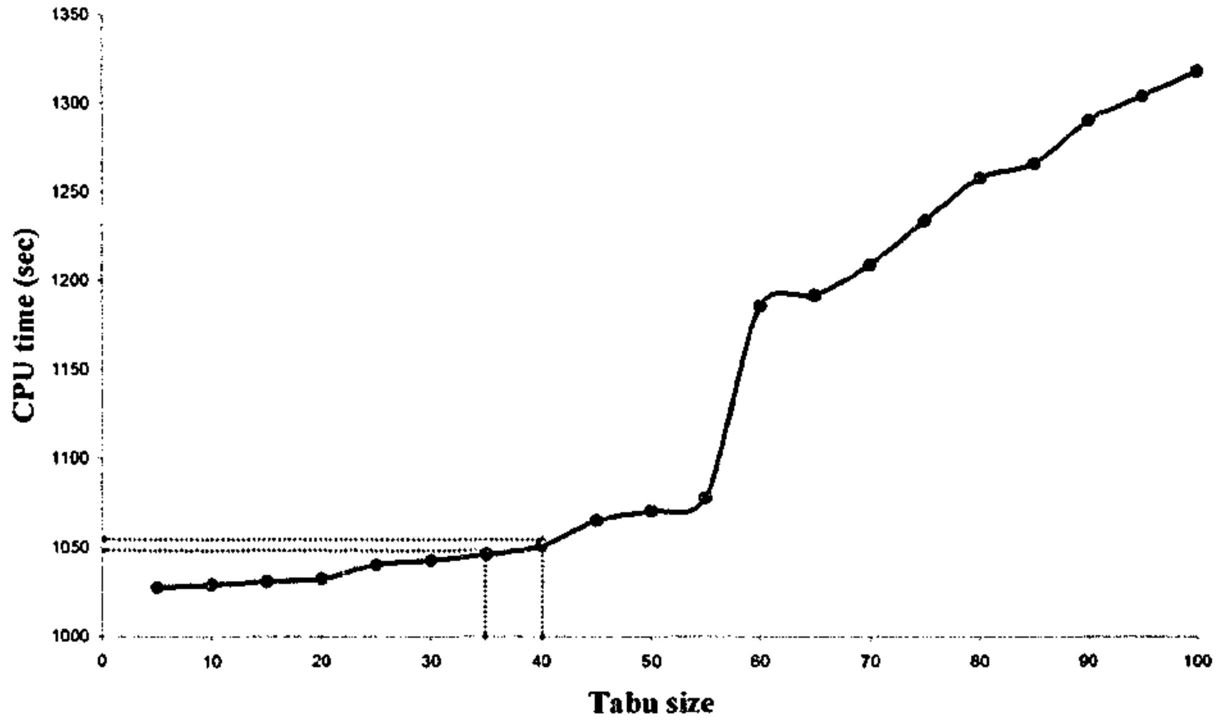
Figure 11.8 shows the solutions and their corresponding CPU time required when the tabu size changes in the experiment with determination of the appropriate tabu size for a project of 100 activities. As shown in Figure 11.8, the range between 35 and 40 for the tabu size are the critical area where the good solutions are likely to be obtained. The computational time required for the size in this range is about 1059 seconds. However, the computational time requires significantly when the tabu size is over 60 as shown in Figure 11.8(b), and the solution is not improved dramatically. Clearly, the appropriate tabu size is between 35 and 40 where the solution quality is in the better performance.

Figure 11.9 shows the experimental results in evaluating the quality of performance of the fuzzy GA with tabu approach with different sizes of tabu, where the project comprises 150 activities. Other parameter settings in this experiment, are the same as those used in evaluating fuzzy GA in Subsection 11.2.1. The number of generations is set at 350, which produced the best solution for the fuzzy GA approach as reported in Subsection 11.2.1. The tabu size of 25 is the appropriate size for obtaining the best solution in terms of minimising the fuzzy project completion time as shown in Figure 11.9(a), whereas the solution is not dramatically improved with a larger tabu size. In addition, the corresponding CPU time required for that size involved is about 34 minutes. Obviously, the processing time required in fuzzy GA with tabu is much more than by using fuzzy GA alone that only requires 48 seconds for the best solution reported in the experiments of Subsection 11.2.1. Figure 11.9(a) clearly shows that the solutions obtained for any tabu size are significantly better than the best solution obtained by the fuzzy GA alone (Figure 11.4(a)).

**(100 activities)**



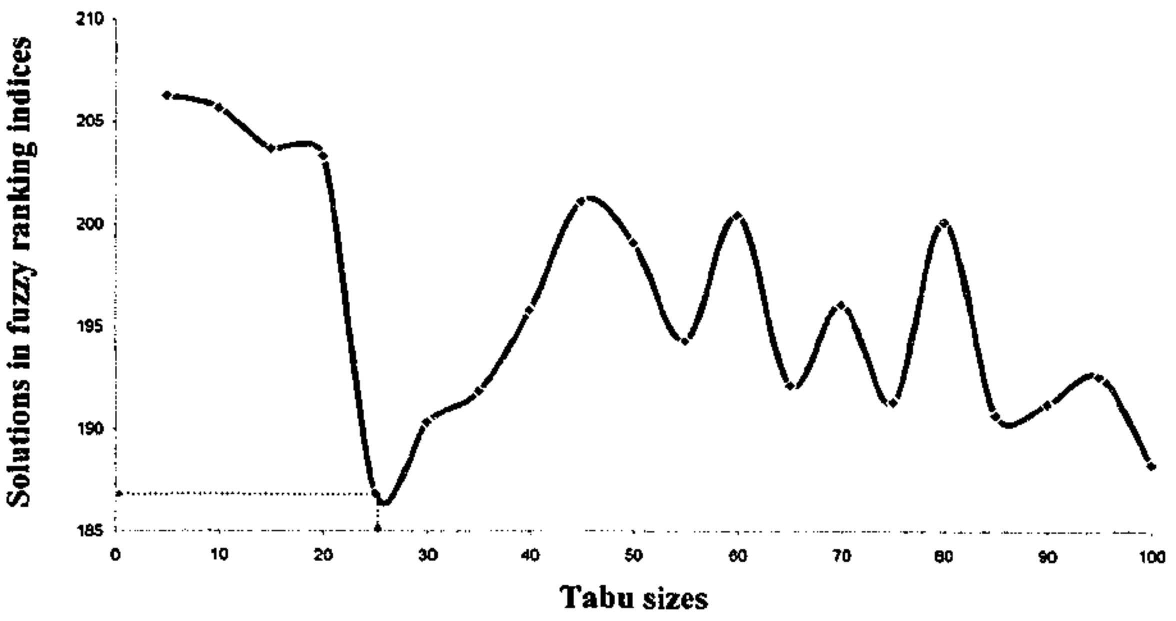
(a) Solutions changes with various tabu sizes



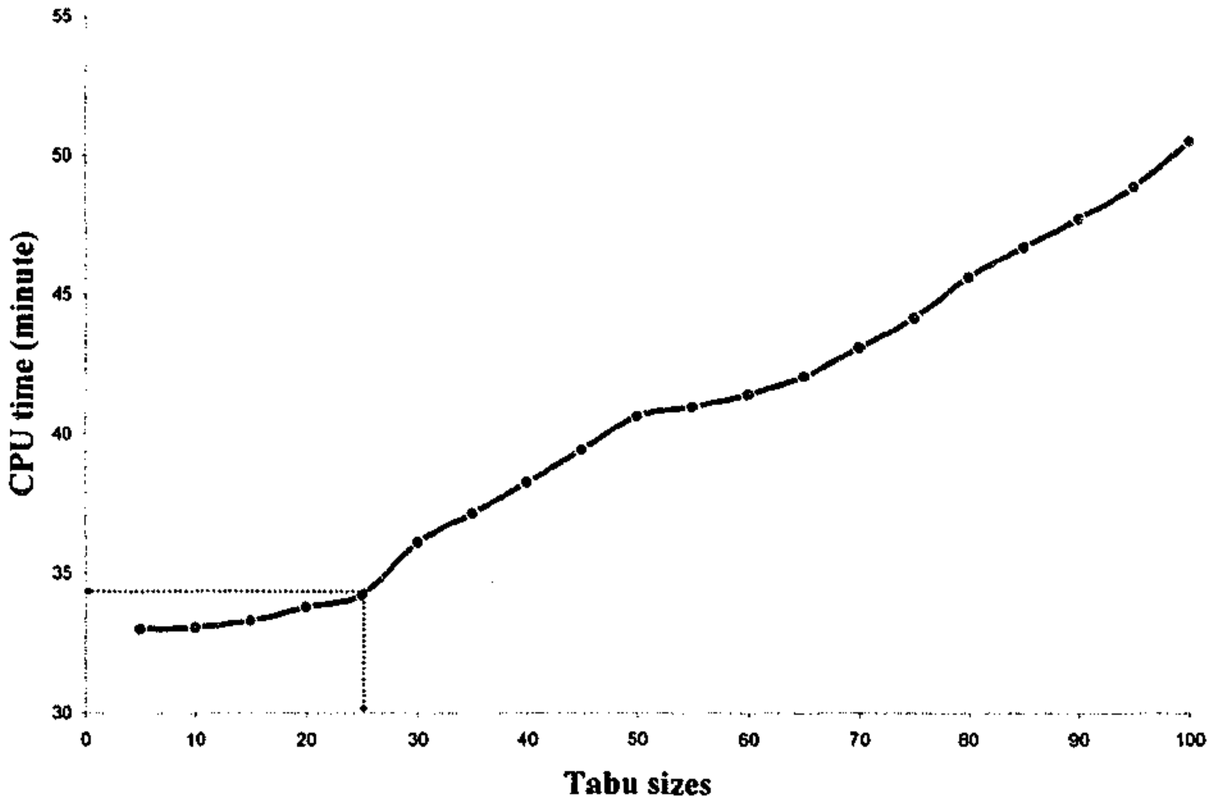
(b) CPU time required for various tabu sizes

Figure 11.8 The evaluation of tabu sizes in a 100-activity project

**(150 activities)**



(a) Solutions changes with various tabu sizes



(b) CPU time required for various tabu sizes

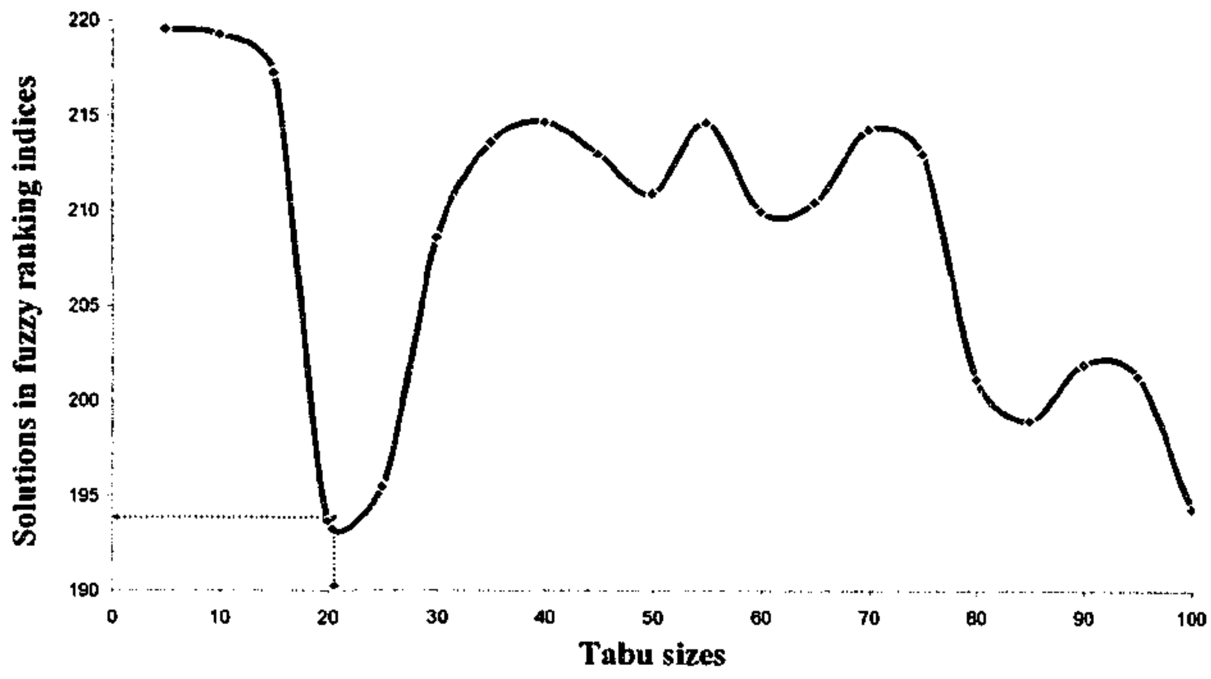
Figure 11.9 The evaluation of tabu sizes in a 150-activity project

For a project consisting of 200 activities, the generation parameter is set to 420. This value was providing the best solution in a previous experiment for this project size in Subsection 11.2.1. The other parameter values chosen are the same as those used in the previous experiment in Subsection 11.2.1. Figure 11.10 shows the solutions and their corresponding CPU times required for different tabu sizes when the fuzzy GA with tabu approach is applied. As shown in Figure 11.10(a), the solution point at a tabu size of 20 is the lowest one. The computational time for that tabu size requires about 45 minutes for solving projects of 200 activities, three executive modes, and four types of resources. When the tabu size is further increased, the quality of solutions is not improved dramatically whilst the processing time required is significantly increased, as shown in Figure 11.10(b). If fuzzy GA is applied to this project size, the computational time required for the best solution is only about 98 seconds. However, the solutions obtained by fuzzy GA are worse than the solutions obtained by the fuzzy GA with tabu approach, as can be seen by comparing between Figures 11.5 and 11.10.

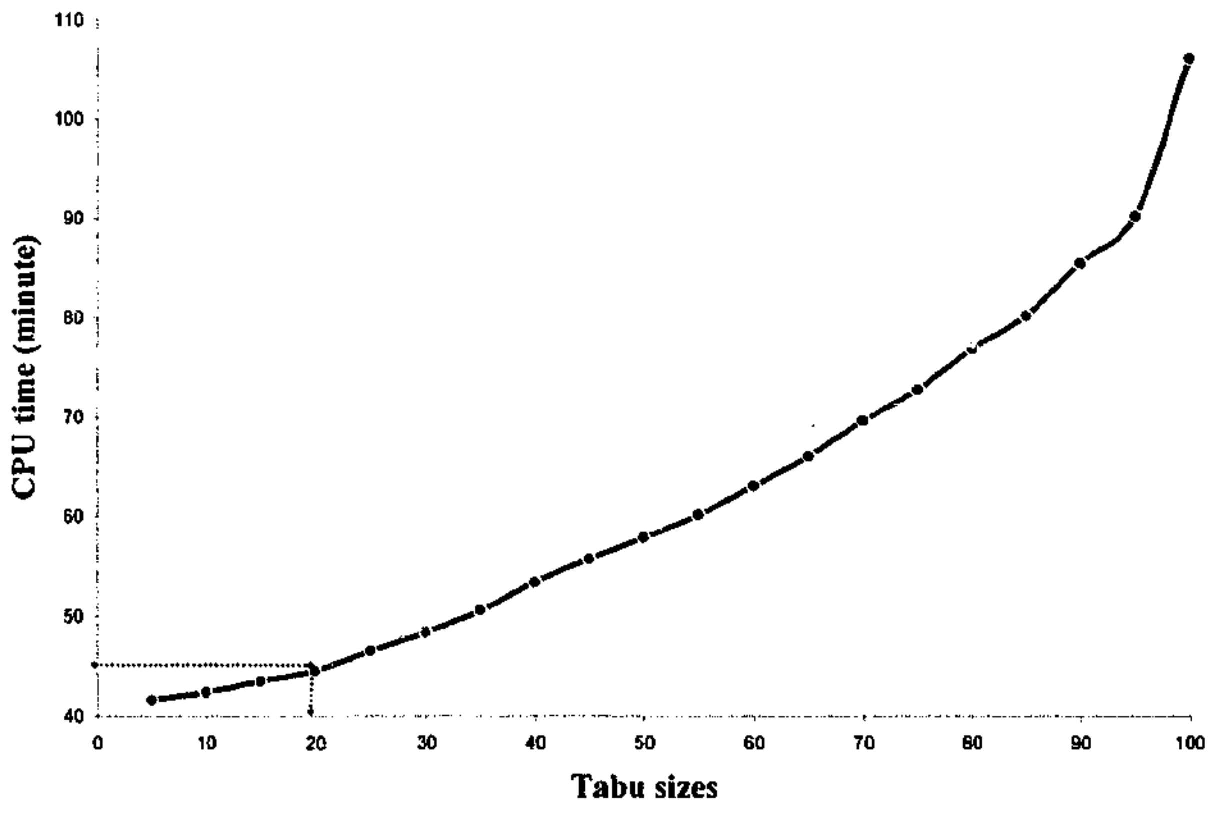
The experiments conducted above show that, although the solutions fluctuate in most of five project instances as the tabu size changes, the best solution(s) can be found when a particular value or the value range of the tabu sizes, shown in Figures 11.6-11.10, are applied. In addition, when the tabu size is beyond this particular value or the value range, solutions may not be further improved, and the computational time increases significantly.

The experiments also indicate that the solutions obtained from the fuzzy GA with tabu approach are much better than the solutions obtained by the fuzzy GA alone. However, the computational time required ranges from 4 to 42 times more than fuzzy GA alone. The amount of computational time required depends on the size of a project and the number of modes and resources involved. A further analysis on performance behaviours of the approaches in terms of the randomised nature, will be reported in Section 11.4.

**(200 activities)**



(a) Solutions changes with various tabu sizes



(b) CPU time required for various tabu sizes

Figure 11.10 The evaluation of tabu sizes in a 200-activity project

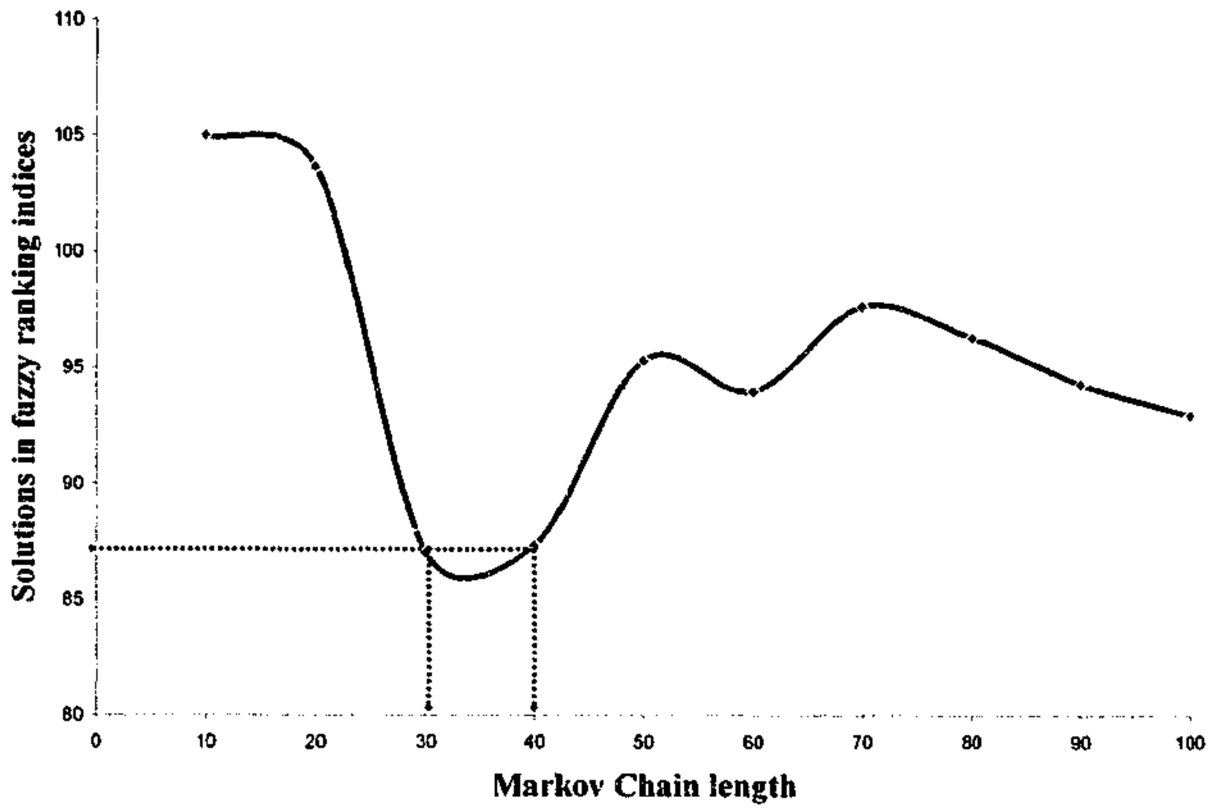
## 11.3 Key Parameters in Fuzzy SA Based Approaches

In this section, the experiments for determining key parameters for fuzzy SA based approaches will be conducted using the same five different sizes of projects, as used in the experiments for fuzzy GA based approaches. The experiments will be conducted to: (a) evaluate appropriate Markov Chain lengths for the different project sizes, using the cooling ratio of 0.8 that is suggested as the appropriate cooling ratio to many applications (Pham and Karaboga 2000); (b) examine how different values of the cooling ratio will affect the solution quality in the annealing process when the length of Markov Chain is a constant set to the value obtained as the appropriate length for an individual project in the experiment; and (c) determine the appropriate sizes of the tabu mechanism for the individual projects using the fuzzy SA with tabu approach.

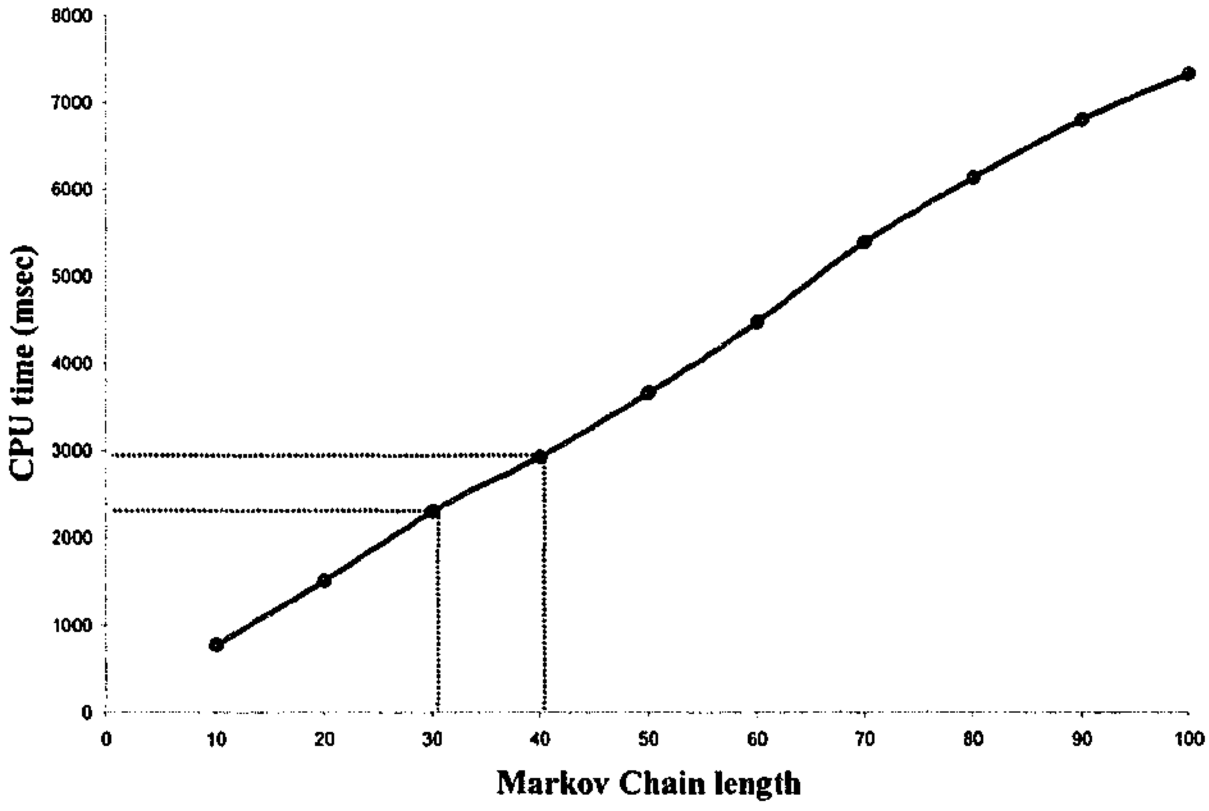
### 11.3.1 Appropriate Markov Chain Length

Markov Chain is one important parameter in SA to give the proper transition probability for the random walk in searching neighbourhood solutions at a certain temperature level before going down to the next lowest temperature level. To search robustly in each temperature level, the length of the Markov Chain in fuzzy SA based approaches I have developed, is set to be changed, depending on the search status at the previous temperature level. Details about the Markov Chain length have been presented in Section 9.6. The experiments presented in this subsection are used to determine the initial appropriate lengths of the Markov Chain for these five projects where the cooling ratio is set to 0.8 and the initial temperature is set to be the number of activities of a project multiplied by factorial of the maximum of modes. This has been found to be an appropriate initial temperature through experiments. Details of the initial temperature setting are presented in Section 9.6.

Figure 11.11 shows solution quality and the corresponding CPU time when Markov Chain length changes for a project with 25 activities. The experiment shows that the appropriate length of Markov Chain is between 30 and 40 where solution in fuzzy ranking index is about 87. This length produces better solution quality where the computational time ranges from 2300 to 2900 milliseconds as shown in Figure 11.11(b).



(a) Solutions with different Markov Chain length



(b) CPU time required for different Markov Chain length

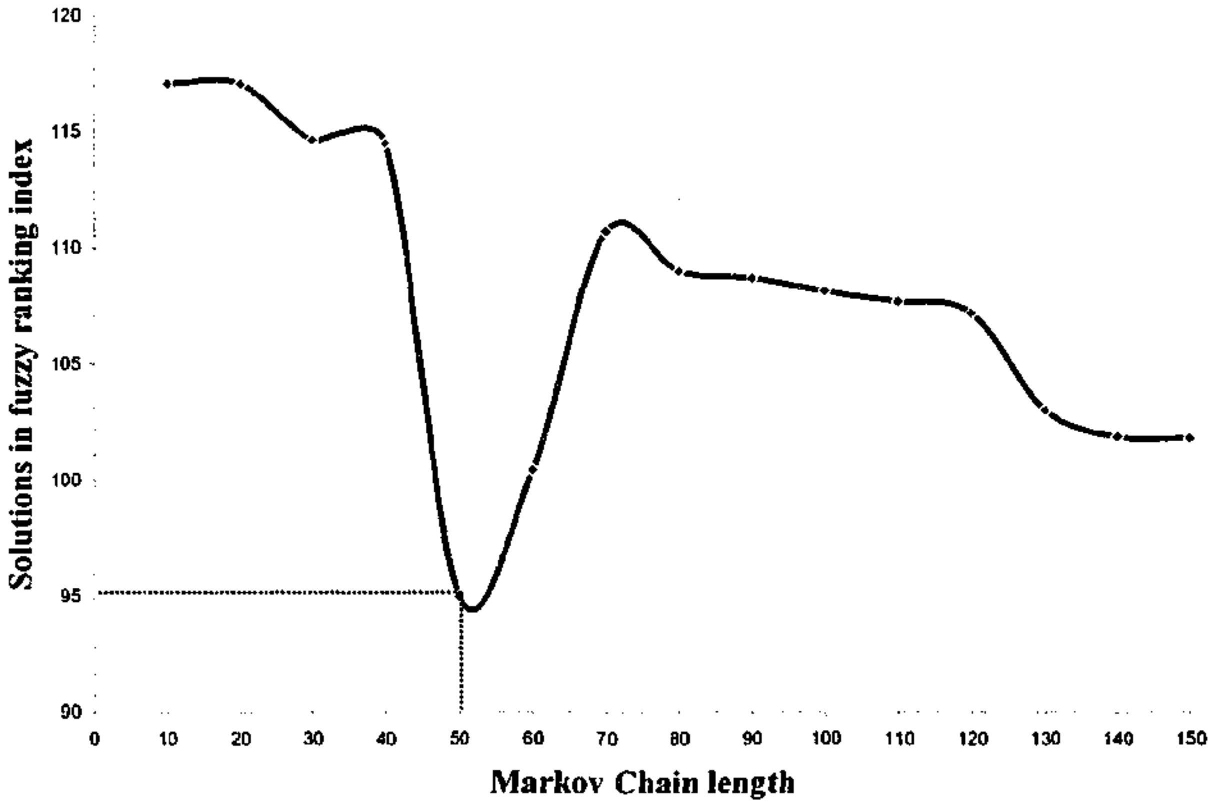
Figure 11.11 The evaluation of Markov Chain lengths in a 25-activity project

Figure 11.12 shows the solution status and the CPU time required with its corresponding Markov Chain length in the experiment for a project with 50 activities. The Figure 11.12(a) clearly shows that the appropriate Markov Chain length is about 50 where the solution in the fuzzy ranking index is at the lowest point of 95. With this length, CPU time requirement is 8343 milliseconds. When Markov Chain length further increases, the solution quality is not really improved and the corresponding computational time is increased significantly.

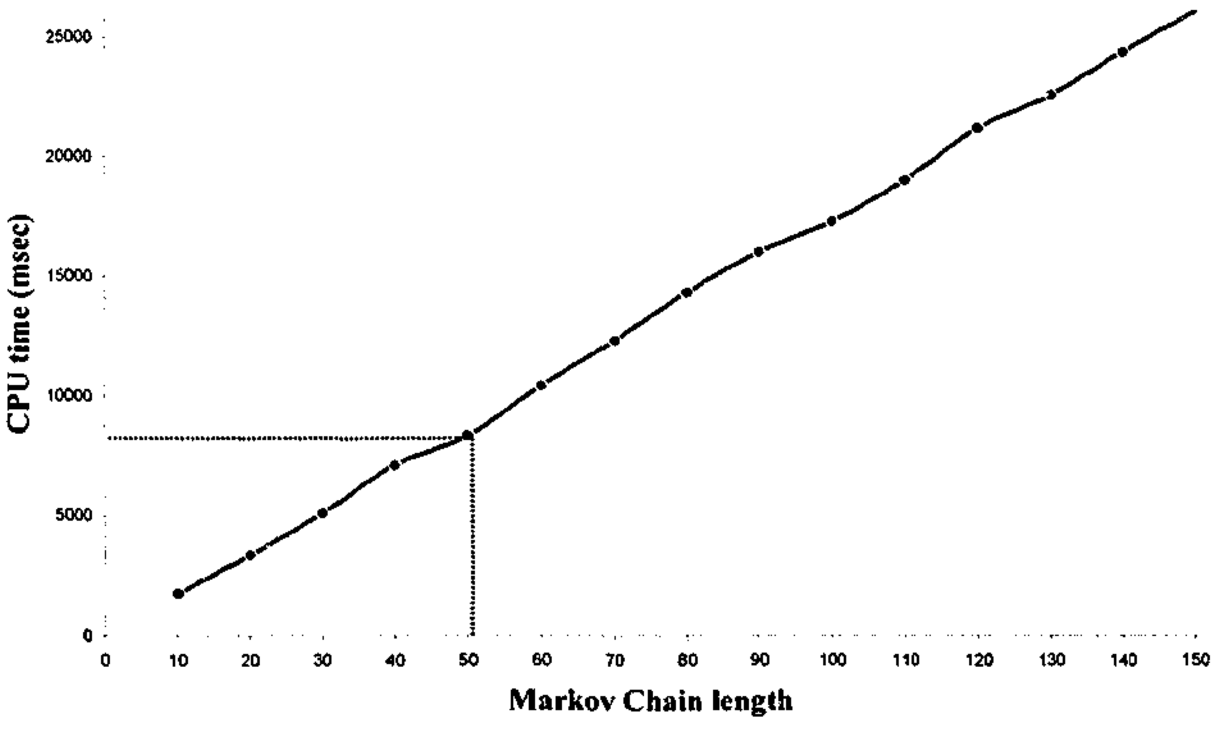
Figure 11.13 shows the experiment for evaluating an appropriate Markov Chain length in a project with 100 activities. For such a size of project, the appropriate Markov Chain length ranges from 100 to 110 where the fuzzy ranking index of the solution is attained at the lowest point. The corresponding CPU time required is about 40 to 46 seconds. If Markov Chain length continues increasing, the solution is not significantly improved. The computational time increases greatly as shown in Figure 11.13 (b).

For a project with 150 activities, the solution quality is very poor when the fuzzy SA approach is implemented with a Markov Chain length less than 30 as shown in Figure 11.14(a). The solution quality is stable when Markov Chain length is greater then 30. However, Markov Chain length is in the range of 150 and 160 with the best performance in terms of obtaining better solutions. Their corresponding CPU time required is between 116 and 127 seconds as indicated in Figure 11.14(b).

Figure 11.15 shows the solution quality and the computational time when Markov Chain length changes in the experiment conducted for a project of 200 activities. Figure 11.15(a) shows that the solutions are stable in general when Markov Chain length is greater then 10. However, the better solutions obtained are in Markov Chain length between 200 and 210. The CPU time requires between 400 and 410 seconds as shown in Figure 11.15(b).

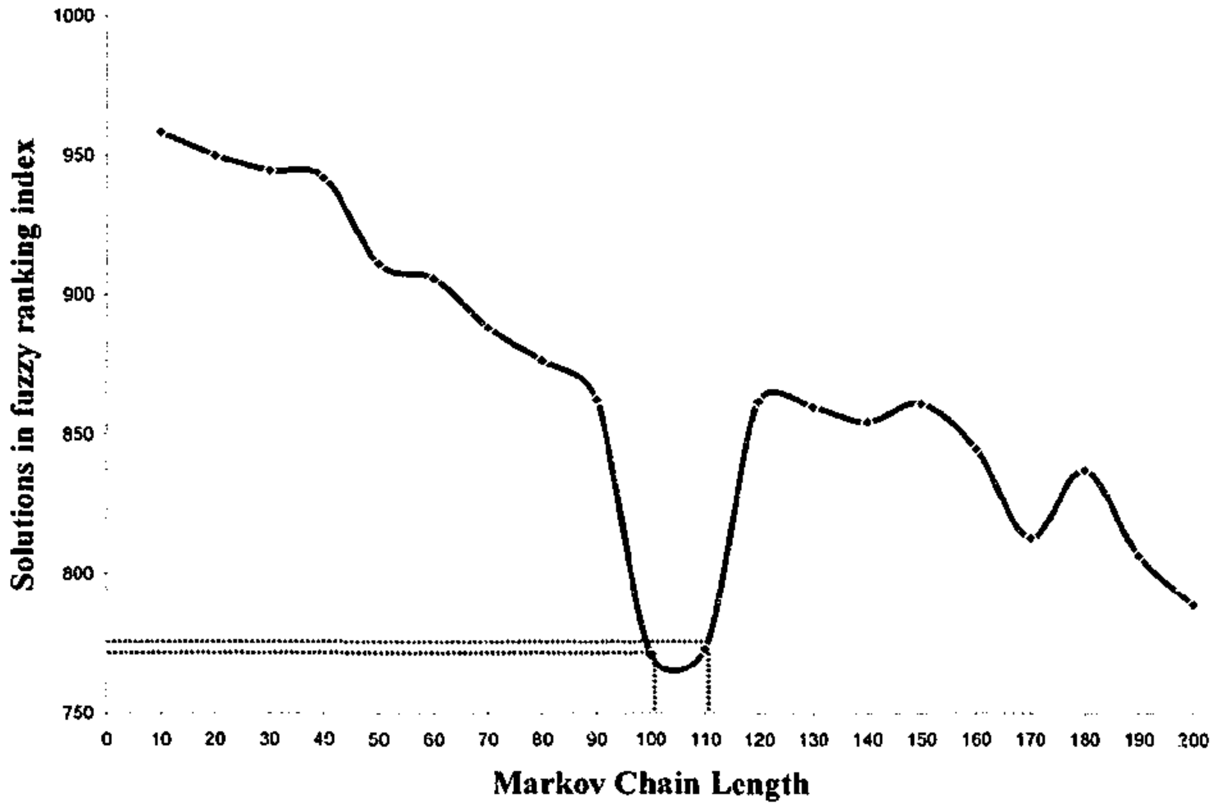


(a) Solutions with different Markov Chain length

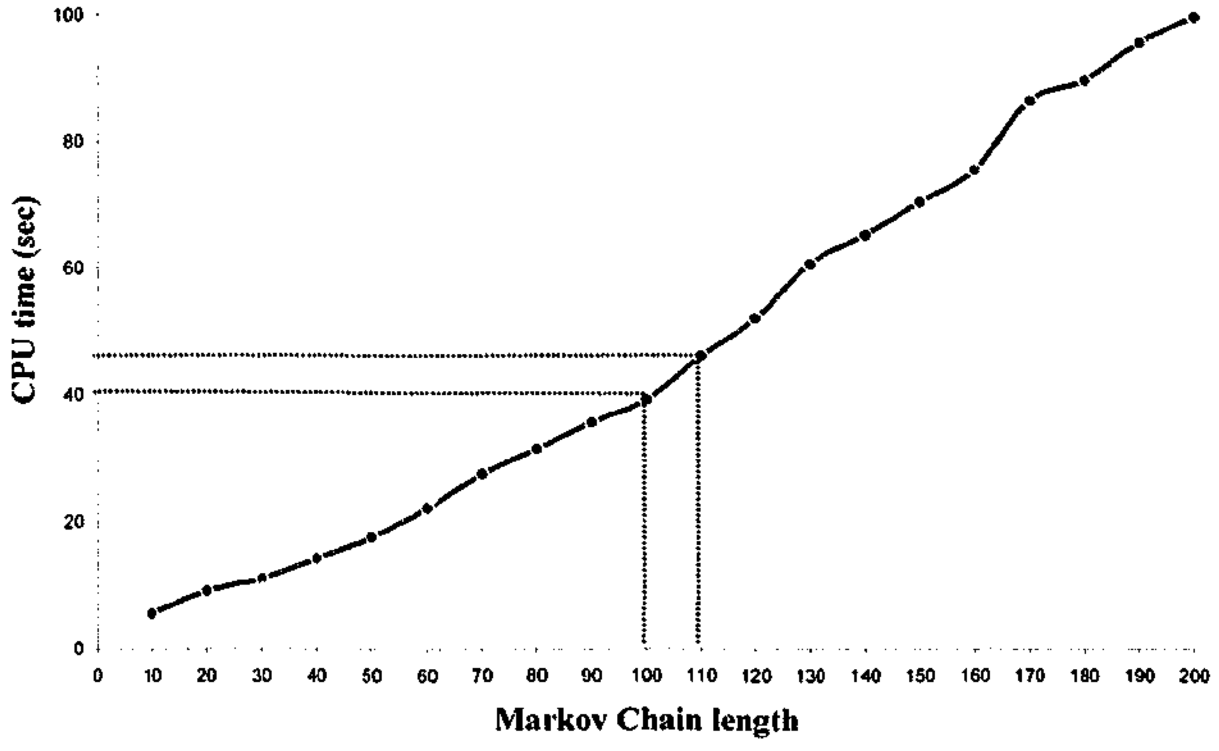


(b) CPU time required for different Markov Chain length

Figure 11.12 The evaluation of Markov Chain lengths in a 50-activity project

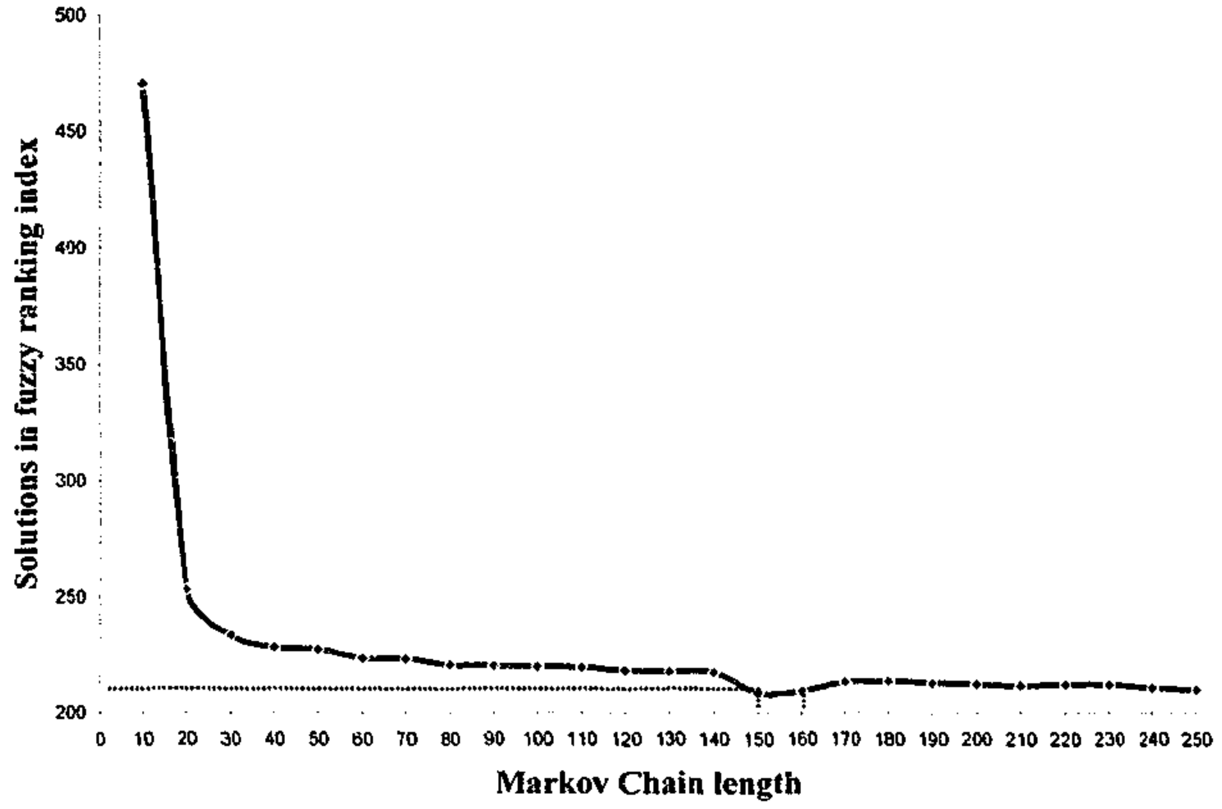


(a) Solutions with different Markov Chain length

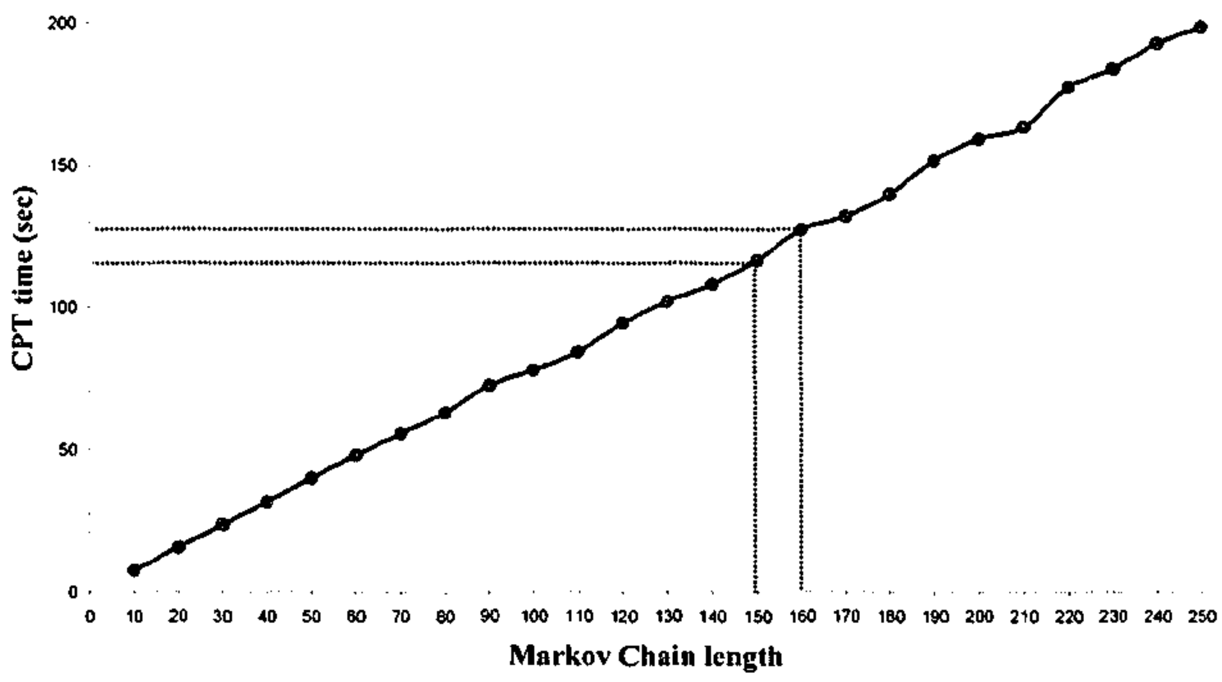


(b) CPU time required for different Markov Chain length

Figure 11.13 The evaluation of Markov Chain lengths in a 100-activity project

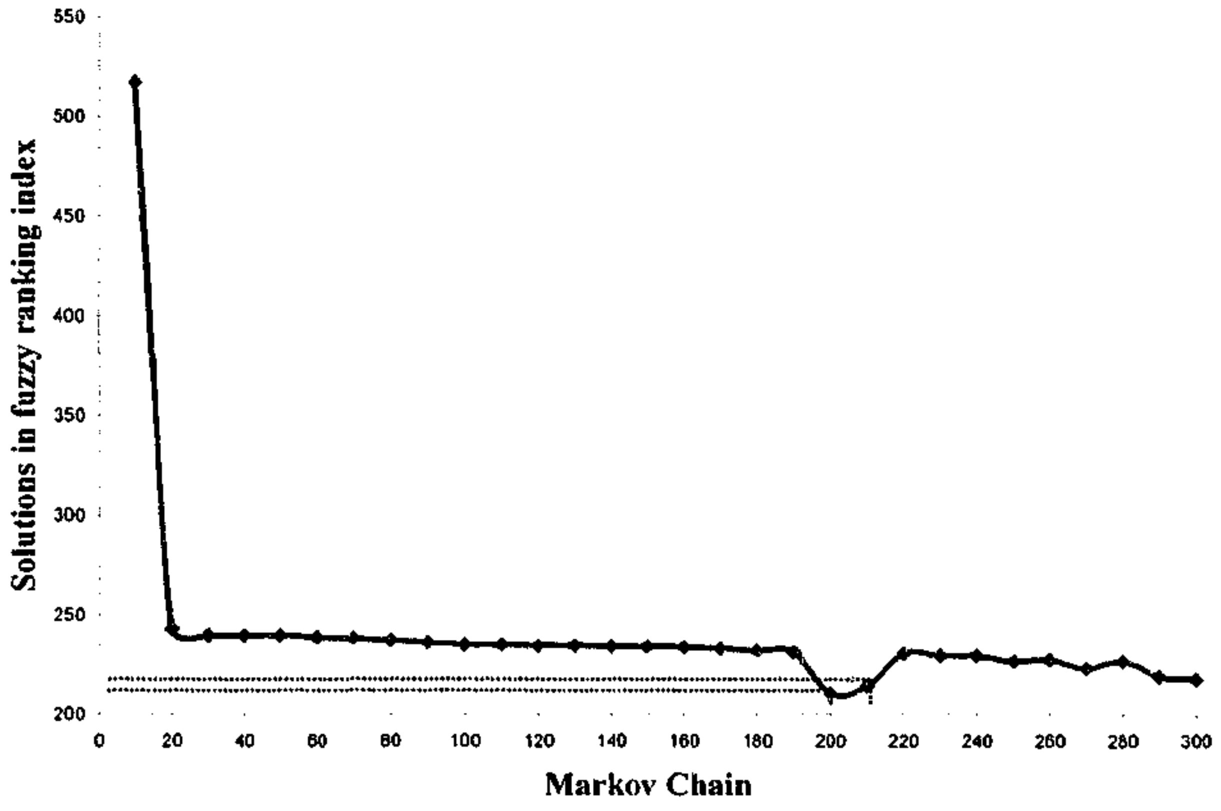


(a) Solutions with different Markov Chain length

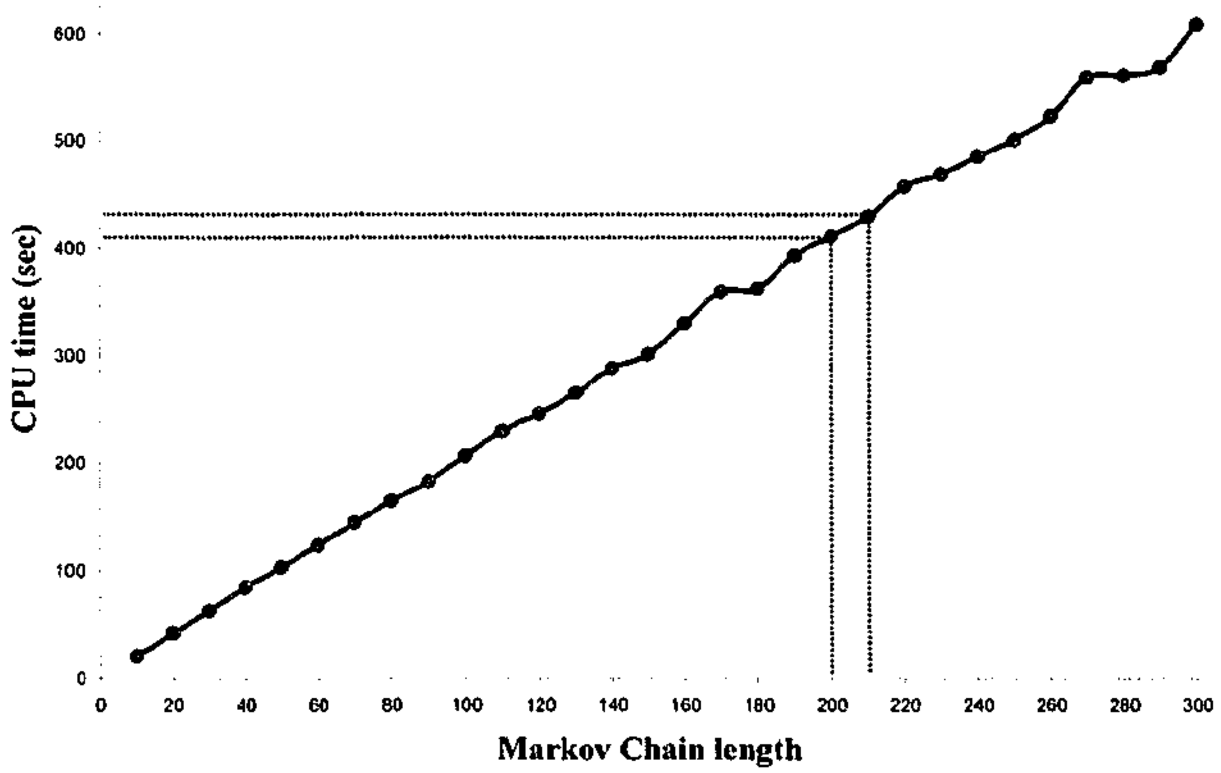


(b) CPU time required for different Markov Chain length

Figure 11.14 The evaluation of Markov Chain lengths in a 150-activity project



(a) Solutions with different Markov Chain length



(b) CPU time required for different Markov Chain length

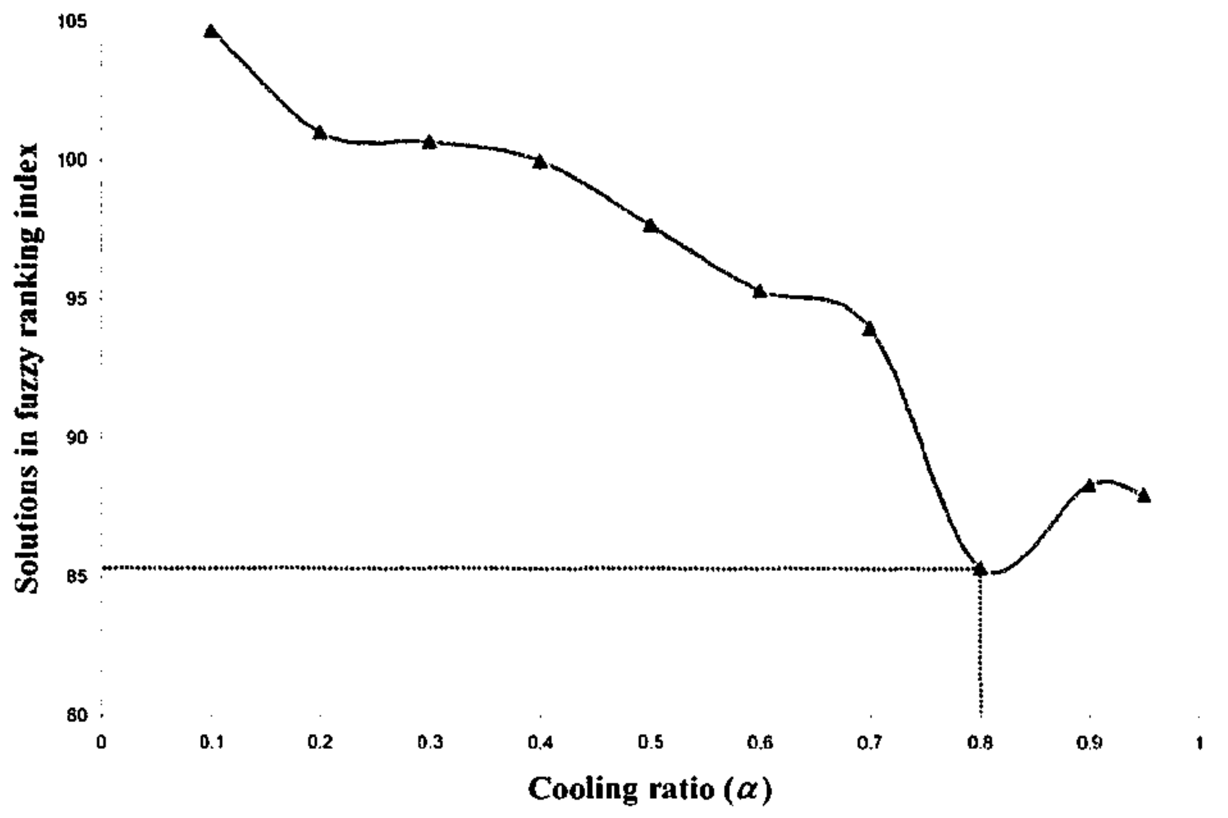
Figure 11.15 The evaluation of Markov Chain lengths in a 200-activity project

The experiments summarised above evaluate appropriate Markov Chain lengths for the five projects with different sizes. The appropriate Markov Chain length is approximately equivalent to the number of activities in a project. The experiments indicate that the solution quality is poor and unstable if Markov Chain length is small. The experiments also demonstrate that the solutions fluctuate greatly when the Markov Chain length is changed for projects with less than 150 activities. However, Figures 11.11-11.13 indicate that there is a range of the Markov Chain lengths where solution points suddenly decrease significantly. That is, the best global solutions fall within this range. Such a range of the values can be set as the appropriate Markov chain length for fuzzy SA. When the number of activities of a project is over 150, the solution changes are stable as shown in Figures 11.14-11.15. There is also a specific range of the Markov Chain length, within which the best solutions can be obtained.

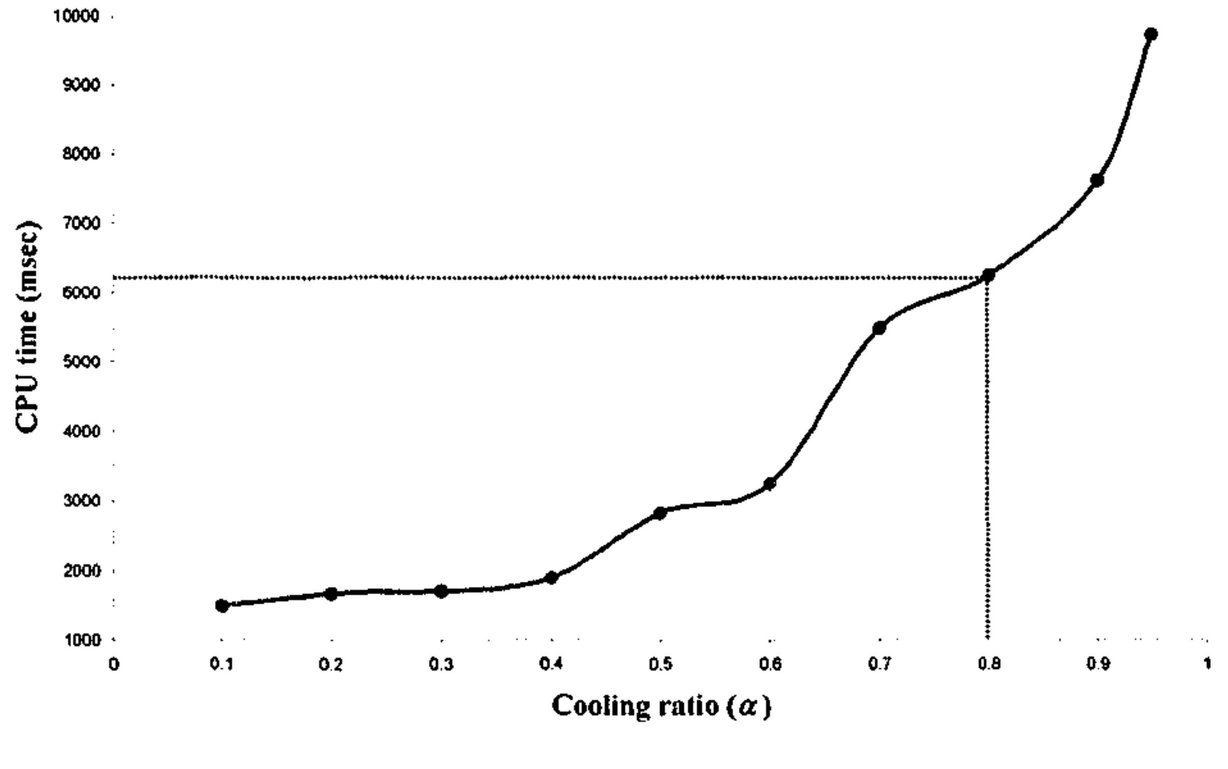
## 11.3.2 Evaluation of Cooling Ratio

The cooling ratio is an important factor in controling the annealing schedule, which dictates how the temperature is reduced throughout the whole process in order to effectively search for an approximate optimal solution in project scheduling. Although it is claimed that the appropriate cooling ratio may be set to about 0.8 for general applications, the following experiments are carried out to appropriate ratios for the individual projects when a fuzzy SA is applied in a situation where Markov Chain length is set to the number of activities of a project and the initial temperature is set as the number of activities in a project multiplied by the factorial of the maximum modes involved in scheduling.

Figure 11.16 shows the solution quality and its corresponding CPU time when cooling ratio changes for a project of 25 activities. As clearly shown in Figure 11.16(a), the fuzzy ranking index of the solution is at the lowest point when the cooling ratio is around 0.8 and the corresponding computational time required is about 6 seconds as shown in Figure 11.16(b).



(a) Solutions with different cooling ratios



(b) CPU time required for the different cooling ratios

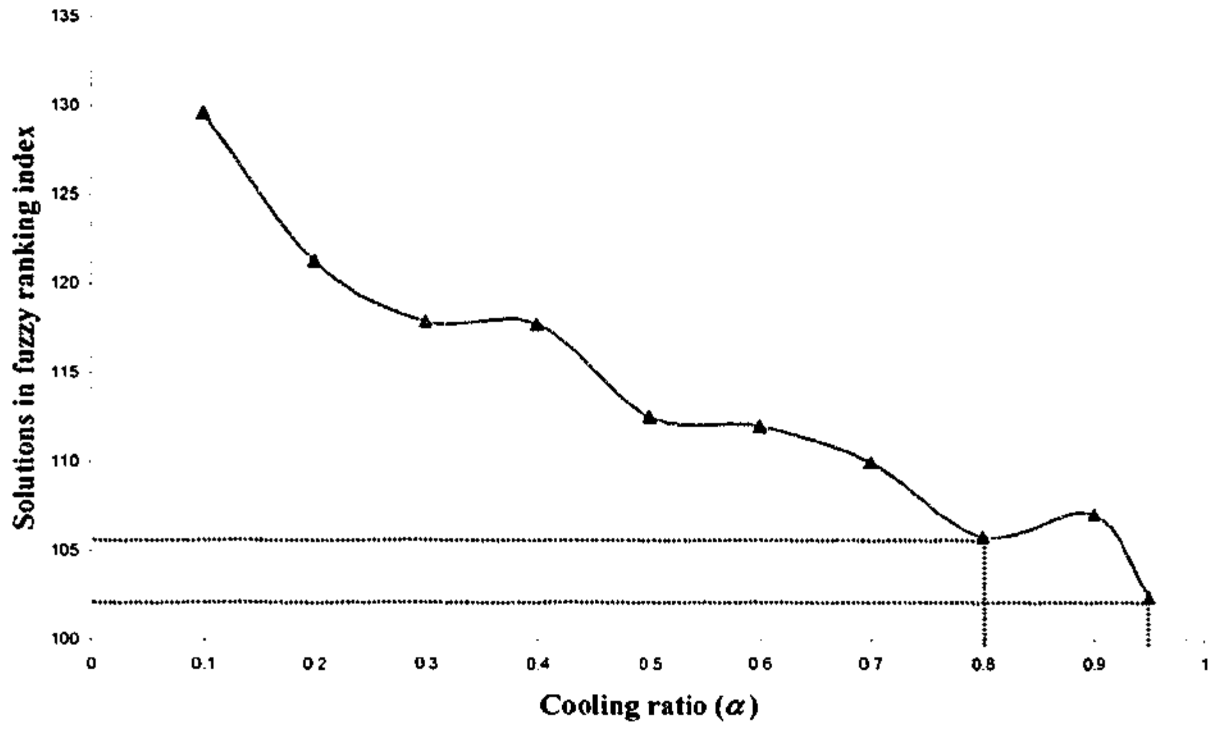
Figure 11.16 Experiment with different cooling ratios for a project of 25 activities

Figure 11.17 shows the solution status and its corresponding CPU time when the cooling ratio changes in a project of 50 activities. As shown in Figure 11.17(a), the solutions are improved when the cooling ratio is increased. The best solution is at the cooling ratio of 0.95. At this point, the computational time is significantly increased, requiring 74 seconds. The second best solution is at the cooling ratio of 0.8 where the computational time required is only 18 seconds. These two solutions are not greatly different.

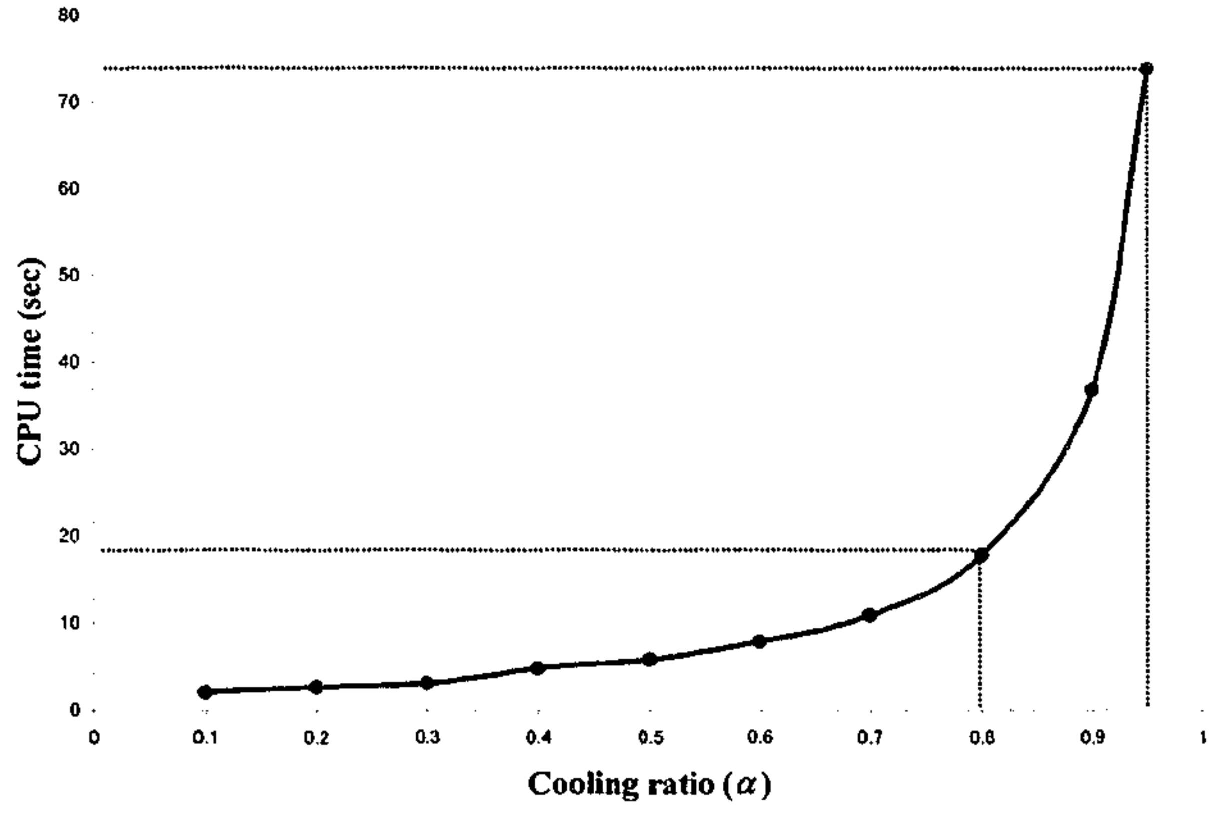
The solution quality with different cooling ratios for a project of 100 activities is shown in Figure 11.18. As the cooling ratio increases, solutions are improved and the best solution is found with a cooling ratio of about 0.8, requiring the computational time of 107 seconds as shown in Figure 11.18(b). However the computational time significantly increases and solutions are not really improved when the cooling ratio further increases.

Figure 11.19 shows the solution status and the computational time required for a project with 150 activities. When the cooling ratio is greater than 0.2, the solution quality is quite stable. But the better solutions are at the cooling ratio of 0.8 and 0.95. But the computational time required for the cooling ratio of 0.95 is 1028 seconds whereas the computational time is only 233 seconds at the cooling ratio of 0.8 as shown in Figure 11.19(b).

Figure 11.20 is the experimental result for solution quality when the cooling ratio is changed for a project with 200 activities. The solution quality is stable when the cooling ratio is over 0.2. Good solutions are found at the cooling ratios of 0.8 and 0.95 with corresponding computational times of 460 seconds and 1975 seconds respectively. However, a better solution obtained at the cooling ratio of 0.8 requires only a quarter of the computational time required at the cooling ratio of 0.95.

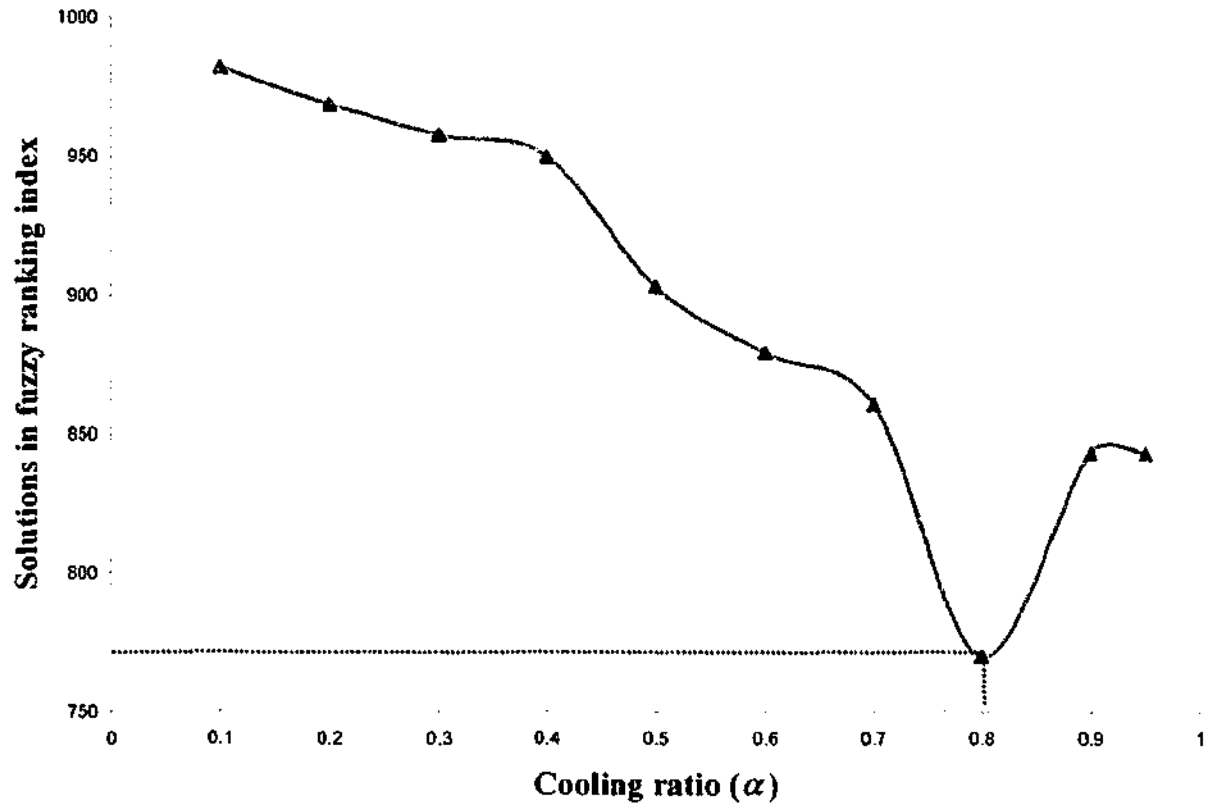


(a) Solutions with different cooling ratios

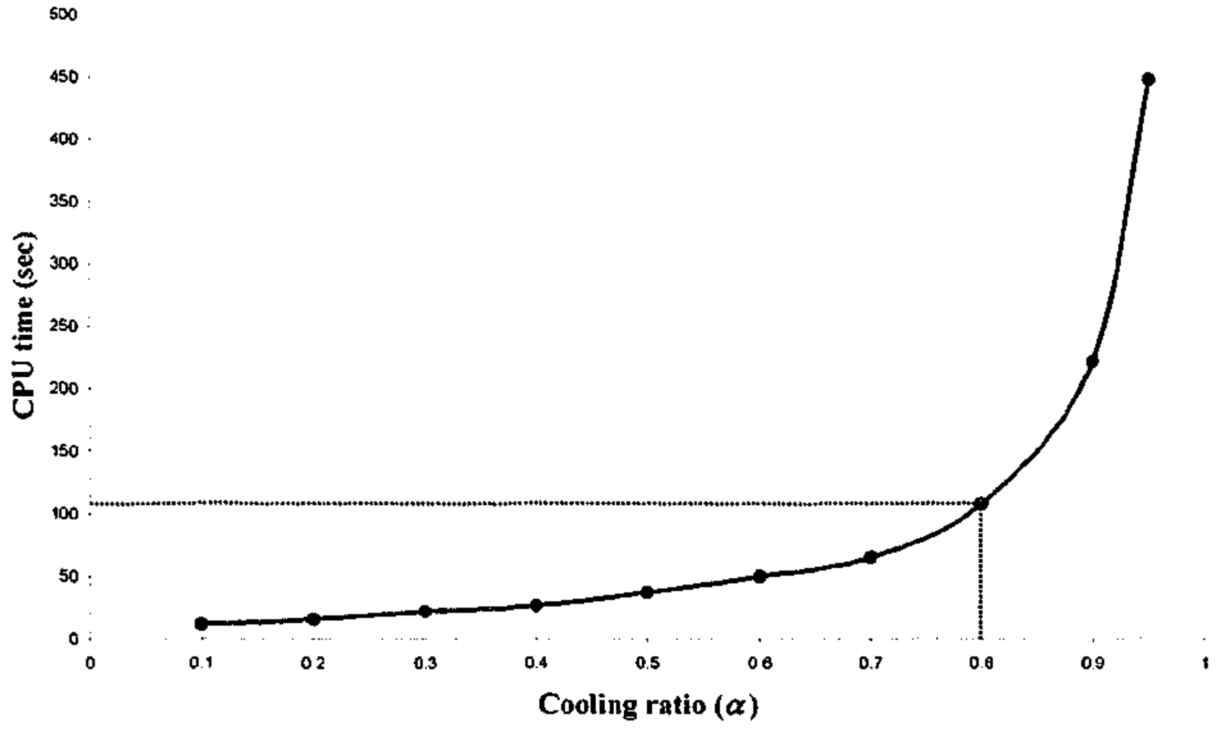


(b) CPU time required for the different cooling ratios

Figure 11.17 Experiment with different cooling ratios for a project of 50 activities

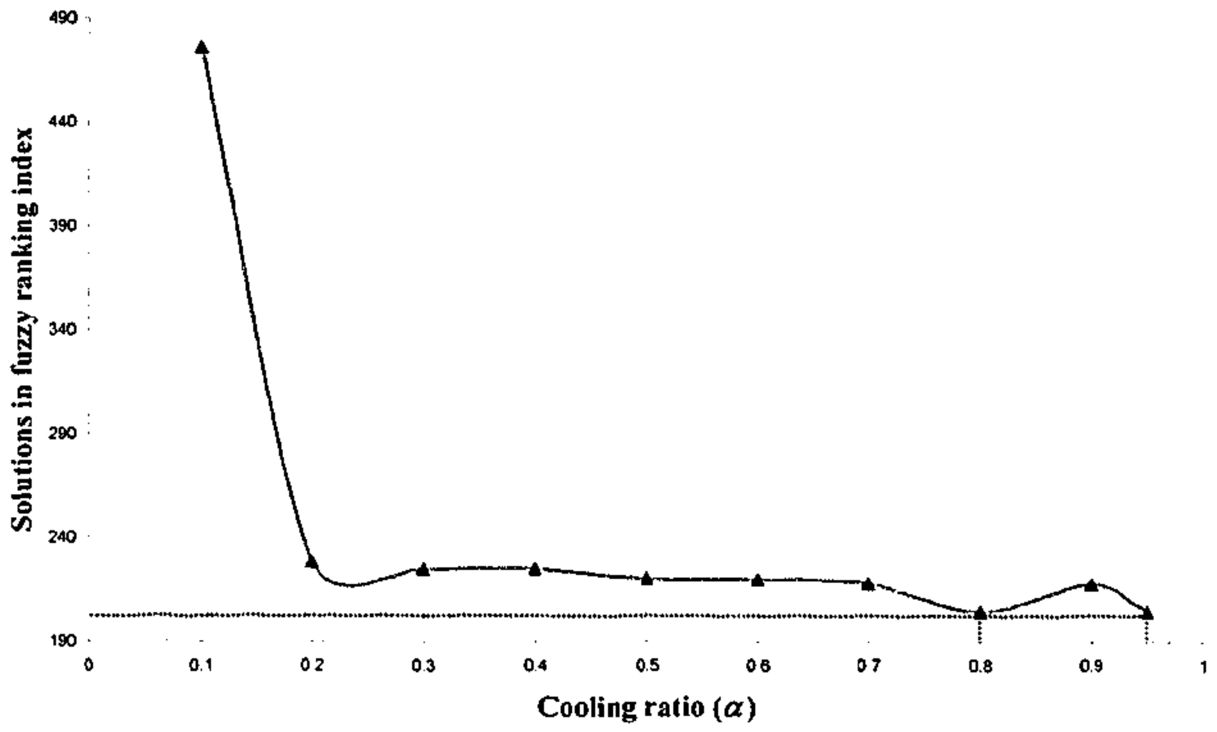


(a) Solutions with different cooling ratios

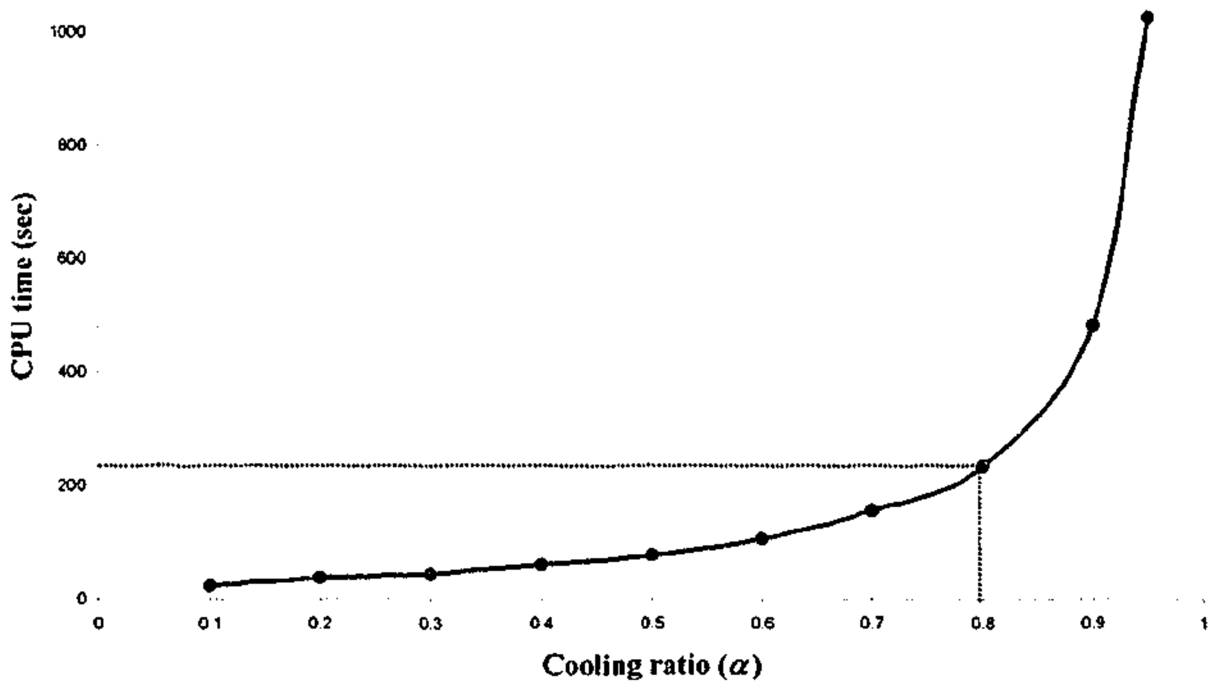


(b) CPU time required for the different cooling ratios

Figure 11.18 Experiment with different cooling ratios for a project of 100 activities

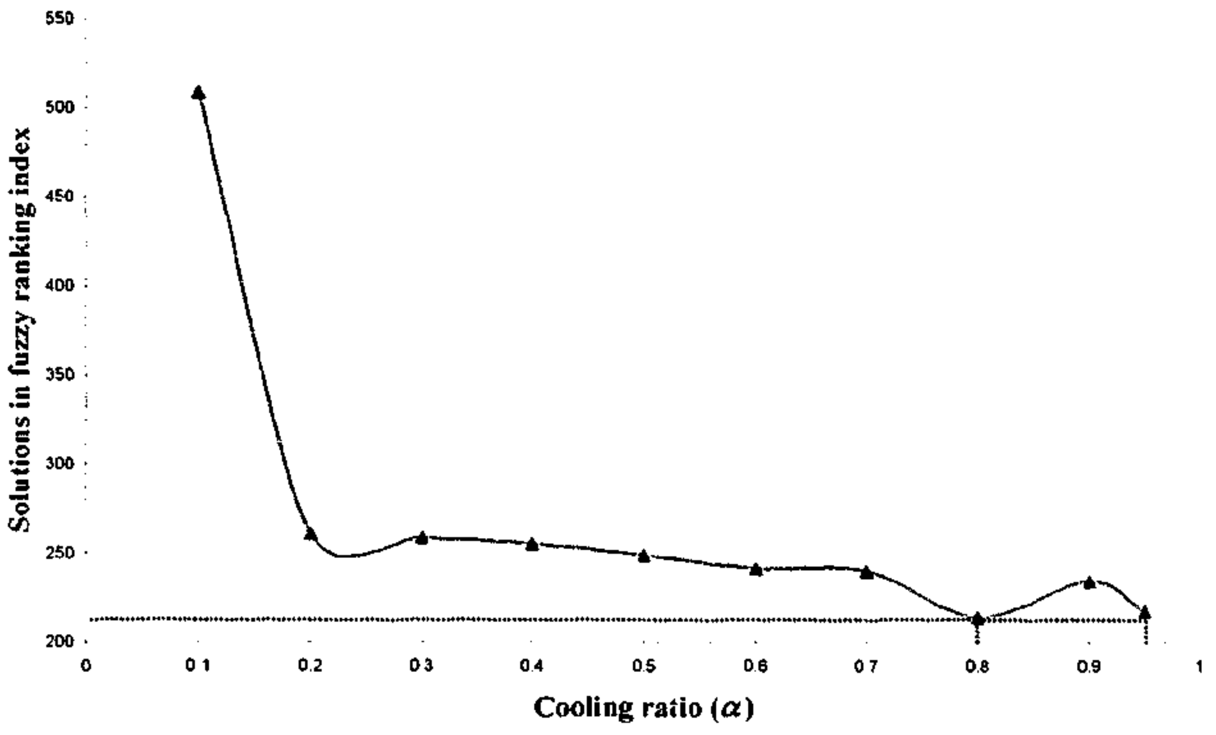


(a) Solutions with different cooling ratios

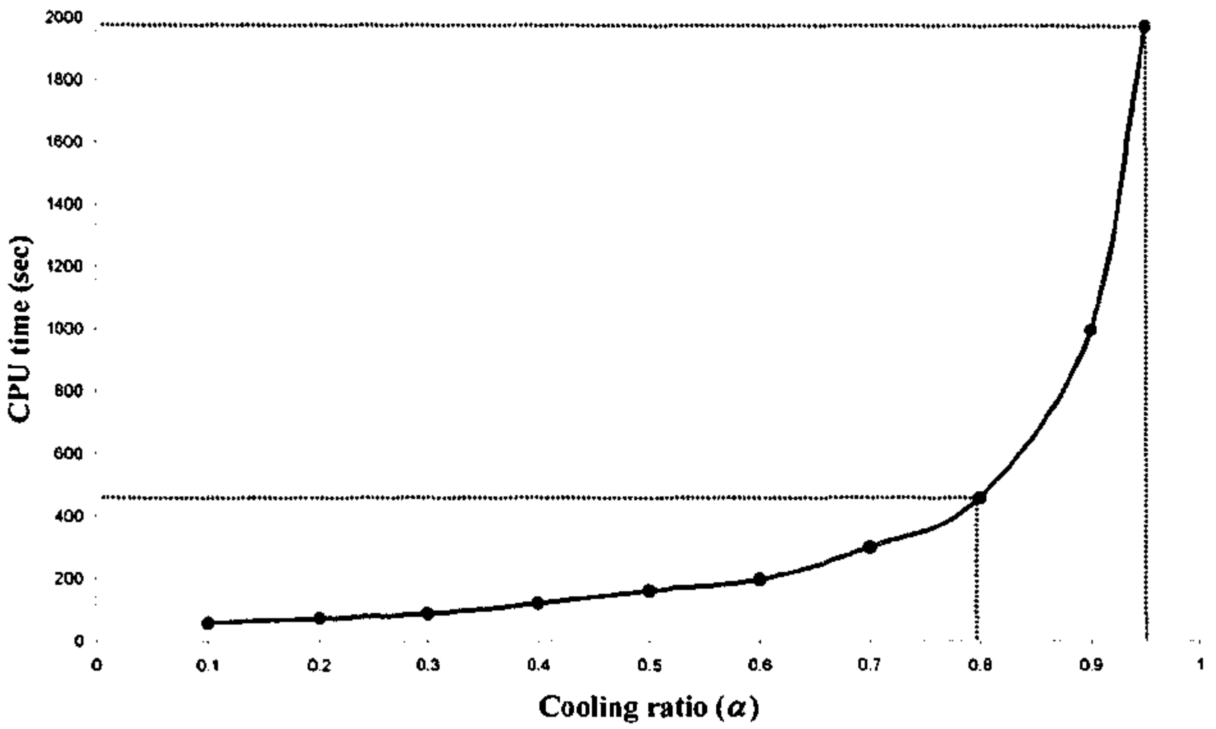


(b) CPU time required for the different cooling ratios

Figure 11.19 Experiment with different cooling ratios for a project of 150 activities



(a) Solutions with different cooling ratios



(b) CPU time required for the different cooling ratios

Figure 11.20 Experiment with different cooling ratios for a project of 200 activities

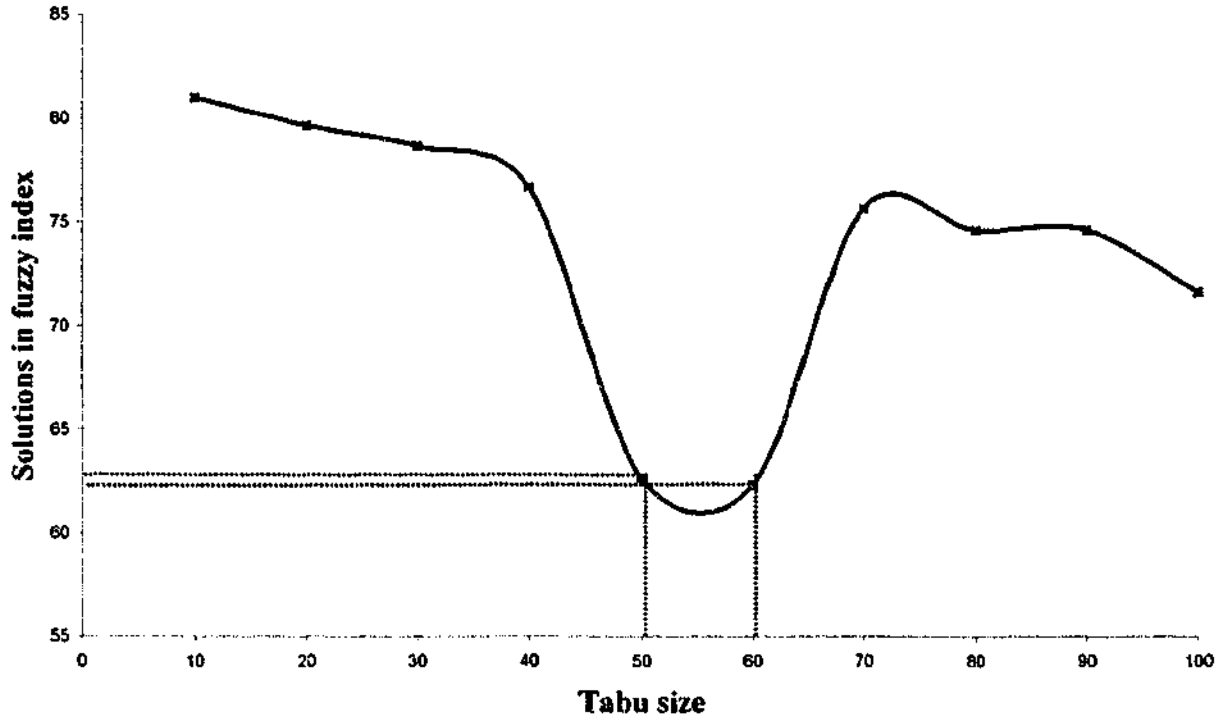
The above experiments demonstrate that the trend of solutions is improved as the cooling ratio increases. However, when the cooling ratio is over 0.8, solutions are not much improved, and the computational time increases exponentially as shown in Figures 11.16-11.20. Therefore, the appropriate cooling ratio can be chosen to be approximate 0.8 in general, although occasionally equivalent or slight better solutions can be obtained at the cooling ratio of 0.95.

## 11.3.3 Determination of Tabu Size

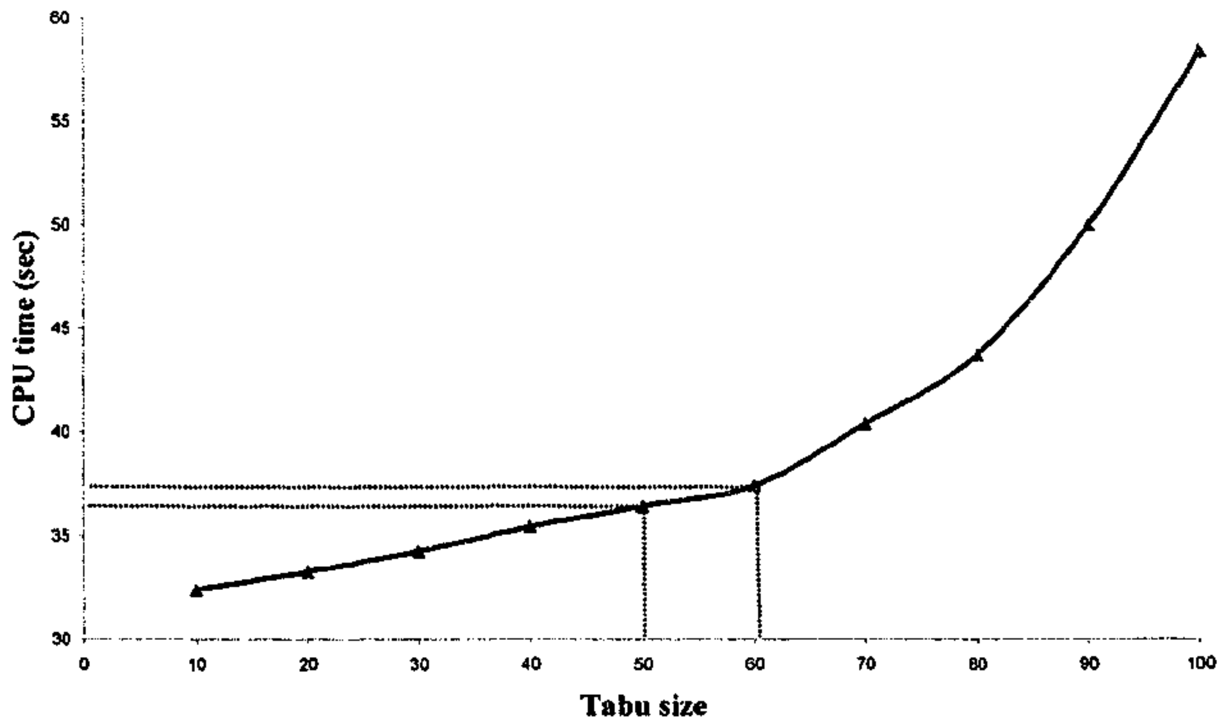
The tabu mechanism with fuzzy SA is designed to monitor the past search behaviour in order to avoid trapping in the regions that have been searched recently. Combining the tabu mechanism with fuzzy SA provides a more effective search by exploring more various solutions in the entire search space. However, an inappropriately large size of tabu may significantly increase the amount of the computational time required to implement the approach. Moreover, the quality of solutions is not improved significantly. The experiments conducted here are to evaluate the appropriate size of tabu for each individual project.

Figure 11.21 shows the solution performance with various tabu sizes for a project with 25 activities. Figure 11.21(a) clearly shows that the appropriate tabu size for this size of the project is between 50 and 60. In this range, the better solutions are obtained with the corresponding CPU time ranging from 36 to 37 seconds. However, when the tabu size increases beyond this range, the solutions are not improved significantly, and consequently, the computational time required becomes significantly higher as indicated in Figure 11.21(b).

The experimental result for the solution quality with the different tabu sizes is shown in Figure 11.22, where a project contains 50 activities. Clearly, the better solutions are obtained when the tabu size ranges from 50 to 60, in which range the CPU time required is about 105 seconds. When the tabu size is further increased, the CPU time is increased exponentially and the solution quality is not improved significantly. Therefore the appropriate tabu size for a project with 50 activities and 3 performance modes is in the range of 50 and 60.

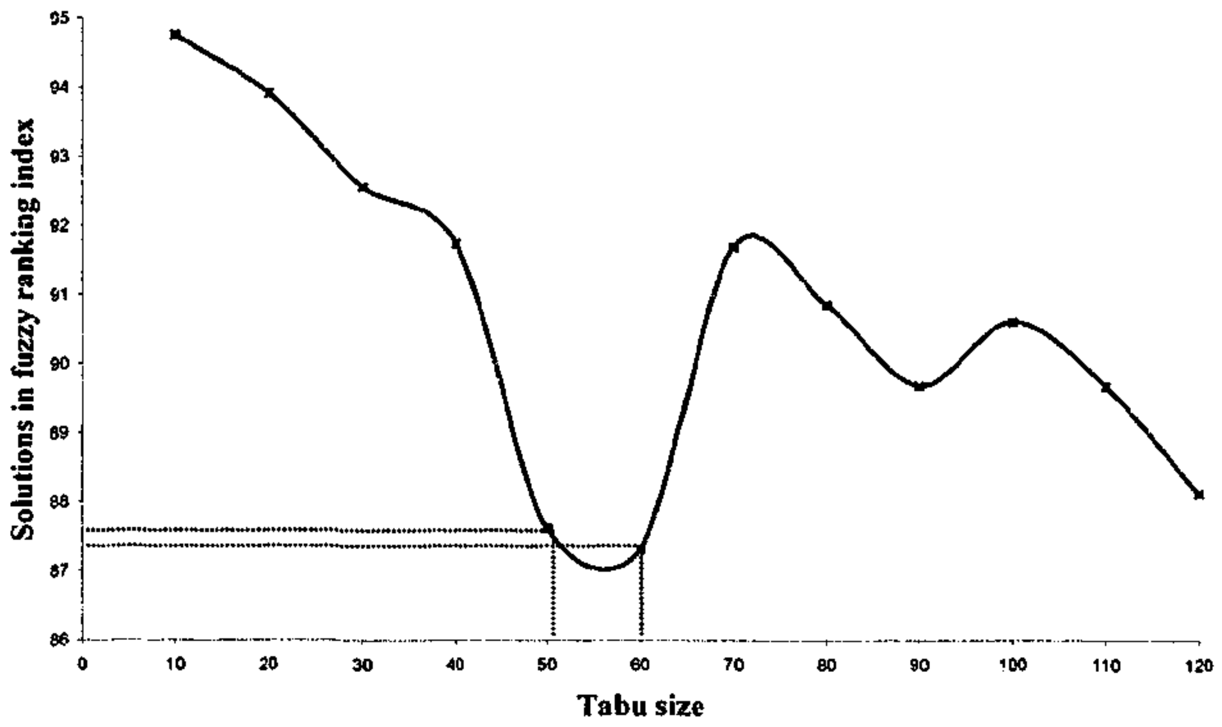


(a) The solution quality with various tabu sizes

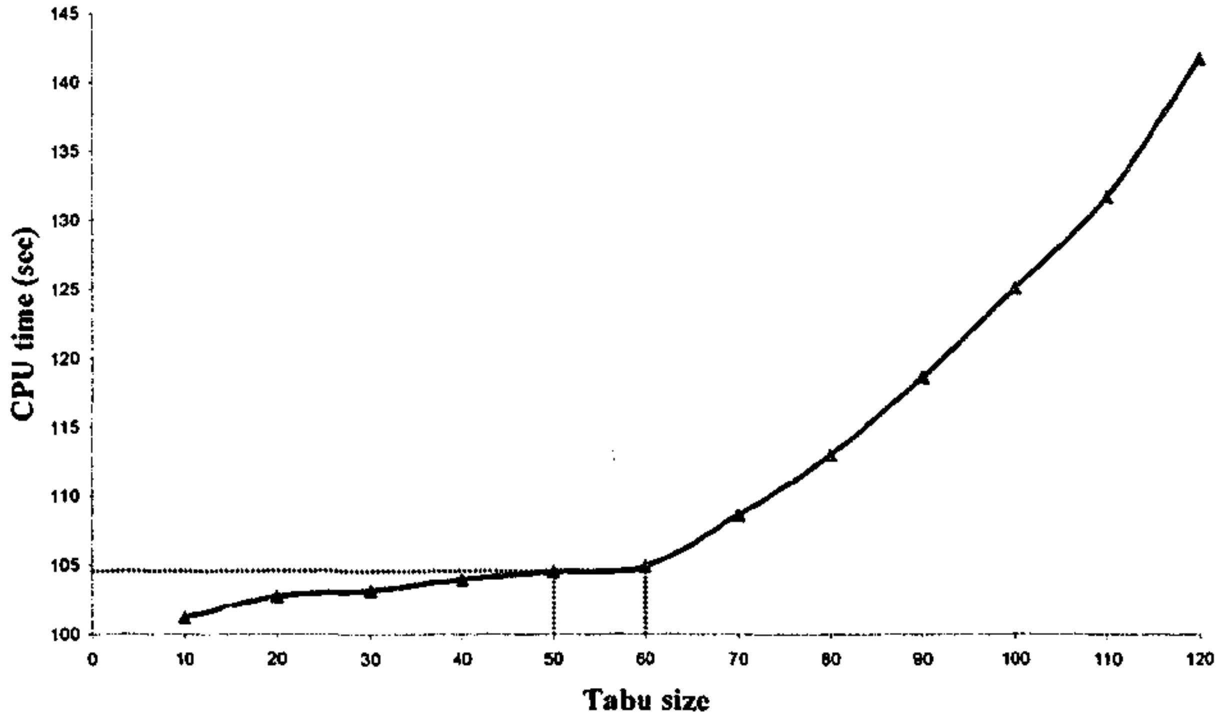


(b) CPU time required for various tabu sizes

Figure 11.21 The evaluation of tabu size for a project with 25 activities



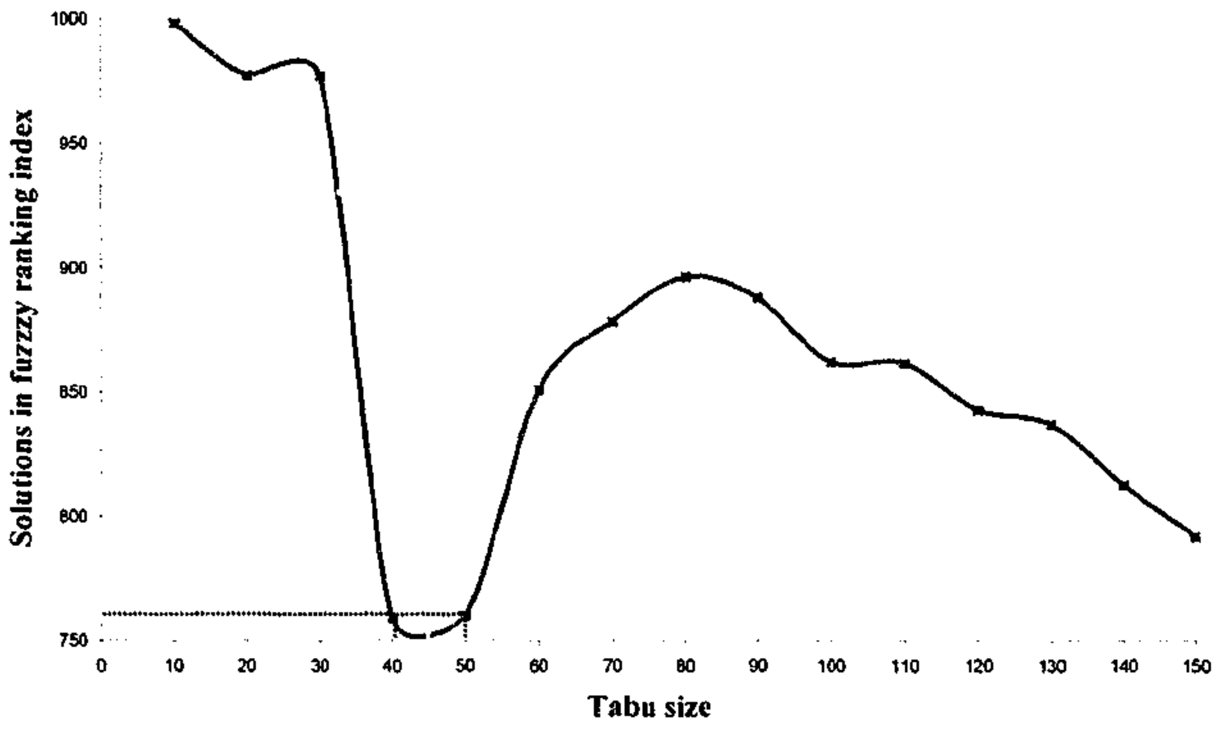
(a) The solution quality with various tabu sizes



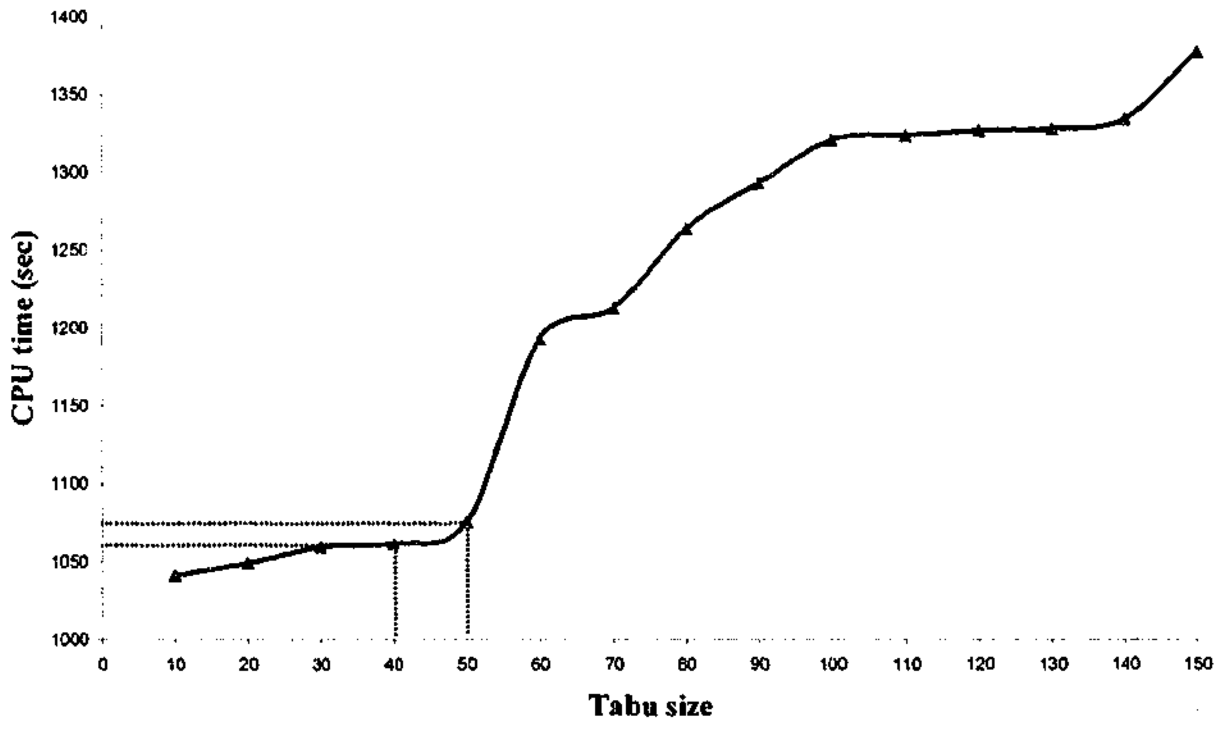
(b) CPU time required for various tabu sizes

Figure 11.22 The evaluation of tabu size for a project with 50 activities

When a project with 100 activities is implemented using the fuzzy SA with tabu approach, the CPU times required are significant. Figure 11.23 shows the variation of solution quality with various tabu sizes. The better solutions are found when the tabu size falls in the range of 40 and 50. The corresponding CPU times required are in the range of 1062 and 1072 seconds. When the tabu size increases further, the computational time escalates rapidly. However, the solutions are not improved significantly. Therefore, the appropriate tabu size is from 40 to 50 when the project size contains 100 activities and 3 performance modes.



(a) The solution quality with various tabu sizes



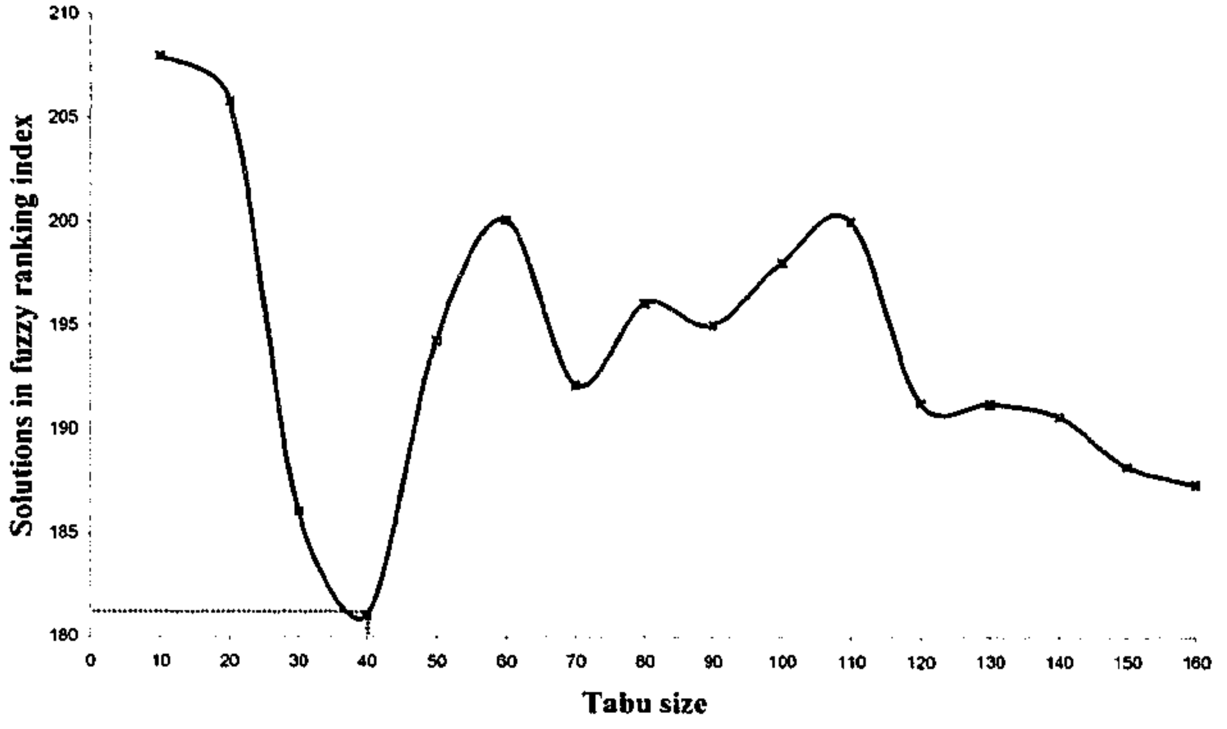
(b) CPU time required for various tabu sizes

Figure 11.23 The evaluation of tabu size for a project with 100 activities

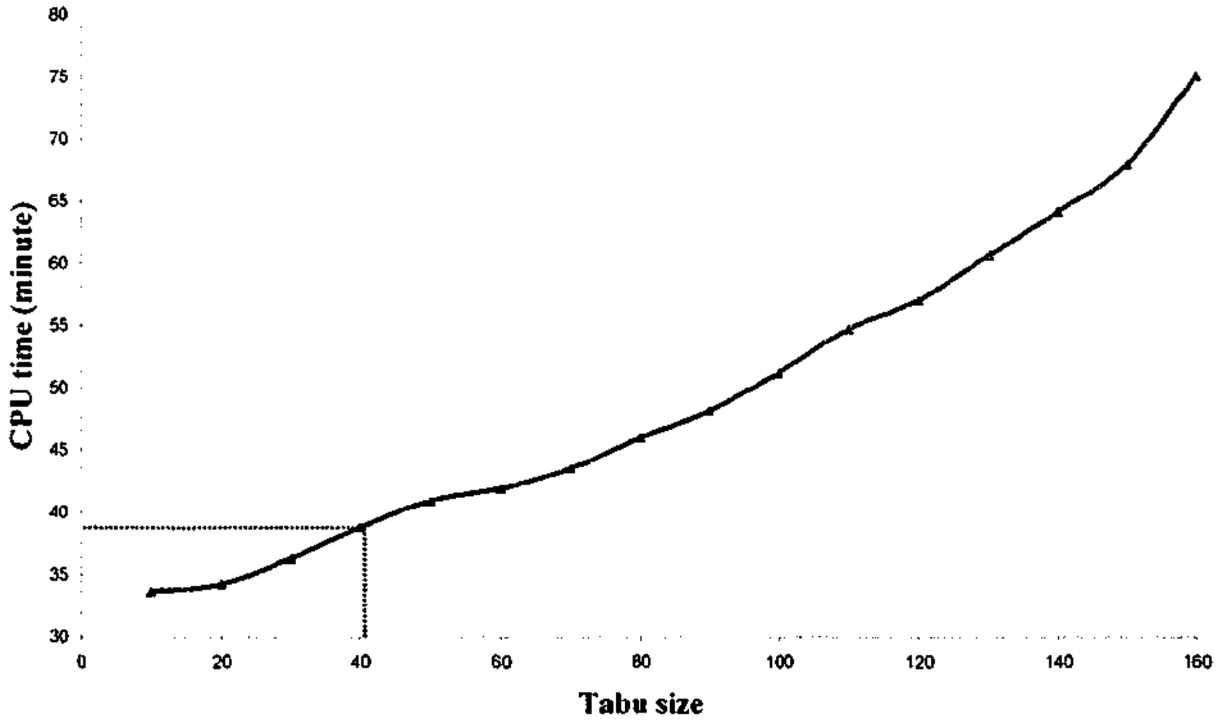
For the project size of 150 activities, the experimental result for evaluating the appropriate tabu size is shown in Figure 11.24. The solution quality is very poor when the tabu size is less than 30. The solution turning point is at the tabu size of 40. However, when the tabu size further increases, the solution quality shows marginal improvement as shown in Figure 11.24(a). Therefore, the appropriate tabu size is about 40 for the project size of 150 activities. It is noted in Figure 11.24(b), that the computational time is significantly increased when tabu size further increases.

Figure 11.25 shows the solution quality when the tabu size changes for a project of 200 activities. When the tabu size is less than 40, the solution quality is poor. When the tabu size is over 50, the solution quality begins to improve, and then to move down a bit in terms of minimising the fuzzy project completion time, but the solution is not improved significantly. Therefore, the appropriate tabu size for the projects of 200 activities ranges from 40 to 50 and the corresponding CPU time is about 193 minutes. As shown in Figure 11.25(b), the computational time is increased dramatically when a project size is over 150.

The experiments reported in this section evaluate the appropriate tabu size of fuzzy SA with tabu mechanism. The solutions fluctuate when the tabu size changes in these five project instances. However, the globally best solutions, rather than locally best ones are detected within a certain range of the tabu size for each project instance, as shown in Figures 11.21-11.25. These ranges of the tabu size are chosen to be appropriate for these individual instances. The above experiments also demonstrate that solutions obtained using this approach all five project instances are better than solutions using any of the other three approaches. However, the time spent on finding a good optimal solution is much longer than any of the other three metaheuristic approaches. If the computational time is not a major concern for implementing a schedule, the fuzzy SA with tabu approach is the best one among these four approaches in general. To measure the performance of these four metaheuristic approaches, in terms of finding a good global solution, a comparative analysis is conducted in the following section.

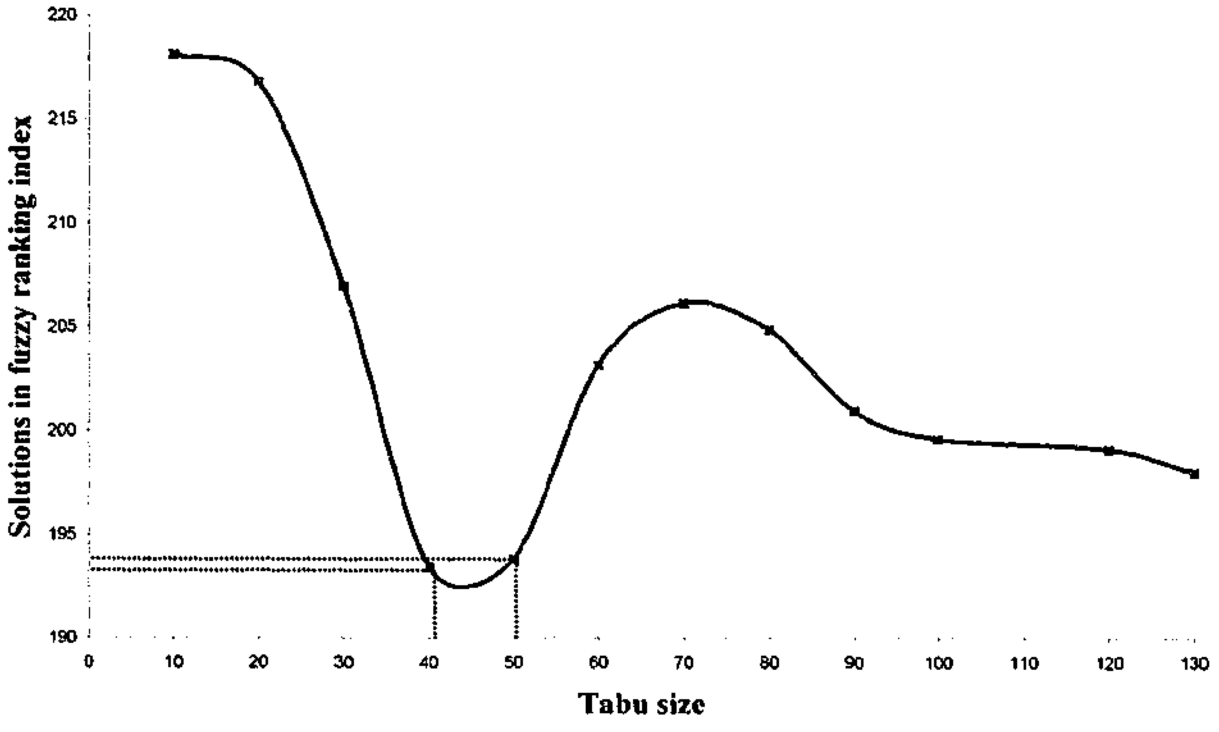


(a) The solution quality with various tabu sizes

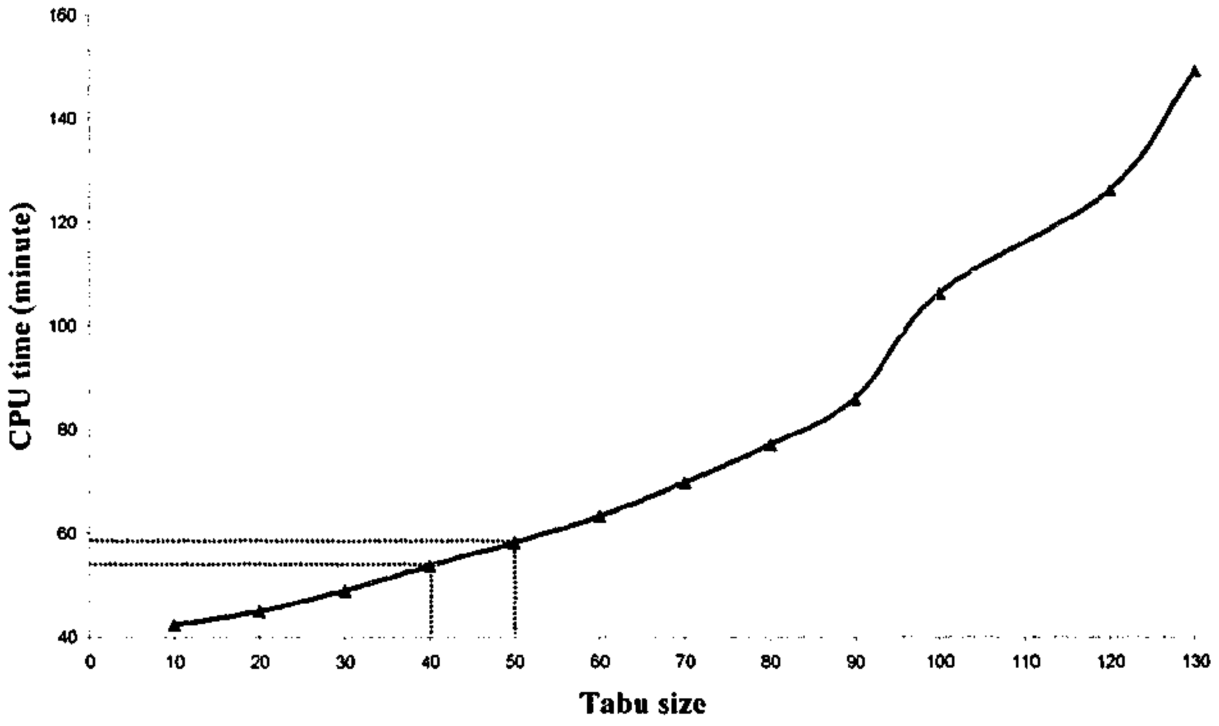


(b) CPU time required for various tabu sizes

Figure 11.24 The evaluation of tabu size for a project with 150 activities



(a) The solution quality with various tabu sizes



(b) CPU time required for various tabu sizes

Figure 11.25 The evaluation of tabu size for a project with 200 activities

## 11.4 Comparative Studies

The experiments with parameter settings reported above provide appropriate values of key parameters for individual metaheuristic approach that is applied to the five project sizes. These appropriate values are used in a comparative analysis of the four metaheuristic approaches. The value settings for key parameters represent the best solution of individual approaches for different project sizes. Therefore, these values may reasonably be used to examine the performance behaviours of these approaches, thus determining the solution differences of these approaches when each approach is applied to the project sizes generated by the FMMRCPS system.

To compare these four metaheuristic approaches, each approach is run 100 times for each of the five project sizes so as to examine their deviations from the optimum solution and whether the solutions obtained are stable in terms of distributions around the optimum solution in randomised nature.

Table 11.1 lists three types of deviations calculated by the FMMRCPS system for the five different-sized projects. The av ge deviations (AD) are the average distances from the optimum solution expressed in percentage. These values of AD are quite reasonable and stable ranging from 17 % to 24 % for all the projects across the four approaches. The maximum deviation (MD) is an absolute deviation between a worse solution and the optimum solution over a number of runs. Such a type of deviation only gives the difference between the optimum and the worse solution intuitively. However, it should be noted that a large value of MD does not necessarily mean that the AD and the standard deviation (SD) will also be large. This is because the large value of the project completion time likely has the large maximum deviation when a metaheuristic approach is implemented for a project over number of runs. As a result, the maximum deviations obtained for these projects are different as shown in Table 11.1.

Table 11.1 Comparative analysis for the four metaheuristic approaches measured by deviations

| Project size | | 25 | 50 | 100 | 150 | 200 |
|---|---|---|---|---|---|---|
| Fuzzy GA | AD (%) | 21.68 | 22.31 | 21.11 | 23.21 | 23.98 |
| | MD | 13.13 | 25.12 | 32.25 | 25.54 | 44.89 |
| | SD (%) | 19.27 | 19.99 | 18.59 | 20.23 | 20.95 |
| Fuzzy GA with tabu | AD (%) | 18.54 | 20.05 | 18.95 | 21.78 | 22.01 |
| | MD | 10.54 | 23.45 | 29.65 | 23.91 | 43.77 |
| | SD (%) | 17.65 | 18.25 | 18.02 | 18.39 | 19.07 |
| Fuzzy SA | AD (%) | 21.01 | 21.65 | 20.01 | 22.98 | 22.56 |
| | MD | 12.54 | 24.89 | 23.25 | 24.35 | 45.25 |
| | SD (%) | 19.02 | 19.56 | 18.60 | 19.65 | 19.54 |
| Fuzzy SA with tabu | AD (%) | 17.63 | 19.69 | 18.02 | 21.07 | 21.65 |
| | MD | 12.09 | 24.68 | 26.21 | 22.31 | 43.01 |
| | SD (%) | 15.84 | 16.54 | 17.88 | 17.59 | 18.37 |

The standard deviation has the significant meaning of reflecting the normal frequency distribution about solution quality in random nature. All values of the SD obtained for the five projects range from 15% to 20%. Therefore, the results of the SD show that these four metaheuristic approaches perform well in terms of the solution quality when appropriate parameter settings are applied to the approaches.

With respect to individual approaches, the fuzzy SA with tabu outperforms the other approaches because this approach has smaller deviations in both the AD and SD. The second approach for performing the good solution quality is the fuzzy GA with tabu. These two approaches have narrow difference in solution performance because the differences between these two types of deviations are small, as listed in Table 11.1.

In addition to the comparative study of these four metaheuristic approaches using deviation analyses, these approaches can be further examined by evaluating the appropriate values of parameters for obtaining the best solutions in these four metaheuristic approaches. These best solutions are summarised in Table 11.2, collected from Sections 11.2-11.3. The best solutions listed in Table 11.2 clearly demonstrate that fuzzy SA with tabu is the best approach when the computational time is not considered. The second and third best approaches are fuzzy GA with tabu and fuzzy SA respectively. The performance of fuzzy GA alone is inferior to the other three metaheuristic approaches. However, fuzzy GA alone requires less computational time than the other three approaches. As shown in Table 11.2, both fuzzy GA with tabu and fuzzy SA with tabu need a great amount of computational time.

Table 11.2 The best solutions and their corresponding CPU times obtained by evaluating parameters four metaheuristic approaches

| Project size | Fuzzy GA | | Fuzzy GA with tabu | | Fuzzy SA | | Fuzzy SA with tabu | |
|---|---|---|---|---|---|---|---|---|
| | Solutions (*FRI) | CPU time (sec) | Solutions (*FRI) | CPU time (sec) | Solutions (*FRI) | CPU time (sec) | Solutions (*FRI) | CPU time (sec) |
| 25 | 88.22 | 2.187 | 65.335227 | 30.678 | 85.33479 | 6.25 | 62.335227 | 37.4171 |
| 50 | 103.6969 | 5.516 | 88.6856227 | 96.828 | 105.8239 | 17.859 | 87.315894 | 104.9312 |
| 100 | 785.9069 | 19.859 | 764.930030 | 1050.951 | 769.9233 | 107.89 | 749.056145 | 1061.7578 |
| 150 | 210.9546 | 48.578 | 186.86113 | 2054.4 | 204.3694 | 233.359 | 181.039228 | 2329.2 |
| 200 | 219.5052 | 89.594 | 193.67784 | 2674.4 | 214.0006 | 459.562 | 193.429206 | 3228.6 |

*FRI: fuzzy ranking index

Overall, the comparative studies in view of several types of deviation analyses indicate that these four metaheuristic approaches perform well for obtaining good and stable solutions. These approaches are particularly suited to solving FMMRCPS problems of practical sizes efficiently and effectively, whereas exact approaches are not available in realistic settings. In addition, experiments both in the deviation analyses and the data from parameter evaluation analysis suggest that there are some differences among these four metaheuristic approaches in terms of the solution quality and the CPU time required. The suitability of each approach in a particular situation will be addressed in Section 12.4. This section will also state the suitability of the other two approaches, the fuzzy hybrid goal programming approach and the fuzzy heuristic approach that were described in Chapters 6 and 7. These two approaches are not directly comparable with the four metaheuristic approaches, because the project environment and condition settings under which they are used are totally different from the four metaheuristic approaches.

## 11.5 Concluding Remarks

In this chapter, intensive experiments have been conducted: (a) to determine appropriate values of key parameters for individual approaches, and (b) to conduct a comparative analysis of the four metaheuristic approaches using appropriate parameter settings obtained in the experiments.

The experiments have suggested that the settings of key parameter influence the solution quality significantly when a fuzzy metaheuristic approach is applied. To get the best solution, appropriate values of parameters must be determined prior to solving any project scheduling problem using metaheuristic approaches.

The comparative analysis indicates that both hybrid approaches, fuzzy GA with tabu and fuzzy SA with tabu approaches, perform better than pure approaches such as fuzzy GA alone and fuzzy SA alone. However, hybrid approaches consume much more computational time than pure approaches. Experimental results, using the five project sizes randomly generated by the FMMRCPS system, have indicated that the fuzzy SA with tabu approach is slightly superior to the fuzzy GA with tabu approach.

# Chapter 12

# Conclusion

## 12.1 Introduction

This study has focused on developing six optimisation approaches for efficiently and effectively solving FMMRCPS of practical sizes with single or multiple objectives. FMMRCPS problems are complex as two subproblems need to be solved simultaneously in a fuzzy environment: (a) mode assignment, and (b) the sequence of activities in a schedule

A fuzzy hybrid goal programming approach has been developed, in conjunction with a rule knowledge base for mode assignment, for solving multiple objectives in FMMRCPS. Five heuristic and metaheuristic approaches have been developed to deal with the single objective of minimising the fuzzy project completion time. These five approaches are (a) a fuzzy heuristic approach, (b) a fuzzy GA, (c) a fuzzy GA with tabu, (d) a fuzzy SA approach, and (e) a fuzzy SA with tabu approach. These approaches provide a robust framework for solving complex FMMRCPS problems of practical sizes under situations where exact approaches are not viable.

This chapter summarises six optimisation approaches developed for solving practical FMMRCPS problems with single or multiple objectives. The features of the FMMRCPS system are outlined. The implications of six optimisation approaches are then highlighted from case studies and intensive experiments conducted by the FMMRCPS system. Finally, the contributions of this study are summarised, and suggestions for future research are presented.

## 12.2 Summary of Approaches Developed for FMMRCPS

FMMRCPS is a realistic project scheduling model, allowing activities to have flexible performance modes under resource constraints where activity duration times are fuzzy and each mode of an activity has its unique fuzzy duration time. This scheduling model covers most practical situations, and it is a generalised case in project resource-constrained scheduling. However, it is more complex than classical single mode project scheduling. To tackle FMMRCPS under realistic settings, six optimisation approaches have been developed for solving single or multiple objectives in my PhD studies. The six optimisation approaches are listed in Table 12.1 and briefly summarised below.

Table 12.1 The features of individual approaches developed in my PhD studies

| Objective | Approach | Features |
|---|---|---|
| Multiple ⇓ Minimising the project completion time and the project cost | Fuzzy hybrid goal programming | Decomposing complex FMMRCPS into simple single mode project scheduling |
| Single ⇓ minimising the project completion time | Fuzzy heuristic approach | A fast and simple way in solving FMMRCPS problems of realistic sizes |
| | Fuzzy GA | Chromosomes designed provide insights into the nature of FMMRCPS for obtaining an approximate optimal solution efficiently and effectively |
| | Fuzzy GA with tabu | Generating a wide variety of solutions, thus reducing the probability of missing a good nearly global optimal solution |
| | Fuzzy SA | Using a time pointer to avoid a full schedule in neighbourhood generations, thus saving enormous computational time |
| | Fuzzy SA with tabu | Forbidding the search of neighbourhoods already visited to explore more search areas in digging the best solution |

To solve multiple objectives in FMMRCPS, a fuzzy hybrid goal programming approach has been developed by combining a rule knowledge base for mode assignment with fuzzy goal programming. The mechanism of the rule knowledge base constructed here is to allocate modes to activities accurately and efficiently, based on both resource availabilities and criticalities on paths in the project network at the scheduled time. In addition, the rule knowledge base incorporated into fuzzy goal programming can decompose a complex FMMRCPS problem into a simpler single mode project scheduling problem during the solution procedure. The approach has been applied to the practical case study of a dredge repair project, and the result shows that this novel approach provides a realistic framework to solve FMMRCPS by yielding the best compromised solution where the objectives conflict with each other.

To solve the single objective of minimising the fuzzy project completion time, five heuristic and metaheuristic approaches have been developed. These approaches provide a methodological framework for solving complex FMMRCPS problem of realistic sizes, which exact approaches are not viable. These five approaches are briefly presented individually below.

A fuzzy heuristic approach developed for FMMRCPS is composed of three components: (a) priority rules for determining activity priorities in scheduling, (b) a mode assignment policy for allocating modes to individual activities, and (c) a fuzzy scheduling mechanism to schedule activities in a fuzzy environment, once both activity modes and priorities have been determined by (a) and (b). To get a good scheduling result in terms of minimising the fuzzy project completion time, a set of priorities rules and mode policy are applied to the fuzzy scheduling mechanism simultaneously. To demonstrate the practicability of this approach, a real case of a dredge overhaul project has been studied. The case study has shown that this approach gives reasonable practical solutions in a simple and fast way for solving realistic FMMRCPS problems where traditional heuristic approaches suffer from the major shortcoming of being unable to deal with fuzziness.

A fuzzy GA approach applies both fuzzy forward and backward scheduling simultaneously to new generated chromosomes, thus allowing the survival of fitter chromosomes and the elimination of weaker chromosomes for minimising the fuzzy project completion time. To perform fuzzy GA effectively, a chromosome, representing a schedule, has been carefully designed to always produce feasible schedules during the process of fuzzy GA. In addition, chromosomes designed for FMMRCPS avoid the decoding procedure that is often required in GA. Furthermore, the chromosome designed can provide insight into the nature of complex FMMRCPS, thus enabling the approach to be applied more efficiently in order to procure a good globally optimal solution.

A fuzzy GA with tabu approach is developed, based on GA, into which the tabu mechanism is incorporated. In addition to the functions of fuzzy GA, fuzzy GA with tabu has the extra function of preventing newly generated chromosomes from being the same as those generated recently in order to have more opportunities to generate more various chromosomes. Thus, the best-fit chromosomes may hardly escape from chromosome generations globally in the search space. This approach is one of the effective ways of searching for an approximate globally optimal solution in terms of minimising the fuzzy project completion time.

A fuzzy SA approach is developed for obtaining a good approximate globally optimal solution through searching neighbourhoods by perturbing both activity modes and priorities from the current individual schedules, and subsequently by using fuzzy forward and backward schedule mechanisms. The solution representation is an important component in fuzzy SA. In the approach, the solution representation designed not only directly reflects the nature of FMMRCPS, but can also be easily manipulated for neighbourhood generations in the process of fuzzy GA. More importantly, a novel technique (called a time pointer) has been proposed to avoid a full scheduling process each time when a new neighbourhood is generated, thus saving the enormous computational time.

A fuzzy SA with tabu approach is built up on the basis of SA to enhance its search ability effectively for a globally optimal solution. This integrated tabu mechanism avoids returning to previous search areas in the search process that may occur when applying the fuzzy SA approach. The tabu mechanism built into fuzzy SA keeps track of the search attributes of recently visited neighbourhoods and the purpose of this mechanism is to explore more search areas in order to dig for a better global solution.

## 12.3 FMMRCPS System

The FMMRCPS system has been developed in VB.NET. As project scheduling is popularly applied under various work environments of our society, the system is operated easily and friendly on standalone computers, or across a network, or any operating systems.

Table 12.2 lists the features of the system. The system allows a choice to deal with either single or multiple modes of project scheduling. In terms of activity duration times, the system provides options for dealing with all fuzzy or all crisp times or times containing both forms. To solve FMMRCPS problems of practical sizes effectively, the system is equipped with four metaheuristic approaches: (a) fuzzy GA, (b) fuzzy GA with tabu, (c) fuzzy SA, and (d) fuzzy SA with tabu.

Table 12.2 The feature of the FMMRCPS system

| Executive mode(s) | Activity duration times involved | Approaches | Functions |
|---|---|---|---|
| Single<br>Multiple | All crisp duration<br>All fuzzy duration<br>Crisp & fuzzy duration | Fuzzy GA<br>Fuzzy GA with tabu<br>Fuzzy SA<br>Fuzzy SA with tabu | (a) generating projects manually or automatically<br>(b) implementing four meta-heuristic approaches<br>(c) testing parameter settings for individual approaches<br>(d) producing a scheduling result with details of all activities<br>(e) experimenting with statistical analysis<br>(f) generating a report |

The FMMRCPS system has user-friendly interfaces from data input to result output. The functions of the system are listed in Table 12.2, and are briefly described below:

(a) For inputting project data, the system has a facility for generating a single, or a set of projects either manually or randomly by the system at the user's request.

(b) The system provides several flexible options for applying any of the four metaheuristic approaches in the user's preference interface.

(c) The system has a facility for conducting experiments on parameter settings under different conditions in order to make general recommendations on appropriate parameter settings prior to implementing an approach.

(d) The system provides detailed information on the best schedule result, including mode assignment to activities at each fuzzy scheduled time, the start and finish times, and the resource requirements of each activity.

(e) The system can examine the performance behaviour of an individual approach and carry out a comparative analysis of the approaches implemented, and provide statistical information on experimental performance.

(f) The system can produce a report for either a single schedule result with detailed information on activities or analysis details of the experimental result with statistical information.

The FMMRCPS system is a useful tool for efficiently and effectively solving any practically sized project scheduling problems where the activity duration times are fuzzy and activities can be performed in one of several executive modes under precedence relationships and resource constraints.

## 12.4 Implication of Empirical Studies

Empirical studies of six optimisation approaches developed have been conducted by a practical application and experimental analysis to demonstrate the characteristics and applicability of the approaches in solving realistic FMMRCPS problems. These studies reveal their differences in solution results and their suitability in different applications. In the following, the characteristics and implication of each approach is summarised.

- The fuzzy hybrid goal programming approach in incorporation with a rule knowledge base has been applied to a real project of dredge breakdown repair, and a satisfactory result has been obtained with detailed schedule information. The knowledge base allows the addition, removal or change of mode conditions for an activity, in respect to the resource requirements, at any time when the project environments are changed. This approach is robust, and has a quick response to changing circumstances during executing the project. This approach is particularly suitable for projects with frequent changes in conditions such as mode, resource requirements and activity duration times.

- The practical case study of dredge overhaul scheduling shows that this approach is a fast and simple way of providing a reasonable schedule result. This approach suits the situations where the DM needs an immediate schedule result so that the company can smoothly arrange a project beforehand without interrupting other ongoing projects of the company in an environment where resources are limited.

- Fuzzy GA is indicated as an efficient and fast way of solving FMMRCPS in terms of computational time among the four metaheuristic approaches as experiment conducted. This fuzzy GA approach can be applied in a situation where a company or an organisation needs immediately to know a good globally schedule result in terms of minimising the fuzzy project completion time. However, this approach is somewhat complex in operation, requiring the DM to know the proper parameter settings which are not required in the fuzzy heuristic approach. Familiarity with parameter settings can be gained through training and experience in the application of the approach in practice.

- The fuzzy GA with tabu approach performs better than fuzzy GA alone in terms of obtaining an approximate globally optimal solution, as reported in the experiment. However, the fuzzy GA with tabu approach requires significantly more time for computation than fuzzy GA alone. The fuzzy GA with tabu approach can be applied in a situation where a good globally optimal solution is required and the time required to generate a schedule result is less important.

- The fuzzy SA approach performs better than the fuzzy GA approach, but it is worse than the fuzzy GA with tabu approach in solution results, as indicated in the comparative analysis. The experiments reported have also indicated that the fuzzy SA approach takes somewhat more computational time than fuzzy GA and much less time than fuzzy GA with tabu. To effectively use the fuzzy SA approach, it is important to set the cooling ratio and Markov Chain length appropriately. If the user is familiar with the parameter settings, the fuzzy SA approach is the best approach for gaining a good globally optimal solution where the computational time is critical and a company or organisation needs to know the result as soon as possible.

- The fuzzy SA with tabu approach is the best approach of the four metaheuristic approaches in solution results, as shown clearly in experiments. The experiments also reports that this approach takes much more computational time than any of the other three metaheuristic approaches. If a company requires a very good globally optimal solution without an urgent need for the schedule result, the fuzzy SA with tabu approach is the best option to choose.

## 12.5 Contributions of the Research

This research has developed six optimisation approaches for efficiently and effectively solving complex FMMRCPS problems with single or multiple objectives. The outcome of the research has significant meanings in methodological development and in practical applications for handling FMMRCPS with fuzzy activity duration times. The contributions of this research are shown in Figure 12.1 and summarised as follows:

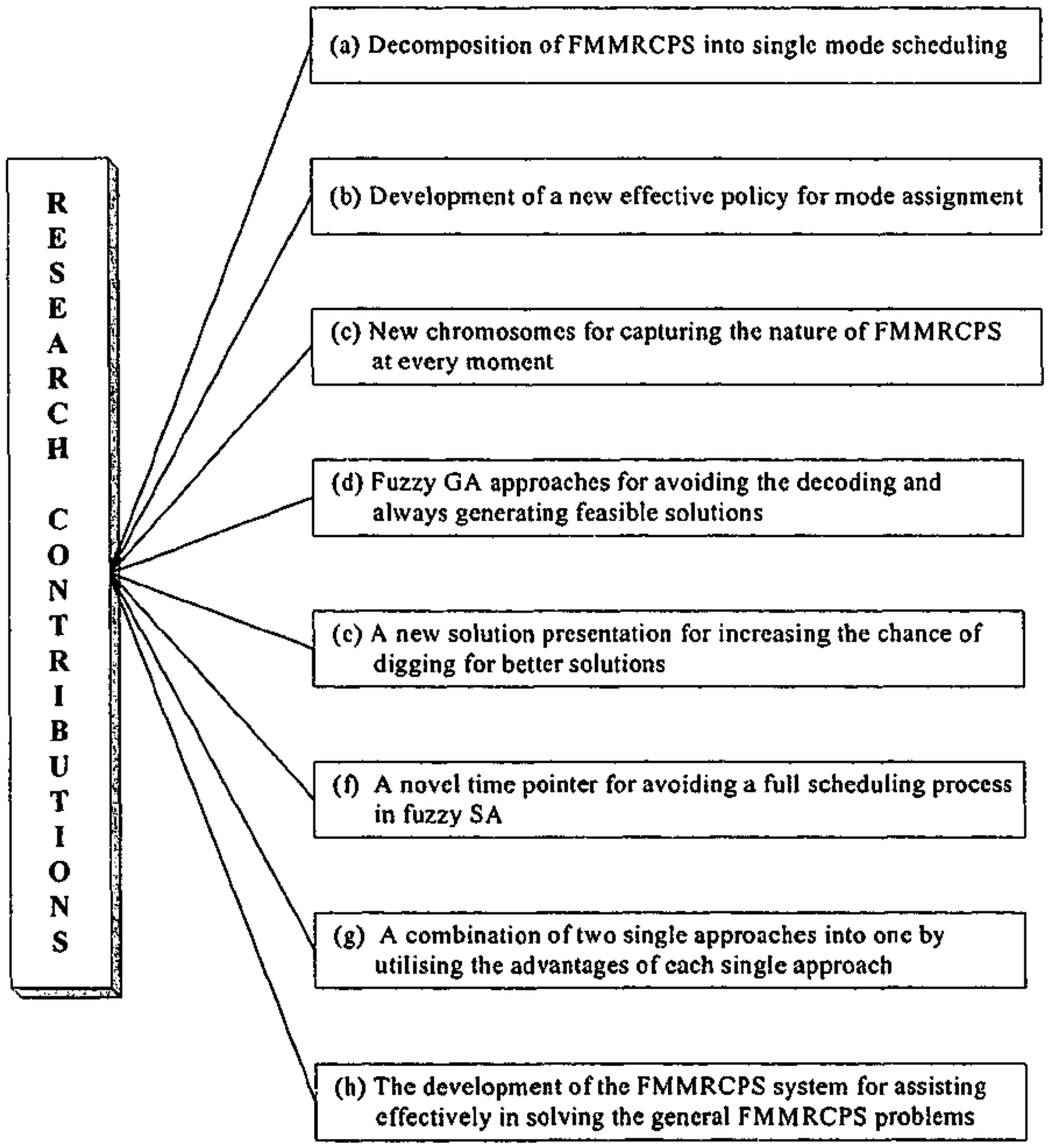

RESEARCH CONTRIBUTIONS
(a) Decomposition of FMMRCPS into single mode scheduling
(b) Development of a new effective policy for mode assignment
(c) New chromosomes for capturing the nature of FMMRCPS at every moment
(d) Fuzzy GA approaches for avoiding the decoding and always generating feasible solutions
(e) A new solution presentation for increasing the chance of digging for better solutions
(f) A novel time pointer for avoiding a full scheduling process in fuzzy SA
(g) A combination of two single approaches into one by utilising the advantages of each single approach
(h) The development of the FMMRCPS system for assisting effectively in solving the general FMMRCPS problems

Figure 12.1 Contributions of the research

**(a) Development of a novel technique to decompose a complex FMMRCPS problem into a simple single mode scheduling problem**

An FMMRCPS problem is a complex scheduling problem that requires solving two problems simultaneously: mode assignment and the sequence of activities. To simplify the complexity of FMMRCPS and to reduce the burden of constraints, a knowledge base is incorporated in a fuzzy goal programming approach so that the FMMRCPS problem is decomposed into a single mode project scheduling problem, thus improving the computational efficiency tremendously. This idea is enlightened on how to reduce some complex problems into simple problems that can be solved simply, rather than using more complicated methods.

**(b) Development of an effective policy for mode assignment**

Mode assignment is an important decision in FMMRCPS, because different modes assigned to activities can significantly affect the results of the schedule. A newly developed policy for mode assignment concerns both the current resource availabilities and the activities on critical path(s) at each scheduled time. The principle of the policy is to ensure that modes assigned to activities can be completed as soon as possible and at the same time, the completion times on each path are as close to each other as possible under available resources. This policy helps assigning an appropriate mode to an activity effectively at each scheduling stage.

**(c) Design of problem-specific chromosomes for FMMRCPS to capture the nature of complex scheduling at every moment.**

A chromosome is a key element in GA. The chromosome structure significantly influences the performance of fuzzy GA. To reflect the nature of FMMRCPS, the chromosome is composed of three sub-chromosomes: (a) the 1st sub-chromosome responds to mode assignment, (b) the 2nd sub-chromosome represents activity priorities, and (c) the 3rd sub-chromosome contains detailed scheduling information. The problem-specific chromosome is effectively designed to gain insights into the nature of every scheduling operation in order to gain a good globally optimal solution efficiently and effectively.

**(d) Development of fuzzy GA based approaches to avoid the decoding procedure for chromosomes and to always generate feasible solutions**

The fuzzy GA based approaches avoid the requirement for decoding chromosomes that is commonly required in GA. In addition, the approaches always generate feasible schedules. Thus, the novel fuzzy GA based approaches are computationally efficient for solving complex FMMRCPS problems.

**(e) Development of specifically designed solution representation in fuzzy SA to increase the possibility of digging for better solutions.**

To reflect the nature of FMMRCPS without distortion, the specially designed solution representation contains four elements: (a) mode assignments in each stage, (b) activity priority values in each stage, (c) all stages of partial schedules, and (d) the objective function. The design of this solution representation allows effective perturbation by both modes and priorities. The designed solution presentation is able to examine the status of partial scheduling at a scheduled time so as to guide the search for better solutions during neighbourhood generations.

**(f) Development of a novel technique, using a time pointer in solving fuzzy SA to avoid a full scheduling run each time.**

The time pointer is designed to point to a specific scheduled time in order to produce neighbourhoods by perturbing modes and priorities only in the indicated time zone without rescheduling each search. This novel technique reduces the computational time significantly.

(g) **Combination of different approaches into one by utilising the advantages of each single approach.**

In my PhD studies, two fuzzy hybrid approaches, (a) fuzzy GA with tabu, and (b) fuzzy SA with tabu, are developed by combining two different single approaches into one. These hybrid approaches take the advantages of each single approach to improve solution results that either single approach is unable to achieve. The experiments conducted have shown that the two fuzzy hybrid approaches perform better than the fuzzy GA alone or the fuzzy SA alone.

(h) **Development of the FMMRCPS system to assist efficiently in solving practically sized FMMRCPS problems**

In this research, a system for solving FMMRCPS has been developed by incorporating four metaheuristic approaches. The system can efficiently handle single or multiple project scheduling.

## 12.6 Future Research in FMMRCPS

The FMMRCPS model defined in my PhD research has the significance of practical value, being applicable to most practical situations. However, research publications dealing with fuzziness are scarce. More vigorous research on this model needs to be undertaken to tackle project scheduling that has practical significance. The majority of research on project scheduling still deals with either deterministic or probabilistic situations, which rarely exist in the real world. The study conducted here represents a significant and practical achievement for solving realistically sized FMMRCPS problems in a fuzzy environment. Research areas worth exploring further are briefly described below:

(a) In my PhD research, I have developed five heuristic and metaheuristic approaches for solving the single objective of minimising the fuzzy project completion time because this objective is of most common concern to industry and organisations. However, in some situations, FMMRCPS may need to take several objectives into consideration simultaneously to meet specific requirements. Further research is needed to solve multiple objectives based on the four metaheuristic approaches developed in my PhD study.

(b) More realistic approaches need to be developed for solving FMMRCPS problems of practical sizes. Approximation algorithms seem to be a good way of solving such NP-hard problems where exact algorithms are not viable. The multi-agent system technique is claimed as one of the more promising ways for effectively solving NP-hard problems with intelligent agents to effectively guide the search in a dynamic environment. This approach may be suitable for FMMRCPS where the scheduling environment is changed frequently at each scheduling stage as a result of changing organisational requirements or resource availabilities.

(c) To efficiently and effectively solve realistic FMMRCPS problems, advanced approaches also need to be developed that combine several algorithms into one approach, utilising the advantages of individual algorithms for strong performance in problem solving.

(d) In the current fuzzy project scheduling research, resource availabilities and resource requirements available to activities are assumed to be certain. However, in some situations, resource availabilities and resources requirements can be uncertain because some resources may be taken by other projects that involve fuzzy times or because resource requirements for a certain activity have to be described as linguistic terms due to conditions of resources. For example, considering for the resource of electricians: some electricians have much experience in a certain job and others may have little knowledge of the job. Therefore, the resource requirements for the same job can be different. Such situations should be taken into consideration when developing approaches for solving fuzzy resource-constrained project scheduling.

# References

Aarts, E. and J. Korst (1990). *Simulated Annealing and Boltzmann and Machines – A Stochastic Approach to Combinatorial Optimization and Neural Computing.* John Wiley & Sons, Chichester.

Abeyasinghe, M. C. L., D. J. Greenwood and D. E Johansen (2001). An efficient method for scheduling construction projects with resource constraints. *International Journal of Project Management* 19(1): 29-45.

Adamo, J. M. (1980). Fuzzy decision tree. *Fuzzy Sets and Systems* 4(3): 207-220.

Agin, N. (1966). Optimum seeking with branch-and-bound. *Management Science* 13(4): 176-185.

Ahn, T. and S. S. Erenguc (1998). The resource constrained project scheduling problem with multiple crashable modes: A heuristic procedure. *European Journal of Operational Research* 107(2): 250-259.

Akpan, E. O. P. (2000). Priority rules in project scheduling: a case for random activity selection. *Production Planning & Control* 11(2): 165-170.

Alcaraz, J and C. Maroto (2001). A robust genetic algorithm for resource allocation in project scheduling. *Annals of Operations Research* 102: 83-109.

Alcaraz, J and C. Maroto and R. Ruiz (2003). Solving the multi-mode resource-constrained project scheduling problem with genetic algorithms. *Journal of the Operational Research Society* 54(6): 614-626.

Alvarez-Valdés, R. and J. M. Tamarit (1989). Heuristic algorithms for resource-constrained project scheduling: a review and empirical analysis. In *Advances in Project Scheduling*, Eds. R. Slowinski and J. Weglarz. Elsevier, Amsterdam. 113-134.

Alvarez-Valdés, R. and J. M. Tamarit (1993). The project scheduling polyhedron: Dimension, facets and lifting theorems. *European Journal of Operational Research* 67(2): 204-220.

Amor, J.-P. and C. J. Teplitz (1998). An efficient approximation for project composite learning curves. *Project Management Journal* 29(3): 28-42.

Anthonisse, J. M., K. M. V. Hee and J. K. Lenstra (1988). Resource-constrained project scheduling: An international exercise in DSS development. *Decision Support Systems* 4(2): 249-257.

Aquilano, N. J. and D. E. Smith (1980). A formal set of algorithms for project scheduling with critical path scheduling/material requirements planning. *Journal of Operations Management* 1(2): 57-67.

Arinze, B. and F. Y. Partovi (1992). A knowledge-based decision support system for project management. *Computers & Operations Research* 19(5): 321-334.

Arora, R. K. and R. K. Sachdeva (1989). Distributed simulation of resource constrained project scheduling. *Computers & Operations Research* 16(4): 295-304.

Ash, R. C. (1999). Activity scheduling in the dynamic, multi-project setting: Choosing heuristics through deterministic simulation. *1999 Winter Simulation Conference Proceedings*, Phoenix, AZ, USA.

Baar, T., P. Brucker and S. Knust (1998). Tabu-search algorithms for the resource-constrained project scheduling problem. In S. Voss, S. Martello, I. Osman and C. Roucairol (Eds.), *Meta-Heuristics: Advances and Trends in Local Search Paradigms for Optimization*, Kluwer, Dordrecht. 1-18.

Baker, K. R. (1974). Sequencing with due-dates and early start times to minimize maximum tardiness. *Naval Research Logistics Quarterly* 21(1): 171-176.

Bandelloni, M., M. Tucci and R. Rinaldi (1994). Optimal resource levelling using non-serial dynamic programming. *European Journal of Operational Research* 78: 162-177.

Baptiste, P. and O. Grunder (1999). Cognitive science and project scheduling: more realistic representation. *Project Scheduling: Recent Models, Algorithms, and Applications*. J. Weglarz (Ed). Boston, Kluwer Academic Publishers: 451-475.

Baptiste, P. and C. Le Pape (1999). Satisfiability tests and time-bound adjustments for cumulative scheduling problems. *Annals of Operations Research* 92: 305-333.

Baroum, S. M. and J. H. Patterson (1996). The development of cash flow weight procedures for maximizing the net present value of a project. *Journal of Operations Management* 14(3): 209-227.

Baroum, S. M. and J. H. Patterson (1999). An exact solution procedure for maximizing the net present value of cash flows in a network. *Project Scheduling: Recent Models, Algorithms and Applications*. J. Weglarz. Boston, Kluwer Academic Publishers: 107-134.

Bautista, J. and J. Pereira (2002). Ant colonies for the RCPS Problem. In M. T. Escrig, F. Toledo and E. Golobardes (Eds.), *Topics in Artificial Intelligence*, 5th Catalonian Conference on AI (CCIA 2002), Springer-Verlag, Berlin. 257-268.

Bell, C. E. and K. Park (1990). Solving resource-constrained project scheduling problems by A* search. *Naval Research Logistics* 37(1): 61-84.

Bey, R. B. , R. H. Doersch and J. H. Patterson (1981). The net present value criterion: Its impact on project scheduling. *Project Management Quarterly* 12: 35-45.

Bianco, L., M. Caramia and P. Dell'Olmo (1999). Solving a preemptive project scheduling problem with coloring techniques. *Project Scheduling: Recent Models, Algorithms and Applications.* (Eds) J. Weglarz. Boston, Kluwer Academic Publisher: 135-145.

Bianco, L., P. Dell'Olmo and M. G. Speranza (1998). Heuristics for multimode scheduling problems with dedicated resources. *European Journal of Operational Research* 107(2): 260-271.

Blazewicz, J., Lenstra, K and Rinnooy Kan, A. H. G. (1983). Scheduling subject to resource constraints: Classification and complexity. *Discrete Applied Mathematics* 5: 11-24.

Blum, C and A. Roli (2003). Metaheuristics in combinational optimisation: Overview and conceptual comparison. *ACM Computing Survey* 35(3): 268-308.

Boctor, F. F. (1990). Some efficient multi-heuristic procedures for resource-constrained project scheduling. *European Journal of Operational Research* 49(1): 3-13.

Boctor, F. F. (1993). Heuristics for scheduling projects with resource restrictions and several resource-duration modes. *International Journal of Production Research* 31(11): 2547-2558.

Boctor, F. F. (1996a). A new and efficient heuristic for scheduling projects with resource restrictions and multiple execution modes. *European Journal of Operational Research* 90(2): 349-361.

Boctor, F. F. (1996b). Resource-constrained project scheduling by simulated annealing. *International Journal of Production Research* 34(8): 2335-2351.

Böttcher, J., A. Drexl, R. Kolisch and F. Salewski (1999). Project scheduling under partially renewable resource constraints. *Management Science* 45(4): 543-559.

Bouleimen, K. and H. Lecocq (2003). A new efficient simulated annealing algorithm for the resource-constrained project scheduling problem and its multiple mode version. *European Journal of Operational Research* 149: 268-281.

Bowers, M. R., K. Groom and R. Morris (1996). A practical application of a multi-project scheduling heuristic. *Production and Inventory Management Journal* 37(4): 19-25.

Brinkmann, K and Neumann K. (1996). Heuristic procedures for resource-constrained project scheduling with minimal and maximal time lags: The minimum-project duration and resource levelling problem. *Journal of Decision Systems* 5: 129-156.

Britney, R. R. (1976). Bayesian point estimation and the PERT scheduling of stochastic activities. *Management Science* 22(9): 938-948.

Brook, G. H. and C. R. White (1965). An algorithm for finding optimal or near optimal solutions to the production scheduling problem, *Journal of Industrial Engineering* Jan-Feb: 34-40.

Brucker, P., A. Drexl, R. Möhring, K. Neumann and E. Pesch (1999). Resource-constrained project scheduling: Notation, classification, models, and methods. *European Journal of Operational Research* 112(1): 3-41.

Brucker, P. and S. Knust (1999). Solving large-sized resource-constraing project scheduling problems. Project Scheduling: Recent Models, Algorithms and Applications. J. Weglarz (Ed). Boston, Kluwer Academic publishers: 27-51.

Brucker, P. and S. Knust (2000). A linear programming and constraint propagation-based lower bound for the RCPSP. *European Journal of Operational Research* 127(2): 355-362.

Brucker, P. and S. Knust (2003). Lower bound for resource-constrained project scheduling problems. *European Journal of Operational Research* 149: 302-313.

Brucker, P., S. Knust, A. Schoo and O. Thiele (1998). A branch and bound algorithm for the resource-constrained project scheduling problem. *European Journal of Operational Research* 107(2): 272-288.

Carlier, J. and E. Néron (2000). A new LP-based lower bound for the cumulative scheduling. *European Journal of Operational Research* 127(2): 363-382.

Carlier, J. and E. Pinson (1998). Jackson's pseudo preemptive schedule for the Pm/ri,qi/Cmax scheduling problem. *Annals of Operations Research* 83: 41-58.

Carruthers, J. A. and A. Battersby (1966). Advances in critical path methods. *Operational Research Quarterly* 17: 359-380.

Chan, F. T. S. (1997). Resource management in project scheduling through simulation. *International Journal of Computer Applications in Technology* 10(1/2): 81-89.

Chanas, S. and J. Kamburowski (1981). The use of fuzzy variables in PERT. *Fuzzy Sets and Systems* 5(1): 11-19.

Chang, T. C. and C. W. Ibbs (1989). A fuzzy expert system for priority ranking in network resource allocation. *Artificial Intelligence Techniques and Applications* *for Civil and Structural Engineers*. B. H. V. Topping. Edingurgh (Ed.), Civil-Comp Press: 101-107.

Chen, Y.-L., D. Rinks, et al. (1997). Critical path in an activity network with time constraints. *European Journal of Operational Research* 100(1): 122-133.

Cheng, C-H. (1998). A new approach for ranking fuzzy numbers by distance method. *Fuzzy Sets and Systems* 95: 307-317.

Cheng, R. and M. Gen (1994). Evolution program for resource constrained project scheduling problem. *Proceedings of The 1st IEEE Conference on Evolutionary Computation: IEEE World Congress on Computational Intelligence* Part Vol 2: 736-741.

Cheng, R. and M. Gen (1998). An evolution programme for the resource-constrained project scheduling problem. *International Journal of Computer Integrated Manufacturing* 11(3): 274-287.

Cho, J.-H. and Y.-D. Kim (1997). A simulated annealing algorithm for resource constrained project scheduling problems. *Journal of the Operational Research Society* 48(7): 736-744.

Christofides, N., R. Alvarez-Valdes and J. M. Tamarit (1987). Project scheduling with resource constraints: A branch and bound approach. *European Journal of Operational Research* 29(3): 262-273.

Ciobanu, G. (1981). Characteristics of feasible solutions for project scheduling problem. *Economic Computation & Economic Cybernetics Studies & Research* 1: 53-62.

Coffin, M. A. and B. W. Taylor III (1996). R&D project selection and scheduling with a filtered beam search approach. *IIE Transactions* 28(2): 167-176.

Cooper, D. F. (1976). Heuristics for scheduling resource-constrained projects: an experimental investigation. *Management Science* 22(11): 1186-1194.

Cooper, D. F. (1977). A note on serial and parallel heuristics for resource-constrained project scheduling. *Foundations of Control Engineering* 2(3): 131-134.

Cornell, G and J. Morrison (2002), *Programming VB.NET: A Guide for Experienced Programmers*, APress, Boston, USA.

Corner, J. L., L. R. Foulds, and K. Neumann (1997). Heuristics and applications for resource-constrained project scheduling with minimal and maximal time lags. *1997 IEEE International Conference on Intelligent Engineering Systems Proceedings*, Budapest, Hungary, IEEE. 403-408.

Crandall, K. C. (1973). Project Planning with precedence lead/lag factors. *Project Management Quarterly* 4(3): 18-27.

Czyzak, P. and A. Jaszkiewicz (1998). Pareto simulated annealing - A metaheuristic technique for multiple-objective combinatorial optimization. *Journal of Multi-criteria Decison Analysis* 7: 34-47.

Davies, E (1973). An experimental investigation of resource allocation in multiactivity projects. *Operational Research Quarterly* 24: 587-591.

Davis, E. W. and G. E. Heidorn (1971). An algorithm for optimal project scheduling under multiple resource constraints. *Management Science* 17(12): B803-B816.

Davis, E. W. and J. H. Patterson (1975). A comparison of heuristic and optimum solutions in resource-constrained project scheduling. *Management Science* 21(8): 944-955.

Davis, K. R., A. Stam, and R. Grzybowski. (1992). Resource constrained project scheduling with multiple objectives: A decision support approach. *Computers & Operations Research* 19(7): 657-669.

Davis, W. J. and S. M. West (1987). An Integrated approach to stochastic decisionmaking: a project scheduling example. *IEEE Transactions on Systems, Man, and Cybernetics* SMC-17(2): 199-209.

Dayanand, N. and R. Padman (1999). On payment schedules in contractor client negotiations in projects: an overview of the problem and research issues. *Project Scheduling: Recent Models, Algorithms, and Applications*. J. Weglarz (Ed). Boston, Kluwer Academic Publishers: 477-508.

De, P., E. J. Dunne, J. B. Ghosh and C. E. Wells. (1995). The discrete time-cost tradeoff problem revisited. *European Journal of Operational Research* 81(2): 225-238.

De, P., E. J. Dunne, J. B. Gosh and C. E. Wells (1997). Complexity of the discrete time-cost tradeoff problem for project networks, *Operations Research* 81: 225-238.

De Reyck, B., E. Demeulemeester and W. Herroelen (1998). Local search methods for the discrete time/resource trade-off problem in project networks. *Naval Research Logistics* 45(6): 553-578.

De Reyck, B., E. Demeulemeester and W. Herroelen (1999). Algorithms for scheduling project with generalized precedence relations. *Project Scheduling: Recent Models, Algorithms and Applications*. J. Weglarz (Ed). Boston, Kluwer Academic Publishers: 77-105.

De Reyck, B. and W. Herroelen (1997). Assembly line balancing by resource-constrained project scheduling techniques - A critical appraisal. *Foundations of Computing and Decision Sciences* 22(3): 143-167.

De Reyck, B. and W. Herroelen (1998a). A branch-and -bound procedure for the resource-constrained project scheduling problem with generalized precedence relations. *European Journal of Operational Research* 111(1): 152-174.

De Reyck, B. and W. Herroelen (1998b). An optimal procedure for the resource-constrained project scheduling problem with discounted cash flows and generalized precedence relations. *Computers & Operations Research* 25(1): 1-17.

De Reyck, B. and W. Herroelen (1999). The multi-mode resource-constrained project scheduling problem with generalized precedence relations. *European Journal of Operational Research* 119(2): 538-556.

Deckro, R. F. and J. E. Hebert (1990). A multiple objective programming framework for tradeoffs in project scheduling. *Engineering Costs and Production Economics* 18(3): 255-264.

Demeulemeester, E. L. (1995). Minimizing resource-availability costs in time-limited project networks. *Management Science* 41(10): 1590-1598.

Demeulemeester, E. L and W. S. Herroelen (1992). A branch-and-bound procedure for the multiple resource-constrained project scheduling problem. *Management Science* 38: 1803-1818.

Demeulemeester, E. L. and W. S. Herroelen (1996). An efficient optimal solution procedure for the preemptive resource-constrained project scheduling problem. *European Journal of Operational Research* 90(2): 334-348.

Demeulemeester, E. L. and W. S. Herroelen (1997a). New benchmark results for the resource-constrained project scheduling problem. *Management Science* 43(11): 1485-1492.

Demeulemeester, E. L. and W. S. Herroelen (1997b). A branch-and-bound procedure for the generalized resource-constrained project scheduling problem. *Operations Research* 45(2): 201-212.

Demeulemeester, E. L and W. S. Herroelen (2002). *Project Scheduling – A Research Handbook.* Kluwer Academic Publishers. Boston.

Demeulemeester, E. L., W. S. Herroelen and S. E. Elmaghraby (1996). Optimal procedures for the discrete time/cost trade-off problem in project networks. *European Journal of Operational Research* 88(1): 50-68.

Deng, H., C.-H. Yeh and H Pan (1999). A knowledge-based approach for criteria weighting in multi-criteria analysis. *Computational Intelligence for Modelling, Control & Automation*. M. Mohammadian (Ed.). Ohmsha, IOS Press: 435-440.

DePorter, E. L. and K. P. Ellis (1990). Optimization of project networks with goal programming and fuzzy linear programming. *Computers & Industrial Engineering* 19(1-4): 500-504.

Devroye, L (1979). Inequalities for the completion times of stochastic PERT networks. *Mathematics of Operations Research* 4: 441-447

Dimsdale, B. (1963). Computer construction of minimal project networks. *IBM Systems Journal*, March Issue: 24-36.

Dodin, B. (1999). Project management in audit staff scheduling. *Project Scheduling: Recent Models, Algorithms, and Applications*. J. Weglarz. Boston (Ed.). Kluwer Academic Publishers: 509-527.

Doersch, R. H. and J. H. Patterson (1977). Scheduling a project to maximize its present value: A zero-one programming approach. *Management Science* 23(8): 882-889.

Dorigo, M., and Gambardella M. (1997a). Ant colonies for the travelling salesman problem. *BioSystems* 43: 73-81.

Dorigo, M., and Gambardella M. (1997b). Ant colony system: A cooperative learning approach to the travelling salesman problem. *IEEE Transactions on Evolutionary Computation* 1(1): 53-66.

Dorigo, M., Manniezzo V. and Colorni A. (1996). The ant system: optimization by a colony of cooperating agents. *IEEE Transactions on Systems, Man, and Cybernetics – Part B* 26(1): 29-41.

Dorigo, M. and Di Caro, G. (1999a). The ant colony optimization meta-heuristic. In D. Corne, M. Dorigo and F. Glover (Eds). *New Ideas in Optimization*. McGraw-Hill, Maidenhead, UK.

Dorigo, M. and Di Caro, G. (1999b). Ant algorithm for discrete optimization. *Artificial Life* 5(2): 137-172.

Dorndorf, U., T. P. Huy and E. Pesch (1999). A survey of interval capacity consistency tests for time-and resource-constrained scheduling. *Project Scheduling: Recent Models, Algorithms, and Applications*. J. Weglarz (Ed). Boston, Kluwer Academic Publishers: 213-238.

Dorndorf, U., E. Pesch and T. Phan-Huy (2000). A time-oriented branch-and-bound algorithm for resource-constrained project scheduling with generalised precedence constraints. *Management Science* 46(10): 1365-1384.

Drexl, A. (1991). Scheduling of project network by job assignment. *Management Science* 37: 1590-1602.

Drexl, A. and J. Gruenewald (1993). Nonpreemptive multi-mode resource-constrained project scheduling. *IIE Transactions* 25(5): 74-81.

Drexl, A., J. Juretzka, F. Salewski and A. Schirmer (1999). New modelling concepts and their impact on resource-constrained project scheduling. *Project Scheduling: Recent Models, Algorithms, and Applications*. J. Weglarz (Ed). Boston, Kluwer Academic Publishers: 413-432.

Drexl, A., R. Nissen, J. H. Patterson and F. Salewski. (2000). ProGen/px - An instance generator for resource-constrained project scheduling problems with partial renewable resources and further extensions. *European Journal of Operational Research* 125(1): 59-72.

Dubois, D., Ostasiewicz, W. and H. Prade (2000). Fuzzy sets: history and basic notions. *Fundamentals of Fuzzy Sets*, Part 1: 21-106.

Dubois, D. and H. Prade (1980). *Fuzzy Sets and Systems: Theory and Applications*. Acedemic Press, New York.

Dubois, D. and H. Prade (1988). *Possibility Theory: An Approach to Computerized Processing of Uncertainty*. Plenum Press, New York.

Šeda, M. (1997). A comparison of exact methods and genetic algorithm approach to resource constrained scheduling. *Proceedings of the 3NWGA*, Helsinki, Finland 97-108.

Elmaghraby, S. E. (1977). *Activity Networks: Project Planning and Control by Network Models*. Wiley, Chichester.

Elmaghraby, S. E. (1990). Project bidding under deterministic and probabilistic activity durations. *European Journal of Operational Research* 49(1): 14-34.

Elmaghraby, S. E. (1993). Resource allocation via dynamic programming in activity networks. *European Journal of Operational Research* 64(2): 199-215.

Elmaghraby, S. E. (2000). On criticality and sensitivity in activity networks. *European Journal of Operational Research* 127(2): 220-238.

Elmaghraby, S. E. and W. S. Herroelen (1990). The scheduling of activities to maximize the net present value of projects. *European Journal of Operational Research* 49(1): 35-49.

Elmaghraby, S. E. and Kamburowsky, J. (1990). On project representations and activity floats. *The Arabian Journal for Science and Engineering* 14: 627-637.

Elsayed, E. A. (1982). Algorithmes for project scheduling with resource constraints. *Interational Journal of Production Research* 20(1) : 95-103.

Elsayed, E. A. and N. Z. Nasr (1986). Heuristics for resource-constraints project scheduling. *International Journal of Production Research* 24 : 299-310.

Erenguc, S. S. and O. I. Tukel (1999). Integrating quality as a measure of performance in resource-constrained project scheduling problems. *Project Scheduling: Recent Models, Algorithms, and Applications*. J. Weglarz (Ed). Boston, Kluwer Academic Publishers: 433-450.

Fargier, H., V. Galvagnon and D. Dubois (2000). Fuzzy PERT in series-parallel graphs. *9th IEEE International Conference on Fuzzy Systems* 2:717-722.

Fernandez, A. A., R. L. Armacost and J. J. Pet-Edwards (1996). The Role of the nonaticipativity conatraint in commercial software for stochastic project schieduling. *Computers & Industrial Engineering* 31(1/2): 233-236.

Fernandez, A. A., R. L. Armacost and J. J. Pet-Edwards (1998). Understanding simulation solutions to resource constrained project scheduling problems with stochastic task durations. *Engineering Management Journal* 10(4): 5-13.

Fetz, T., M. Oberguggenberger (1999). Fuzzy models in geotechnical engineering and construction management. *Computer-Aided Civil and Infrastructure Engineering* 14(2): 93-106.

Fortemps, P. and M. Hapke (1997). On the disjunctive graph for project scheduling. *Foundations of Computing and Decision Sciences* 22(3): 195-209.

Fu, C.-C. and H.-F. Wang (1995). Fuzzy project scheduling models under inflation condition. *Proceedings of the International Joint Conference on CFSA/IFIS/SOFT'95 on Fuzzy Theory and Applications,* Taipei, Taiwan, World Scientific, Singapore 362-367.

Galambos, J (1995). *Advanced Probability Theory,* 2nd Edition, New York : M. Dekker.

Garavelli, A. C. and P. Pontrandolfo (1995). A heuristic method for the estimation of the project duration in a stochastic network scheduling. *Operations Research* 29(3): 285-298.

Gaul, W. (1981). Bounds for the expected duration of a stochastic project planning model. *Journal of Information & Optimization Sciences* 2(1): 45-63.

Gemmill, D. D. and Y.-W. Tsai (1997). Using a simulated annealing algorithm to schedule activities of resource-constrained projects. *Project Management Journal* 28(4): 8-20.

Glover, F. (1989). Tabu search - part I, *ORSA Journal on Computing* 1(3):190-206.

Glover, F. (1990). Tabu search - part II, *ORSA Journal on Computing* 2(1):4-32.

Golenko-Ginzburg, D. and A. Gonik (1997). Stochastic network project scheduling with non-consumable limited resources. *International Journal of Production Economics* 48(1): 29-37.

Golenko-Ginzburg, D. and A. Gonik (1998). A heuristic for network project scheduling with random activity durations depending on the resource allocation. *International Journal of Production Economics* 55(2): 149-162.

Goulter, I. C. and Ramlogan, R. N. (1987). A mixed integer approach to resource leveling. *Journal of Information & Optimization Sciences* 8(1): 101-111.

Grinold, R. C. (1972). The payment scheduling problem. *Naval Research Logistics Quarterly* 19: 123-136.

Gupta, M. M. (1977). Fuzzyism, the first decade. In Gupta, M. M., Saridis, G. N. and Gaines, B. R. (eds.). *Fuzzy Automata and Decision Process.* North-Holland, Amsterdam 5-10.

Gutjahr, W. J., C. Strauss and E. Wagner (2000). A stochastic branch-and bound approach to activity crashing in project management. *INFORMS Journal of Computing* 12(2): 125-135.

Hall, L. A. (1997). Approximation algorithms for scheduling. In D. S. Hochbaum (Ed.) *Approximation Algorithms for NP-Hard Problems*, PWS Publishing, USA 1-45.

Hapke, M., A. Jaszkiewicz and R. Slowinski (1994). Fuzzy project scheduling system for software development. *Fuzzy Sets and Systems* 67: 101-117.

Hapke, M., A. Jaszkiewicz and R. Slowinski (1998). Interactive analysis of multiple-criteria project scheduling problems. *European Journal of Operational Research* 107(2): 315-324.

Hapke, M., A. Jaszkiewicz and R. Slowinski (1999). Fuzzy multi-mode resource-constrained project scheduling with multiple objectives. *Project Scheduling: Recent Models, Algorithms, and Applications.* J. Weglarz (Eds.). Boston, Kluwer Academic Publishers: 355-381.

Hapke, M., A. Jaszkiewicz and R. Slowinski. (2000). Pareto simulated annealing for fuzzy multi-objective combinatorial optimization. *Journal of Heuristics* 6(3): 329-345.

Hapke, M. and R. Slowinski (1993). A DSS for resource-constrained project scheduling under uncertainty. *Journal of Decision Systems* 2(2): 111-128.

Hapke, M. and R. Slowinski (1996). Fuzzy priority heuristics for project scheduling. *Fuzzy Sets and Systems* 83: 291-299.

Harris, R. B. (1990). Packing method for resource levelling (PACK). *Journal of Construction Engineering and Management* 116: 331-350.

Harrison, S. R. and R. H. U. Tamaschke (1993). *Statistics for Business, Economics and Management*, Prentice Hall, NSW.

Hartmann, S. (1998). A competitive genetic algorithm for resource-constrained project scheduling. *Naval Research Logistics* 45(7): 733-750.

Hartmann, S. (1999). *Project Scheduling under Limited Resources – Models, Methods and Applications*, Lecture Notes in Economics and Mathematical Systems 478. Springer, Germany.

Hartmann, S. (2000). Packing problems and project scheduling models: an integrating perspective. *Journal of the Operational Research Society* 51(9): 1083-1092.

Hartmann, S. (2001). Project scheduling with multiple modes: a genetic algorithm. *Annals of Operations Research* 102: 111-135.

Hartmann, S. and A. Drexl (1998). Project scheduling with multiple modes: A comparison of exact algorithms. *Networks* 32(4): 283-297.

Hartmann, S. and R. Kolisch (2000). Experimental evaluation of state-of-the-art heuristics for the resource-constrained project scheduling problem. *European Journal of Operational Research* 127(2): 394-407.

Hartmann, S. and A. Sprecher (1996). A note on "hierarchical models for multi-project planning and scheduling". *European Journal of Operational Research* 94(2): 377-383.

Haugen, K. K. (1996). A Stochastic Dynamic Programming model for scheduling of offshore petroleum fields with resource uncertainty. *European Journal of Operational Research* 88(1): 88-100.

Heilmann, R. (2003). A branch-and-bound procedure for the multi-mode resource-constrained project scheduling problem with minimum and maximum time lags. *European Journal of Operational Research* 144(2): 348-365.

Herroelen, W. and B. De Reyck (1998). Resource-constrained project scheduling: A survey of recent developments. *Computers & Operations Research* 25(4): 279-302.

Herroelen, W. and E. Demeulemeester (1999). A classification scheme for project scheduling. Project Scheduling: *Recent Models, Algorithms and Applications*. J. Weglarz (Ed). Boston, Kluwer Academic Publishers 1-26.

Herroelen, W. S., P. Van Dommelen and E. L. Demeulemeester (1997). Project network models with discounted cash flows a guided tour through recent developments. *European Journal of Operational Research* 100(1): 97-121.

Herroelen, W. S. and E. Gallens (1993). Computational experience with an optimal procedure for the scheduling of activities to maximize the net present value of projects. *European Journal of Operational Research* 65: 274-277.

Hindi, K. S., H. Yang and K. Fleszar (2002). An evolutionary algorithm for resource-constrained project scheduling. *IEEE Transactions on Evolutionary Computation* 6(5): 512-518.

Hobbs, B. F. and P. M. Meier (1994). Multicriteria methods for resource planning: An experimental comparison. *IEEE Transactions on Power Systems* 9(4): 1811-1817.

Holland, H. J. (1975). *Adaptation in Natural and Artificial Systems*. University of Michigan Press, Ann Arbor.

Holloway, C. A., R. T. Nelson and V. Suraphongschai (1979). Comparison of a multi-pass heuristic decomposition procedure with other resource-constrained project scheduling procedures. *Management Science* 25(9): 862-872.

Hopefield, J. J. (1982) Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the National Academy of Sciences of the United States of America* 79(8): 2554-2558.

Hopefield, J. J. and D. W. Tank (1985). "Neural" computation of decisions in optimization problems. *Biological Cybernetics* 52(3): 141-152.

Icmeli, O. (1993). Project scheduling problems: A survey. *International Journal of Operations & Production Management* 13(11): 80-91.

Icmeli, O. and S. S. Erenguc (1994). A tabu search procedure for the resource constrained project scheduling problem with discounted cash flows. *Computers & Operations Research* 21(8): 841-853.

Icmeli, O. and S. S. Erenguc (1996a). The resource constrained time/cost tradeoff project scheduling problem with discounted cash flows. *Journal of Operations Management* 14(3): 255-275.

Icmeli, O. and S. S. Erenguc (1996b). A branch and bound procedure for the resource constrained project scheduling problem with discounted cash flows. *Management Science* 42(10): 1395-1408.

Icmeli, O. and W. O. Rom (1996). Solving the resource constrained project scheduling problem with optimization subroutine library. *Computers & Operations Research* 23(8): 801-817.

Icmeli-Tukel, O. and W. O. Rom (1997). Ensuring quality in resource constrained project scheduling. *European Journal of Operational Research* 103(3): 483-496.

Igelmund, G. and F. J. Radermacher (1983a). Preselective strategies for the optimization of stochastic project networks under resource constraints. *Networks* 13: 1-28.

Igelmund, G. and F. J. Radermacher (1983b). Algorithmic approaches to preselective strategies for stochastic scheduling problems. *Networks* 13: 29-48.

Ikeuchi, T., Y. Ikkai, et al. (1998). Project scheduling using a genetic algorithm with adaptable changing genetic operators. *Electrical Engineering in Japan* 124(2): 36-42.

Ishibuchi, H., N. Yamamoto, T. Murata and H. Tanaka (1994). Genetic algorithms and neighhood search algorithms for fuzzy flowshop scheduling problems. *Fuzzy Sets and Systems* 67: 81-100.

Ito, T. (1998). A neural network approach to project scheduling. *Bulletin of Faculty of Engineering, the University of Tokushima* 43: 73-79.

Johnson, R. V. (1988). Efficient modular implementation of branch-and-bound algorithms. *Decision Sciences* 19: 17-38.

Jörnsten, K. and R. Leisten (1995). Linear programming aggregation: A heuristic for hierarchical production planning. *Asia-Pacific Journal of Operational Research* 12(2): 161-177.

Józefowska, J., M. Mika, R. Rózycki G. Waligóra and J. Weglarz. (1999). Project scheduling under discrete and continuous resources. *Project Scheduling: Recent Models, Algorithms, and Applications.* J. Weglarz (Ed). Boston, Kluwer Academic Publishers: 289-307.

Józefowska, J., M. Mika, R. Rózycki, G. Waligóra and J. Weglarz (2001). Simulated annealing for multi-mode resource-constrained project scheduling. *Annals of Operations Research* 102: 137-155.

Józefowska, J., M. Mika, G. Waligóra and J. Weglarz (2002). Multi-mode resource-constrained project scheduling problem with discounted positive cash flows – Simulated annealing vs. tabu search. *Foundations of Computing & Decision Sciences* 27(2): 97-114.

Kaefer, F. (2000). A note on "Project scheduling with resource constraints: A branch and bound approach." *European Journal of Operational Research* 125(1): 216-217.

Karshenas, S. and D. Haber (1990). Economic optimization of construction project scheduling. *Construction Management and Economics* 8(2): 135-146.

Kazaz, B. and C. Sepil (1996). Project scheduling with discounted cash flows and progress payments. *Journal of the Operational Research Society* 47(10): 1262-1272.

Kelley, J. E. (1961). Critical-path planning and scheduling: Mathematical basis. *Operational Research* 9: 296-320.

Kelley, J. E. (1963). The critical-path method: Resources planning and scheduling. *Industrial Scheduling*. J. F. Muth, G. L. Thompson and P. R. Winters (Eds). Englewood Cliffs, New Jersey, Prentice Hall 347-365.

Khamooshi, H. (1998). (DP)2SM: A dynamic approach to resource constraint project scheduling. *Computers & Industrial Engineering* 35(3-4): 507-510.

Khamooshi, H. (1999). Dynamic Priority-Dynamic Programming Scheduling Method (DP)2SM: a dynamic approach to resource constraint project scheduling. *International Journal of Project Management* 17(6): 383-391.

Khattab, M. and F. Choobineh (1990). A new heuristic for project scheduling with a single resource constraint. *Computers & Industrial Engineering* 19(1-4): 514-518.

Khattab, M. and F. Choobineh (1991). A new heuristic for project scheduling with a single resource constraint. *Computers & Industrial Engineering* 20(3): 381-387.

Khattab, M. and F. Choobineh (1992). A basis for the design of a multiattribute heuristic for single resource project scheduling. *Computers & Industrial Engineering* 22(2): 157-161.

Kim, S. O. and M. J. Schniederjans (1989). Heuristic framework for the resource constrained multi-project scheduling problem. *Computers & Operations Research* 16(6): 541-556.

Kim, S.-Y. and R. C. Leachman (1993). Multi-project scheduling with explicit lateness costs. *IIE Transactions* 25(2): 34-44.

Kirkpatrick, Jr. S., Gelatt, C. and Vecchi, M. (1983). Optimisation by simulated annealing. *Science* 220: 671-680.

Klein, R. (2000a). Bidirectional planning: Improving priority rule-based heuristics for scheduling resource-constrained projects. *European Journal of Operational Research* 127(3): 619-638.

Klein, R. (2000b). Project scheduling with time-varying resource constraints. *International Journal of Production Research* 38(16): 3937-3952.

Klein, R (2000c). *Scheduling of Resource-Constrained Project*. Kluwer Academic Publishers, Boston.

Klein, R. and A. Scholl (1999). Computing lower bounds by destructive improvement: An application to resource-constrained project scheduling. *European Journal of Operational Research* 112(2): 322-346.

Klir, G. J. and Yuan, B. (1995). *Fuzzy Sets and Fuzzy Logic: Theory and Applications.* Prentice Hall PTR, New Jersey.

Knotts, G., M. Dror, B. C. Hartman (2000). Agent-based project scheduling. *IIE Transactions* 32(5): 387-401.

Kogan, K. (1998). Scheduling under common due date, a single resource and precedence constraints - a dynamic approach. *Discrete Event Dynamic Systems: Theory and Applications* 8(4): 353-364.

Kogan, K. and A. Shtub (1999). Scheduling projects with variable-intensity activities: The case of dynamic earliness and tardiness costs. *European Journal of Operational Research* 118(1): 65-80.

Kolisch, R. (1995). *Project Scheduling under Resource Constraints-Efficient Heuristics for Several Problem Classes.* Heidelberg, Physica-Verlag.

Kolisch, R. (1996a). Efficient priority rules for the resource-constrained project scheduling problem. *Journal of Operations Management* 14(3): 179-192.

Kolisch, R. (1996b). Serial and parallel resource-constrained project scheduling methods revisited: Theory and computation. *European Journal of Operational Research* 90(2): 320-333.

Kolisch, R. and A. Drexl (1996). Adaptive search for solving hard project scheduling problems. *Naval Research Logistics* 43(1): 23-40.

Kolisch, R. and A. Drexl (1997). Local search for nonpreemptive multi-mode resource-constrained project scheduling. IIE *Transactions* 29(11): 987-999.

Kolisch, R. and S. Hartmann (1999). Heuristic algorithms for the resource-constrained project scheduling problem: classification and computational analysis. *Project Scheduling: Recent Models, Algorithms, and Applications.* Boston, Kluwer Academic Publishers 147-178.

Kolisch, R. and R. Padman (2001). An integrated survey of deterministic project scheduling. *Omega* 29: 249-272.

Kolisch, R., C. Schwindt and A. Sprecher (1999). Benchmark instances for project scheduling problems. *Project Scheduling: Recent Models, Algorithms, and Applications.* J. Weglarz (Ed). Boston, Kluwer Academic Publishers 197-212.

Kolisch, R. and A. Sprecher (1995). Characterization and generation of a general class of resource-constrained project scheduling problems. *Management Science* 41(10): 1693-1703.

Kramer, B. A. and C.-L. Hwang (1991). Resource constrained project scheduling: modelling with multiple alternatives. *Mathematical and Computer Modelling* 15(8): 49-63.

Kurtulus, I. and E. W. Davis (1982). Multi-project scheduling: Categorization of heuristic rules performance. *Management Science* 28(2): 161-172.

Kurtulus, I. S. and S. C. Narula (1985). Multi-project scheduling: Analysis of project performance. *IIE Transactions* 17(1): 58-66.

Lacksonen, T. A. and C.-Y. Hung (1998). Project scheduling algorithms for re-layout projects. *IIE Transactions* 30(1): 91-99.

Lambourn, S. (1963). Resource allocation and multi-project scheduling (RAMPS) - a new tool in planning and control. *The Computer Journal* 5: 300-304.

Lawrence, S. R. and T. E. Morton (1993). Resource-constrained multi-project scheduling with tardy costs: Comparing myopic, bottleneck, and resource pricing heuristics. *European Journal of Operational Research* 64(2): 168-187.

Leachman, R. C. and S. Kim (1993). A revised critical path method for networks including both overlap relationships and variable-duration activities. *European Journal of Operational Research* 64(2): 229-248.

Lee, J.-K. and Y.-D. Kim (1996). Search heuristics for resource constrained project scheduling. *Journal of the Operational Research Society* 47(5): 678-689.

Lee-Kwang, H. and J. Favrel (1988). The SSD graph: A tool for project scheduling and visualization. *IEEE Transactions on Engineering Management* 35(1): 25-30.

Leon, V. J. and R. Balakrishnan (1995). Strength and adaptability of problem-space based neighbourhoods for resource-constrained scheduling. *OR Spektrum* 17: 173-182.

Leu, S.-S., A.-T. Chen and C.-H. Yang (2001). A GA-based fuzzy optimal model for construction time-cost trade-off. *International Journal of Project Management* 19(1): 47-58.

Levner, E. V. and A. S. Nemirovsky (1994). A network flow algorithm for just-in-time project scheduling. *European Journal of Operational Research* 79(2): 167-175.

Li, K. Y. and R. J. Willis (1992). An iterative scheduling technique for resource-constrained project scheduling. *European Journal of Operational Research* 56(3): 370-379.

Li, R. K.-Y. and R. J. Willis (1991). Alternative resources in project scheduling. *Computers & Operations Research* 18(8): 663-668.

Li, R. J. and Lee, E. S. (1987). Ranking fuzzy numbers – a comparison. *Proceedings of the North American Fuzzy Information Processing Society Workshop* (NAFIPS'87).

Lin, C. T. and Lee, C. S. G (1995). *Neural Fuzzy Systems: A Neuro-Fuzzy Synergism to Intelligent Systems*. Prentice Hall, N.J, USA

Liou, T. and Wang, J. (1992). Ranking fuzzy numbers with integral value. *Fuzzy Sets and Systems* 54: 247-255.

Liu, S. and M. Wang (2000). An object-oriented methodology for solving the RCPSPs with heuristics and metaheuristics. *Production Planning & Control* 11(5): 434-442.

López, O. C., R. M. Barcia, O. Eyada and F. O. Gauthier (1996). An evolutionary algorithm for resource-constrained project scheduling and multiple execution modes. In D. L. Borges and C. A. A. Kaestner (Eds). *Advances in Artificial Intelligence - 13th Brazilian Symposium on Artificial Intelligence*, Brazil, Springer-Verlag, Berlin, Germany 101-110.

Lorterapong, P. (1994). A fuzzy heuristic method for resource-constrained project scheduling. *Project Management Journal* 25(4): 12-18.

Lova, A., C. Maroto and P. Tormos (2000). A multicriteria heuristic method to improve resource allocation in multiproject scheduling. *European Journal of Operational Research* 127(2): 408-424.

Maniezzo, V. and A. Mingozzi (1999). A heuristic procedure for the multi-mode project scheduling problem based on benders' decomposition. *Project Scheduling: Recent Models, Algorithms, and Applications*. J. Weglarz (Editor). Boston, Kluwer Academic Publishers 179-196.

Maniezzo, V. and A. Mingozzi (1999). The project scheduling problem with irregular starting time costs. *Operations Research Letter* 25(4): 175-182.

Maroto, C., P. Tormos and A. Lova (1999). The evolution of software quality in project scheduling. *Project Scheduling: Recent Models, Algorithms, and Applications*. J. Weglarz (Ed). Boston, Kluwer Academic Publishers 239-259.

Mason, A. T. and C. L. Moodie (1971). A branch and bound algorithm for minimizing cost in project scheduling. *Management Science* 18(4): B158-B173.

McCahon, C. S. (1993). Using PERT as an Approximation of fuzzy project-network analysis. *IEEE Transactions on Engineering Management* 40(2): 146-153.

McCahon, C. S. and E. S. Lee (1988). Project network analysis with fuzzy activity times. *Computers & Mathematics with Applications* 15(10): 829-838.

Merke, D., M. Middendorf and H. Schemeck (2002). Ant colony optimization for resource-constrained project scheduling. *IEEE Transactions on Evolutionary Computation* 6(4): 333-346.

Metropolis, N., Rosenbluth, A, Rosenbluth, M., Teller, A. & Teller, E. (1953). Equation of state calculation by fast computing machines. *Journal of Chemical Physics* 21: 1087-1092.

Mingozzi, A., V. Maniezzo, S. Ricciardelli and L. Bianco. (1998). An exact algorithm for the resource-constrained project scheduling problem based on a new mathematical formulation. *Management Science* 44(5): 714-729.

Möhring, R. H. (1984). Minimizing costs of resource requirements in project networks subject to a fixed completion time. *Operations Research* 32: 89-120.

Moccellin, J. V. (1989). A two-stage approach to resource-constrained multi-project scheduling problems. *Policy and Information* 13(1): 1-18.

Mohring, R. H., A. S. Schulz, F. Stork and M. Uetz (1999). Resource-constrained project scheduling: Computing lower bounds by solving minimum cut problems. Lecture Notes in Computer Science 1643: 139-150.

Moizuddin, M. and S. Z. Selim (1997). Project scheduling under limited resources. *Proceedings of 1997 AACE International Transactions of the Annual Meeting* P&S.04.1-P&S.04.7.

Mori, M. and C.-C. Tseng (1997). A resource constrained project scheduling problem with reattempt at failure: A heuristic approach. *Journal of the Operations Research Society of Japan* 40(1): 33-44.

Mori, M. and C. C. Tseng (1997). A genetic algorithm for multi-mode resource constrained project scheduling problem. *European Journal of Operational Research* 100(1): 134-141.

Morse, L. C., J. O. McIntosh and G. E. Whitehouse (1996). Using combinations of heuristics to schedule activities of constrained multiple resource projects. *Project Management Journal* 27(1): 34-40.

Motwani, R., J. Naor and P. Raghavan (1997). Randomized approximation algorithms in combinatorial optimization. In D. S. Hochbaum (Ed.) *Approximation Algorithms for NP-Hard Problems*, PWS Publishing, USA 447-481.

Nabrzyski, J. and J. Weglarz (1995). On an expert system with tabu search for multiobjective project scheduling. *The Proceedings of 1995 INRIA / IEEE Symposium on Emerging Technologies and Factory Automation*, Los Alamitos, CA, USA, IEEE Comput Soc. Press 3: 87-94.

Nabrzyski, J. and J. Weglarz (1999). Knowledge-based multiobjective project schduling problems. In J. Weglarz (Ed.), *Project Scheduling: Recent Models, Algorithms, and Applications*. Kluwer Academic Publishers, Boston 383-411.

Naphade, K. S., S. D. Wu and R. H. Storer. (1997). Problem space search algorithms for resource-constrained project scheduling. *Annals of Operations Research* 70: 307-326.

Natsuaki, Y., H. Furuta, K. Takase, T. Iemura, K. Muto and H. Ninomiya. (1993). Application of fuzzy set theory to project scheduling of bridge construction. *The Proceedings of 2nd International Symposium on Uncertainty Modeling and Analysis.* College Park, MD, USA, IEEE 514-519.

Nazareth, T., S. Verma, S. Bhattacharya and A. Bagchi (1999). The multiple resource constrained project scheduling problem: A breadth-first approach. European *Journal of Operational Research* 112(2): 347-366.

Negnevitsky, M. (2002). *Artificial Intelligence: A Guide to Intelligent Systems.* Addison Wesley, Harlow, England.

Negoita, C. V. (2000). *Fuzzy Sets.* New Falcon Publications, Tempe, AZ.

Neumann, K. (1999). Scheduling of projects with stochastic evolution structure. *Project Scheduling: Recent Models, Algorithms, and Applications.* J. Weglarz (Ed). Boston, Kluwer Academic Publishers: 309-332.

Neumann, K. and J. Zhan (1995). Heuristics for the minimum project-duration problem with minimal and maximal time lags under fixed resource constraints. *Journal of Intelligent Manufacturing* 6(2): 145-154.

Neumann, K. and J. Zimmermann (1999a). Methods for resource-constrained project scheduling with regular and nonregular objective functions and schedule-dependent time windows. *Project Scheduling: Recent Models, Algorithms, and Applications.* J. Weglarz (Ed). Boston, Kluwer Academic Publishers 261-287.

Neumann, K. and Zimmermann, J. (1999b). Resource levelling for projects with schedule-dependent time windows. *European Journal of Operational Research* 117(3): 591-605.

Neumann, K. and J. Zimmermann (2000). Procedures for resource leveling and net present value problems in project scheduling with general temporal and resource constraints. *European Journal of Operational Research* 127(2): 425-443.

Nkasu, M. M. and K. H. Leung (1997). A resources scheduling decision support system for concurrent project management. *International Journal of Production Research* 35(11): 3107-3132.

Norbis, M. and J. M. Smith (1988). A multi-objective multi-level heuristic for dynamic resource-constrained scheduling problems. *European Journal of Operational Research* 33: 30-41.

Norbis, M. and J. M. Smith (1996). An interactive decision support system for the resource constrained scheduling problem. *European Journal of Operational Research* 94(1): 54-65.

Noronha, S. J. and V. V. S. Sarma (1991). Knowledge-based approaches for scheduling problems: a survey. *IEEE Transactions on Knowledge & Data Engineering* 3(2): 160-171.

Novák, V., Perfilieva, I, and Močkoř, J (1999). *Mathematical Principles of Fuzzy Logic*. Kluwer Academic Publisher, USA.

Nowicki, E. and C. Smutnicki (1994). A decision support system for the resource constrained project scheduling problem. *European Journal of Operational Research* 79(2): 183-195.

Ntuen, C. A. and E. H. Park (1995). An experiment in scheduling and planning of non-structured jobs: Lessons learned from artificial intelligence and operational research toolbox. *European Journal of Operational Research* 84(1): 96-115.

Nudtasomboon, N. and S. U. Randhawa (1997). Resource-constrained project scheduling with renewable and non-renewable resources and time-resource tradeoffs. *Computers & Industrial Engineering* 32(1): 227-242.

Oguz, O. and H. Bala (1994). A comparative study of computational procedures for the resource constrained project scheduling problem. *European Journal of Operational Research* 72(2): 406-416.

Ohmae, Y., S. Hasegawa, and M. Okuda (1992). Multi-project scheduling. *Journal of Information Processing* 15(2): 267-279.

Olaguíbel, R. A.-V. and J. M. T. Goerlich, Eds. (1989). *Heuristic algorithms for resource-constrained project scheduling: A review and an empirical analysis. Advances in Project Scheduling. Amsterdam,* The Netherlands, Elsevier Science Publishers.

Ovacik, I. M., S. Rajagopalan and R. Uzsoy (2000). Integrating interval estimates of global optima and local search methods for combinatorial optimization problems. *Journal of Heuristics* 6(4): 481-500.

Özdamar, L. (1999). A genetic algorithm approach to a general category project scheduling problem. *IEEE Transactions on systems, Man and Cybernetics - Part C: Applications and Reviews* 29(1): 44-59.

Özdamar, L. and E. Alanya (2001). Uncertainty modelling in software development project (with case study). *Annals of Operations Research* 102: 157-178.

Özdamar, L. and H. Dündar (1997). A flexible heuristic for a multi-mode capital constrained project scheduling problem with probabilistic cash inflows. *Computers & Operations Research* 24(12): 1187-1200.

Özdamar, L. and G. Ulusoy (1994). A local constraint based analysis approach to project scheduling under general resource constraints. *European Journal of Operational Research* 79(2): 287-298.

Özdamar, L. and G. Ulusoy (1995). A survey on the resource-constrained project scheduling problem. *IIE Transactions* 27(5): 574-586.

Özdamar, L. and G. Ulusoy (1996a). An iterative local constraint based analysis for solving the resource constrained project scheduling problem. *Journal of Operations Management* 14(3): 193-208.

Özdamar, L. and G. Ulusoy (1996b). A note on an iterative forward/backward scheduling technique with reference to a procedure by Li and Willis. *European Journal of Operational Research* 89(2): 400-407.

Padman, R. and D. E. Smith-Daniels (1993). Early-tardy cost trade-offs in resource constrained projects with cash flows: An optimization-guided heuristic approach. *European Journal of Operational Research* 64(2): 295-311.

Padman, R., D. E. Smith-Daniels and V. L. Smith-Daniels (1997). Heuristic scheduling of resource-constrained projects with cash flows. *Naval Research Logistics* 44(4): 365-381.

Pan, H. (1997). *A Multi-Criteria Decision Making Model for Dredger Dispatching under Uncertainty*. Master thesis. School of Business Systems, Monash University.

Pan, H., C.-H. Yeh and R. J. Willis (1998). A fuzzy goal programming model for purchasing dredgers under uncertainty, *Proceedings of the 3rd International Conference on Management*, Shanghai, China R115, 1-12.

Pan, H, C-H Yeh and R. J. Willis (2001a). Resource-constrained project scheduling with fuzziness. In N. Mastorakis (Ed). *Advances in Fuzzy Systems and Evolutionary Computation*, WSES Press 173-179.

Pan, H, C-H Yeh and R. J. Willis (2001b). Computer-aided system to solve uncertainty in project management. *Proceedings of the 10th IEEE International Conference on Fuzzy Systems* (*FUZZY-IEEE 2001*), Melbourne 3: 1376-1379.

Pan, H, C-H Yeh and R. J. Willis (2001c). A hybrid genetic algorithm for project scheduling. In C. E. D'Attellis, V. V. Kluev, N. E. Mastorakis (Eds), *Mathematics and Simulation with Biological, Economical and Musicoacoustical Applications*. WSES Press 40-44.

Pan, H., C.-H. Yeh and R. J. Willis (2002). Solving uncertainty in Project Scheduling, *Proceedings of the 19th Annual Conference of Management and International Association of Management* (*AoM/IAoM*), Quebec, Canada 198-205.

Pan, H. and C.-H. Yeh (2003a). Fuzzy project scheduling, *Proceedings of the 2003 IEEE International Conference on Fuzzy Systems* (*FUZZ-IEEE2003*), St. Louis, Missouri, USA, 755-760.

Pan, H. and C.-H. Yeh (2003b). A metaheuristic approach to fuzzy project scheduling. V. Palade, R.J. Howlett and L.C. Jain (Eds). *Knowledge-Based Intelligent Information and Engineering Systems,* University of Oxford, UK, Proceedings (KES 2003), Lecture Notes in Computer Science (Artificial Intelligence), Springer-Verlag, Berlin, Heidelberg 2773:1081-1087.

Pan, H. and C.-H. Yeh (2003c). GA for Fuzzy multi-mode resource-constrained project scheduling, *WSEAS Transactions on Systems – Special Issue on Fuzzy Sets and Fuzzy Systems*, 4(2), 983-990.

Pan, H. and C.-H. Yeh (2003d). Simulated annealing for multi-mode project scheduling, *WSEAS Transactions on Systems – Special Issue on Neural Networks and Applications*, 3(2), 681-687

Pannetier, J. (1990). Simulated annealing: an introductory review. *International Physics Conference*: 107: 23-44.

Patterson, J. H. (1973). Alternate methods of project scheduling with limited resources. *Naval Research Logistics Quarterly* 20(4): 767-784.

Patterson, J. H. (1976). Project scheduling: The effects of problem structure on heuristic performance. *Naval Research Logistics Quarterly* 23(1): 95-123.

Patterson, J. H. (1984). A comparison of exact approaches for solving the multiple constrained resource, project scheduling problem. *Management Science* 30(7): 854-867.

Patterson, J. H. and W. D. Huber (1974). A Horizon-varying, zero-one approach to project scheduling. *Management Science* 20(6): 990-998.

Patterson, J. H. and G. W. Roth (1976). Scheduling a project under multiple resource constraints: A zero-one programming approach. *AIIE Transactions* 8(4): 449-455.

Patterson, J. H, R. Slowinski, F. B. Talbot and J. Weglarz (1989). An algorithm for a general class of precedence and resource constrained scheduling problem. In Slowinski, R. and J. Weglarz (eds.). *Advances in Project Scheduling*.. Elsevier Science Publishers. Amsterdam 3-28.

Patterson, J. H., F. B. Talbot, R. Slowinski and J. Weglarz. (1990). Computational experience with a backtracking algorithm for solving a general class of precedence and resource-constrained scheduling problems. *European Journal of Operational Research* 49(1): 68-79.

Pesch, E. (1999). Lower bounds in different problem classes of project schedules with resource constraints. *Project Scheduling: Recent Models, Algorithms and Applications*. J. Weglarz (Ed). Boston, Kluwer Academic Publishers 53-76.

Pet-Edwards, J. (1995). Application of genetic algorithms in resource constrained network optimization. *The Proceedings of IEEE International Conference on Systems, Man & Cybernetics* 4: 3059-3061

Pham, D. T. and D. Karaboga (2000). *Intelligent optimisation techniques: genetic algorithms, tabu search, simulated annealing and neural networks*. Springer, New York.

Phillips Jr, S. (1996). Project management duration/resource tradeoff analysis: An application of the cut search approach. *Journal of the Operational Research Society* 47(5): 697-701.

Pinder, J. P. and A. S. Marucheck (1996). Using discounted cash flow heuristics to improve project net present value. *Journal of Operations Management* 14(3): 229-240.

Pinson, E., C. Prins and F. Rullier (1994). Using tabu search for solving the resource-constrained project scheduling problem. *Proceedings of the 4th International Workshop on Project Management and Scheduling*, Leuven, 102-106.

Pollack-Johnson, B. (1995). Hybrid structures and improving forecasting and scheduling in project management. *Journal of Operations Management* 12(2): 101-117.

Pontrandolfo, P. (2000). Project duration in stochastic networks by the PERT-path technique. *International Journal of Project Management* 18(3): 215-222.

Potts, C. N. and L. N. Van Wassenhove (1991). Single machine tardiness sequencing heuristics. *IIE Transactions* 23: 346-354.

Prade, H. (1979). Using fuzzy set theory in a scheduling problem: A case study. *Fuzzy Sets and Systems* 2(2): 153-165.

Premachandra, I. M. (2001). An approximation of the activity duration distribution in PERT. *Computers & Operations Research* 28(5): 443-452.

Pritsker, A. A. B., L. J. Watters, and P. M. Wolfe. (1969). Multiproject scheduling with limited resources: A zero-one programming approach. *Management Science* 16(1): 93-108.

Pulat, P. S. and S. J. Horn (1996). Time-Resource Tradeoff Problem Project Scheduling. *IEEE Transactions on Engineering Management* 43(4): 411-417.

Ramudhin, A., B. Montreuil and P. Lefrançois (1994). Scheduling project activities in a distributed environment. *European Journal of Operational Research* 78(2): 242-251.

Rangaswamy, B., A. S. Jain and F. Glover. (1997). Tabu search candidate list strategies in scheduling. *Interfaces in Computer Science and Operations Research: Advances in Metaheuristics, Optimization, and Stochastic Modeling Technologies*. R. S. Barr, R. V. Helgason and J. L. Kennington (Eds). Kluwer Academic 215-233.

Rommelfanger, H. J. (1994). Network analysis and information flow in fuzzy environment. *Fuzzy Sets and Systems* 67(1): 119-128.

Russell, A. H. (1970). Cash flows in networks. *Management Sciences* 16: 357-373.

Russell, R. A. (1986). A comparison of heuristics for scheduling projects with cash flows and resource restrictions. *Management Science* 32(10): 1291-1300.

Rutenbar, R. A. (1989). Simulated annealing algorithms: an overview. *IEEE Circuits and Devices Magazine* 5(1): 19-26.

Rys, T., R. Stanek, et al. (1994). MIPS: A DSS for multiobjective interactive project scheduling. *European Journal of Operational Research* 79(2): 196-207.

Salewski, F. and A. Schirmer (1997). Project scheduling under resource and mode identity constraints: Model, complexity, methods, and application. *European Journal of Operational Research* 102(1): 88-110.

Sampson, S. E. and E. N. Weiss (1993). Local search technique for the generalized project scheduling problem. *Naval Research Logistics* 40: 665-675.

Schirmer, A. (2000). Case-based reasoning and improved adaptive search for project scheduling. *Naval Research Logistics* 47(3): 201-222.

Shewchuk, J. P. and T. C. Chang (1995). Resource-constrained job scheduling with recyclable resources. *European Journal of Operational Research* 81(2): 364-375.

Shi, J. J. and Z. Deng (2000). Object-oriented resource-based planning method (ORPM) for construction. *International Journal of Project Management* 18(3): 179-188.

Shipley, M. F., A. de Korvin, et al. (1997). BIFPET methodology versus PERT in project management: fuzzy probability instead of the beta distribution. *Journal of Engineering and Management* 14: 49-65.

Shtub, A., L. J. LeBlanc, et al. (1996). Scheduling programs with repetitive projects: A comparison of a simulated annealing, a genetic and a pair-wise swap algorithm. *European Journal of Operational Research* 88(1): 124-138.

Shue, L.-Y. and R. Zamani (1999). An intelligent search method for project scheduling problems. *Journal of Intelligent Manufacturing* 10(3/4): 279-288.

Simpson III, W. P. and J. H. Patterson (1996). A multiple-tree search procedure for the resource-constrained project scheduling problem. *European Journal of Operational Research* 89(3): 525-542.

Slowinski, R. (1980). Two approaches to problems of resource allocation among project activities - A comparative study. *Journal of the Operational Research Society* 31: 711-723.

Slowinski, R. (1981). Multiobjective network scheduling with efficient use of renewable and nonrenewable resources. *European Journal of Operational Research* 7(3): 265-273.

Slowinski, R. (1984). Modelling and Solving multicriteria project scheduling problems. *Foundations of Control Engineering* 9(1): 43-52.

Slowinski, R. (1989). Multiobjective project scheduling under multiple-category resources. *Advances in Project Scheduling*. R. Slowinski (Ed). Amsterdam, Elsevier 151-167.

Slowinski, R., B. Soniewicki and J. Weglarz (1994). DSS for multiobjective project scheduling. *European Journal of Operational Research* 79(2): 220-229.

Slowinski, R. and J. Weglarz (1978). Solving the General Project Scheduling Problem with Multiple Constrained Resources by Mathematical Programming, *Lecture Notes in Control and Information Sciences*, Berlin: Springer-Verlag 7: 278-289.

Smith, D. R. (1984). Random trees and the analysis of branch and bound procedures. *Journal of the Association for Computing Machinery* 31(1): 163-188.

Smith-Daniels, D. E. and N. J. Aquilano (1984). Constrained resource project scheduling subject to material constraints. *Journal of Operations Management* 4(4): 369-387.

Smith-Daniels, D. E. and N. J. Aquilano (1987). Using a late-start resource-constrained project schedule to improve project net present value. *Decision Sciences* 18: 617-630.

Smith-Daniels, D. E., R. Padman and V. L. Smith-Daniels (1996). Heuristic scheduling of capital constrained projects. *Journal of Operations Management* 14(3): 241-254.

Smith-Daniels, D. E. and V. L. Smith-Daniels (1987). Maximizing the net present value of a project subject to materials and capital constraints. *Journal of Operations Management* 7(1/2): 33-45.

Speranza, M. G. and C. Vercellis (1993). Hierarchical models for multi-project planning and scheduling. *European Journal of Operational Research* 64(2): 312-325.

Sprecher, A. (1994). *Resource-constrained project scheduling: Exact methods for the multi-mode case*. Number 409 in Lecture Notes in Economics and Mathematical Systems. Pringer, Berlin.

Sprecher, A. (1999). A competitive branch-and bound algorithm for the simple assemply line balancing problem. *International Journal of Production Research* 37(8): 1787-1816.

Sprecher, A. (2000). Scheduling resource-constrained projects competitively at modest memory requirements. *Management Science* 46(5): 710-723.

Sprecher, A. and A. Drexl (1998). Multi-mode resource-constrained project scheduling by a simple, general and powerful sequencing algorithm. *European Journal of Operational Research* 107(2): 431-450.

Sprecher, A. and A. Drexl (1999). Note: On semi-active timetabling in resource-constrained project scheduling. *Management Science* 45(3): 452-454.

Sprecher, A., S. Hartmann, and A. Drexl. (1997). An exact algorithm for project scheduling with multiple modes. *OR Spektrum* 19(3): 195-203.

Sprecher, A., R. Kolisch and R. Drexl (1995). Semi-active, active, and non-delay schedules for the resource-constrained project scheduling problem. *European Journal of Operational Research* 80(1): 94-102.

Stinson, J. P., E. W. Davis and B. M. Khumawala (1978). Multiple resource-constrained scheduling using branch-and-bound. *AIIE Transactions* 10: 252-259.

Sunde, L. and S. Lichtenberg (1995). Net-present-value cost/time tradeoff. *International Journal of Project Management* 13(1): 45-49.

Sung, C. S. and S. K. Lim (1997). A scheduling procedure for a general class of resource-constrained projects. *Computers & Industrial Engineering* 32(1): 9-17.

Syslo, M. M. (1984). On the computational complexity of the minimum-dummy-activity problem in a PERT Network. *Networks* 14: 37-45.

Takamoto, M., Yamada, N., Kobayashi, Y. and Nonaka, H. (1995). Zero-one programming algorithm for resource levelling of manufacturing process schedules. *Systems and Computers in Japan* 26: 68-76.

Talbot, F. B. and J. H. Patterson (1978). An efficient integer programming algorithm with network cuts for solving resource-constrained scheduling problems. *Management Sciences* 24(11): 1163-1174.

Talbot, F. B. (1982). Resource-constrained project scheduling with time-resource tradeoffs: The nonpreemptive case. *Management Science* 28(10): 1197-1210.

Tam, P. W. M. and P. B. G. Dissanayake (1998). Construction project scheduling by ranked positional weight method. *Canadian Journal of Civil Engineering* 25(3): 424-436.

Tavares, L. V. (1990). A multi-stage non-deterministic model for project scheduling under resources constraints. *European Journal of Operational Research* 49(1): 92-101.

Tavares, L. V. and J. A. A. Ferreira (1998). On the optimal management of project risk. *European Journal of Operational Research* 107(2): 451-469.

Than, F. T. S. (1997). Resource management in project scheduling through simulation. *International Journal of Computer Applications in Technology* 10(1/2): 81-89.

Thesen, A. (1976). Heuristic scheduling of activities under resource and precedence restrictions. *Management Science* 23(4): 412-422.

Toklu, Y. C. (2002). Application of genetic algorithms to construction scheduling with or without resource constraints. *Canadian Journal of Civil Engineering* 29(3): 421-429.

Thomas, P. R. and S. Salhi (1997). An investigation into the relationship of heuristic performance with network-resource characteristics. *Journal of the Operational Research Society* 48(1): 34-43.

Thomas, P. R. and S. Salhi (1998). A tabu search approach for the resource constrained project scheduling problem. *Journal of Heuristics* 4(2): 123-139.

Thomas, W. (1969). Four float measures for critical path scheduling. *Journal of Industrial Engineering* 1(10): 19-23.

Tomasz, R., R. Stanek and W. Ziembla (1994). MIPS: A DSS for multiobjetive interactive project scheduling. *European Journal of Operational Research* 79(2): 196-207.

Tormos, P and A. Lova (2001). A competitive heuristic solution technique for resource-constrained project scheduling. *Annals of Operational Research* 102: 65-81.

Tsai, Y.-W. and D. D. Gemmill (1998). Using tabu search to schedule activities of stochastic resource-constrained projects. *European Journal of Operational Research* 111(1): 129-141.

Tsubakitani, S. and R. F. Deckro (1990). A heuristic for multi-project scheduling with limited resources in the housing industry. *European Journal of Operational Research* 49(1): 80-91.

Tukel, O. I. (1996). Scheduling resource-constrained project when non-conformities exist. *Project Management Journal* 27(3): 47-55.

Tukel, O. I. and W. O. Rom (1998). Analysis of the characteristics of projects in diverse industries. *Journal of Operations Management* 16(1): 43-61.

Tukel, O. I. and S. N. Wasti (2001). Analysis of supplier buyer relationships using resource constrained project scheduling strategies. *European Journal of Operational Research* 129(2): 271-276.

Ulusoy, G. and S. Cebelli (2000). An equitable approach to the payment scheduling problem in project management. *European Journal of Operational Research* 127(2): 262-278.

Ulusoy, G. and L. Özdamar (1989). Heuristic performance and network/resource characteristics in resource-constrained project scheduling. *Journal of the Operational Research Society* 40(12): 1145-1152.

Ulusoy, G. and L. Özdamar (1994). A constraint-based perspective in resource constrained project scheduling. *International Journal of Production Research* 32(3): 693-705.

Ulusoy, G. and L. Özdamar (1995). A heuristic scheduling algorithm for improving the duration and net present value of a project. *International Journal of Operations & Management* 15(1): 89-98.

Ulusoy, G. and L. Özdamar (1996). A framework for an interactive project scheduling system under limited resources. *European Journal of Operational Research* 90(2): 362-375.

Vaithyanathan, S. and J. P. Ignizio (1992). A stochastic neural network for resource constrained scheduling. *Computers & Operations Research* 19(3/4): 241-254.

Verhoeven, M. G. A. (1998). Tabu search for resource-constrained scheduling. *European Journal of Operational Research* 106(2/3): 266-276.

Viana, A. and J. P. de Sousa (2000). Using metaheuristics in multiobjective resource constrained project scheduling. *European Journal of Operational Research* 120(2): 359-374.

Wang, H.-F. and C.-C. Fu (1996). Fuzzy project scheduling models under inflation condition. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems* 4(6): 497-514.

Wang, J. R. (1999). A fuzzy set approach to activity scheduling for product development. *Journal of the Operational Research Society* 50: 1217-1228.

Weglarz, J. (1979). Project scheduling with discrete and continuous resources. *IEEE-Transactions on Systems, Man and Cybernetics* SMC-9(10): 644-650.

Weglarz, J. (1981). Project scheduling with constinuously-divisible, doubly constrained resources. *Management Science* 27(9): 1040-1053.

Weiss, E. N. (1988). An optimization based heuristic for scheduling parallel project networks with constrained renewable resources. *IIE Transactions* 20(2): 137-143.

Whitehouse, G. E. and J. R. Brown (1979). Genres: An extension of brooks algorithm for project scheduling with resource constraints. *Computers & Industrial Engineering* 3(3): 261-268.

Wiest, J. D. (1964). Some properties of schedules for large projects with limited resources. *Operations Research* 12(3): 395-418.

Wiest, J. D. (1967). A heuristic model for scheduling large project projects with limited resources. *Management Science* 13: 359-377.

Willis, R. J. (1985a). Critical path analysis and resource constrained project scheduling - Theory and practice. *European Journal of Operational Research* 21(2): 149-155.

Willis, R. J. (1985b). An algorithm for constructing project network diagrams on an ordinary line printer. *Computers & Operations Research* 12(2): 163-168.

Willis, R. J., H. Pan and C.-H. Yeh (1998). Resource-constrained project scheduling under uncertain activity duration. *Business Systems Research*, School of Business Systems, Monash University 12-27.

Willis, R. J., H. Pan and C.-H. Yeh (1999). Resource-constrained project scheduling under uncertain activity duration. In M. Mohammadian (Ed). *Computational Intelligence for Modelling, Control & Automation*. Ohmsha, IOS Press 429-434.

Wong, D. F. and C. L. Liu (1986). A new algorithm for floorplan design. *Proceedings of 23rd ACM/IEEE Design Automation Conference* 101-107.

Woodworth, B. M and Willie, C. J. (1975). A heuristic algorithm for resource levelling in multi-project, multi-resource scheduling. *Decision Sciences* 6: 525-540.

Yager, R. R. (1981). On a general class of fuzzy connectives. *Fuzzy Sets and Sysyems* 4: 235-242.

Yang, K.-K. (1998). A comparison of dispatching rules for executing a resource-constrained project with estimated activity durations. *Omega* 26(6): 729-738.

Yang, K. K., L. C. Tay and C. C. Sum (1995). A comparison of stochastic scheduling rules for maximization project net present value. *European Journal of Operational Research* 85: 327-339.

Yang, K. K., F. B. Talbot and J. H. Patterson. (1993). Scheduling a project to maximize its net present value: An integer programming approach. *European Journal of Operational Research* 64(2): 188-198.

Yao, X (1995). A new simulated annealing algorithm. *International Journal of Computer Mathematics* 56(3-4): 161-168.

Yau, C. and E. Ritchie (1990). Project compression's method for speeding up resource constrained projects which preserve the activity schedule. *European Journal of Operational Research* 49(1): 140-152.

Yeh, C.-H., H. Deng and H. Pan (1999a). Multi-criteria analysis for dredger dispatching under uncertainty, *Journal of the Operational Research Society* 50(1): 35-43.

Yeh, C-H., R. J. Willis, H. Deng and H. Pan (1999b). Task oriented weighting in multi-criteria analysis, *European Journal of Operational Research* 119(1): 130-146.

Yeh, C.-H., H. Pan and R. J. Willis (1999c). A heuristic approach to fuzzy resource-constrained project scheduling. In M. Mohammadian (Ed). *Computational Intelligence for Modelling, Control & Automation*, Ohmsha, IOS Press 423-428.

Yen, V. C. (1996). Building a project scheduling system: a prototyping experience. *Proceedings of 1996 IRMA International Conference* 408-410.

Younis, M. A. and B. Saad (1996). Optimal resource levelling of multi-resource projects. *Computers and Industrial Engineering* 31: 1-4.

Zadeh, L. A. (1965). Fuzzy sets. *Information and Control* 8: 338-353.

Zadeh, L. A. (1978) Fuzzy sets as a basis for a theory of possibility. *Fuzzy Sets and Systems* 1: 3-28.

Zamani, R. and L.-Y. Shue (1997). The application of an admissible heuristic algorithm to project scheduling problem. *International Journal of Computer Applications in Technology* 10(5/6): 308-315.

Zamani, R. and L.-Y. Shue (1998). Solving project scheduling problems with a heuristic learning algorithm. *Journal of the Operational Research Society* 49(7): 709-716.

Zhan, J. (1994). Heuristics for scheduling resource-constrained projects in MPM networks. *European Journal of Operational Research* 76(1): 192-205.

Zhang, P. (1998). An image construction method for visualizing managerial data. *Decision Support Systems* 23(4): 371-387.

Zhu, D. and R. Padman (1995). Neural networks for heuristic selection: an application in resource-constrained project scheduling. *The Impact of Emerging Technologies on Computer Science and Operations Research*. S. G. Nash and A. Sofer. Boston, Kluwer Academic Publishers 297-312.

Zhu, D. and R. Padman (1997). Connectionist approaches for solver selection in constrained project scheduling. *Annals of Operations Research* 72: 265-298.

Zhu, D. and R. Padman (1999). A metaheuristic scheduling procedure for resource-constrained projects with cash flows. *Naval Research Logistics* 46(8): 912-926.

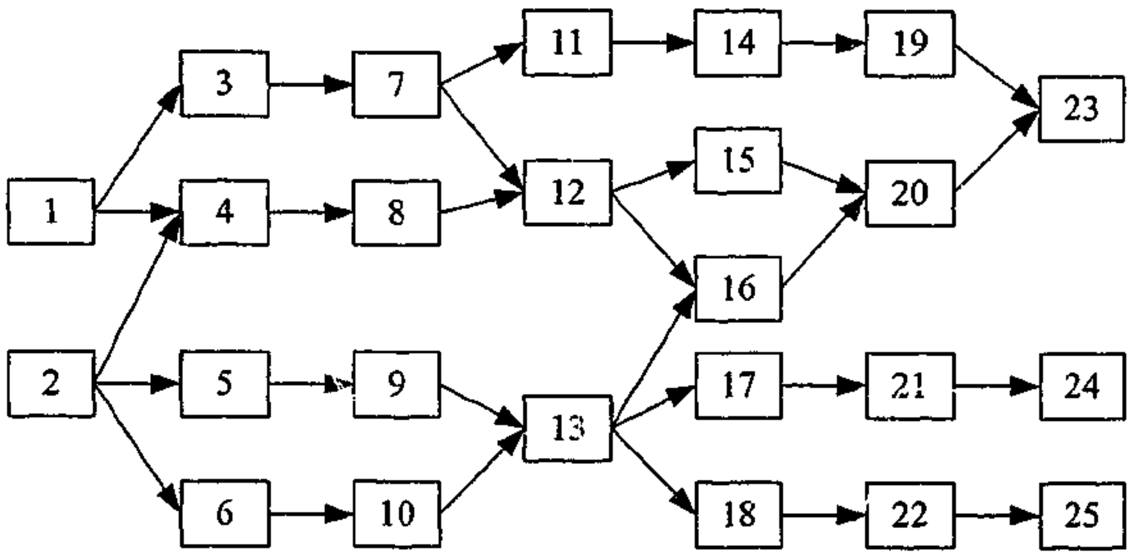
Zimmermann, H. J. (1996). *Fuzzy Set Theory and Its Applications.* Kluwer Academic Publisher, Boston.

Zimmermann, H. J. (1987). *Fuzzy Sets, Decision Making, and Expert Systems.* Kluwer Academic Publisher, Boston.

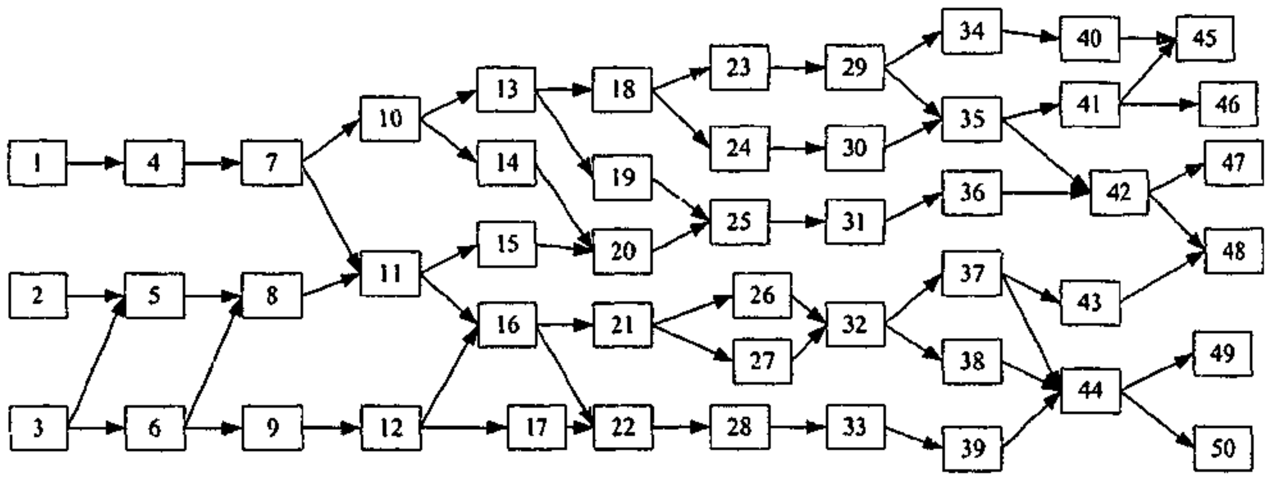
# Appendix A

# Five Project Network Diagrams

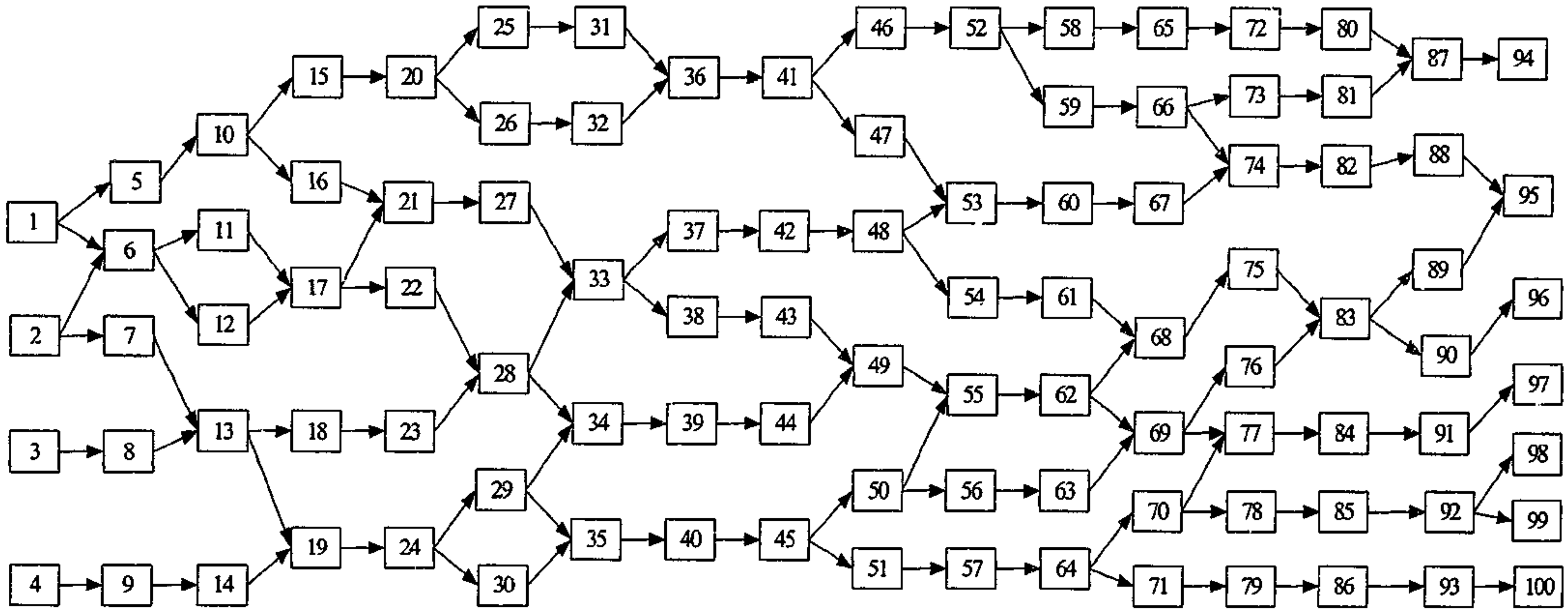
(A) Network Diagram of a Project with 25 activities



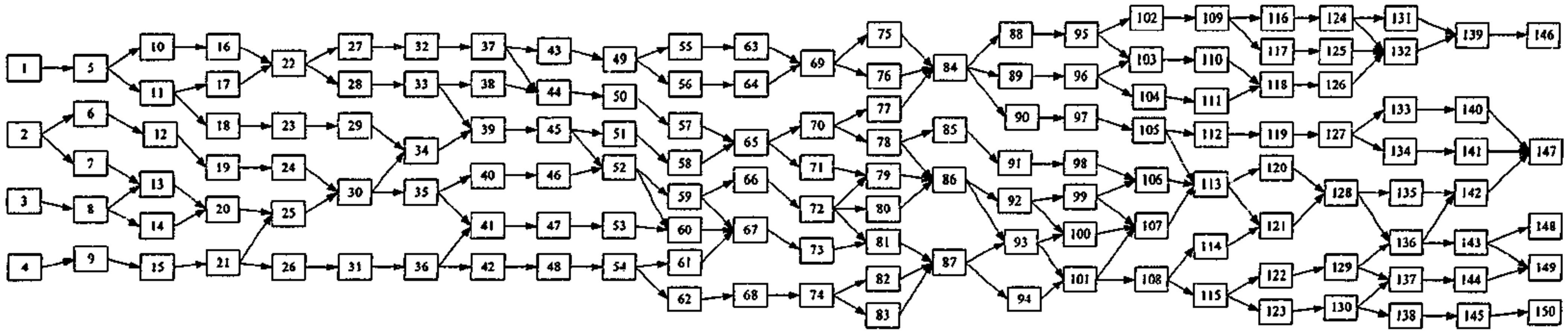
(B) Network Diagram of a Project with 50 activities



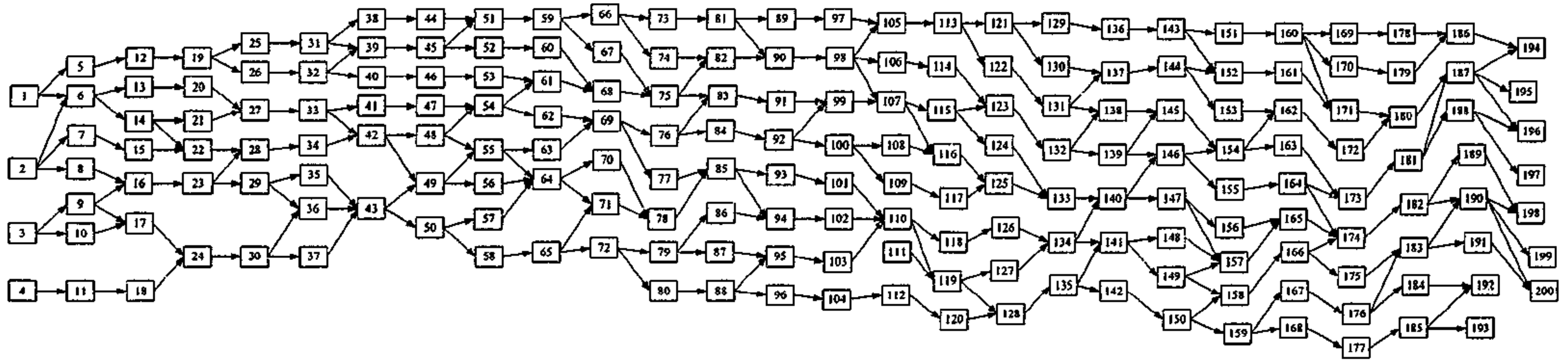
(C) Network Diagram of a Project with 100 activities



(D) Network Diagram of a Project with 150 activities



(E) Network Diagram of a Project with 200 activities



# Appendix B

# Information on Experimental Results

(A) The evaluation of generation parameter for fuzzy GA in five projects

(a) A project with 25 activities

| Generations | Project completion time (Fuzzy ranking index) | CPU time (msec) |
|---|---|---|
| 5 | 111.67 | 1101 |
| 10 | 108 | 1164 |
| 15 | 104.22 | 1226 |
| 20 | 104 | 1320 |
| 25 | 103.67 | 1421 |
| 30 | 102.33 | 1484 |
| 35 | 102.33 | 1570 |
| 40 | 101.67 | 1632 |
| 45 | 100 | 1726 |
| 50 | 99.33 | 1843 |
| 55 | 99 | 1906 |
| 60 | 97.67 | 1976 |
| 65 | 97 | 2086 |
| 70 | 88.22 | 2187 |
| 75 | 90 | 2297 |
| 80 | 89.22 | 2320 |
| 85 | 88.67 | 2398 |
| 90 | 92.66 | 2539 |
| 95 | 90.33 | 2562 |
| 100 | 89.23 | 2656 |

(b) A project with 50 activities

| Generations | Project completion time (Fuzzy ranking index) | Fuzzy project completion time | CPU time (msec) |
|---|---|---|---|
| 10 | 127.3665754 | (102,123,125,142) | 2328 |
| 20 | 127.0009842 | (82,105,116,136) | 2625 |
| 30 | 125.3759997 | (77,103,110,137) | 3000 |
| 40 | 123.5010121 | (84,104,107,128) | 3344 |
| 50 | 122.699437 | (97,116,123,142) | 3672 |
| 60 | 121.3131983 | (85,109,113,133) | 3953 |
| 70 | 121.3131983 | (83,108,113,139) | 4297 |
| 80 | 120.1274745 | (89,109,118,136) | 4562 |
| 90 | 119.501046 | (94,117,125,149) | 4859 |
| 100 | 105.35 | (94,117,125,149) | 5219 |
| 110 | 103.6968654 | (100,125,129,154) | 5516 |
| 120 | 104.95 | (80,107,112,136) | 5875 |
| 130 | 105.8238744 | (84,107,113,132) | 6109 |
| 140 | 107.23 | (100,121,126,147) | 6562 |
| 150 | 106.65 | (95,116,123,146) | 6828 |
| 160 | 106.8170888 | (97,122,130,153) | 7109 |
| 170 | 107.21 | (86,112,117,141) | 7438 |
| 180 | 106.8170888 | (98,116,120,136) | 7906 |
| 190 | 108.6411471 | (86,106,110,132) | 8141 |
| 200 | 108.7418973 | (86,108,115,134) | 8391 |
| 210 | 109.7190949 | (81,104,106,125) | 8609 |
| 220 | 109.1916332 | (84,111,112,132) | 8984 |
| 230 | 109.5857613 | (99,125,131,155) | 9328 |
| 240 | 108.5421405 | (86,111,120,140) | 9719 |
| 250 | 107.6920716 | (84,104,112,131) | 9938 |

(c) A project with 100 activities

| Generations | Project completion time (Fuzzy ranking index) | Fuzzy project completion time | CPU time (sec) |
|---|---|---|---|
| 10 | 1035.063093 | (861,1011,1061,1208) | 6.25 |
| 20 | 997.9498865 | (808,982,1029,1179) | 6.922 |
| 30 | 997.6869695 | (816,973,1023,1179) | 7.812 |
| 40 | 996.1251256 | (820,973,1023,1170) | 8.484 |
| 50 | 995.376113 | (811,969,1023,1179) | 9.203 |
| 60 | 988.7320854 | (815,970,1015,1158) | 9.906 |
| 70 | 982.1220574 | (813,963,1010,1146) | 10.875 |
| 80 | 981.4899554 | (792,959,1009,1168) | 11.344 |
| 90 | 980.8194049 | (797,959,1012,1159) | 12.172 |
| 100 | 979.5465362 | (795,959,1007,1160) | 12.75 |
| 110 | 975.4985635 | (786,954,1007,1159) | 13.781 |
| 120 | 972.0505067 | (795,952,999,1145) | 14.391 |
| 130 | 971.7532819 | (810,949,997,1132) | 15.109 |
| 140 | 969.4409894 | (791,948,999,1143) | 15.875 |
| 150 | 967.9848314 | (783,947,994,1150) | 16.609 |
| 160 | 964.5215408 | (784,950,996,1135) | 17.234 |
| 170 | 955.9965077 | (793,936,981,1116) | 18.266 |
| 180 | 955.3125786 | (781,930,980,1130) | 19.016 |
| 190 | 785.9069178 | (690,765,826,927) | 19.859 |
| 200 | 790.5430125 | (634,788,832,975) | 20.234 |
| 210 | 951.3038739 | (781,931,976,1119) | 21.344 |
| 220 | 946.7553406 | (777,926,975,1112) | 22.031 |
| 230 | 942.1028359 | (775,927,971,1101) | 22.266 |
| 240 | 859.865676 | (703,842,884,1013) | 22.875 |
| 250 | 843.5873213 | (680,825,874,1000) | 24.062 |

(d) A project with 150 activities

| Generations | Project completion time (Fuzzy ranking index) | Fuzzy project completion time | CPU time (sec) |
|---|---|---|---|
| 10 | 503.0818018 | (167,201,207,1141) | 11.359 |
| 20 | 499.4150525 | (163,197,203,1138) | 12.609 |
| 30 | 497.0780391 | (163,195,201,1133) | 13.516 |
| 40 | 497.0519764 | (161,195,199,1135) | 14.469 |
| 50 | 490.7301507 | (157,189,194,1126) | 15.703 |
| 60 | 489.4112939 | (155,187,193,1126) | 16.953 |
| 70 | 489.0649957 | (154,187,192,1126) | 17.969 |
| 80 | 486.0836751 | (149,184,190,1125) | 19.234 |
| 90 | 485.0533964 | (149,184,188,1122) | 20.266 |
| 100 | 480.1001731 | (134,178,184,1128) | 21.312 |
| 110 | 480.0815563 | (143,177,183,1120) | 22.453 |
| 120 | 479.7109165 | (149,177,181,1113) | 23.594 |
| 130 | 472.1240804 | (132,170,178,1114) | 24.828 |
| 140 | 468.4150577 | (132,166,172,1107) | 25.875 |
| 150 | 467.4520269 | (128,164,172,1110) | 26.938 |
| 160 | 232.7646864 | (186,226,237,281) | 28.156 |
| 170 | 229.0599528 | (184,224,236,273) | 29.031 |
| 180 | 228.1338805 | (179,222,233,278) | 30.031 |
| 190 | 226.5998762 | (182,221,229,273) | 31.172 |
| 200 | 226.3202786 | (179,224,226,275) | 32.484 |
| 210 | 225.323135 | (182,222,228,269) | 33.312 |
| 220 | 224.000558 | (178,219,229,270) | 34.578 |
| 230 | 223.6425334 | (174,219,227,274) | 35.516 |
| 240 | 222.6454881 | (180,219,225,266) | 36.5 |
| 250 | 222.1331376 | (183,217,226,262) | 37.734 |
| 260 | 220.8207904 | (180,217,224,262) | 39.078 |
| 270 | 220.3270274 | (174,218,222,267) | 39.812 |
| 280 | 219.6851572 | (178,215,225,261) | 40.859 |
| 290 | 217.6928837 | (167,217,221,267) | 41.797 |
| 300 | 216.3535198 | (168,214,220,264) | 43.125 |
| 310 | 214.6598407 | (171,213,215,259) | 44.438 |
| 320 | 212.9755842 | (175,210,212,253) | 45.578 |
| 330 | 212.8221357 | (167,209,216,259) | 46.219 |
| 340 | 211.4772252 | (161,208,213,263) | 47.625 |
| 350 | 210.9546141 | (172,206,214,251) | 48.578 |
| 360 | 216.2297465 | (172,213,222,259) | 50.031 |
| 370 | 214.7783634 | (175,208,215,252) | 50.906 |
| 380 | 217.3590665 | (168,214,222,266) | 52.344 |
| 390 | 214.2191038 | (172,212,219,255) | 52.766 |
| 400 | 217.5005747 | (171,212,223,264) | 55.203 |

(e) A project with 200 activities

| Generations | Project completion time (Fuzzy ranking index) | Fuzzy project completion time | CPU time (sec) |
|---|---|---|---|
| 10 | 540.2776156 | (187,234,248,1199) | 23.234 |
| 20 | -538.1724627 | (192,234,244,1188) | 25.094 |
| 30 | 538.1634965 | (195,234,244,1185) | 26.438 |
| 40 | 536.5356768 | (191,230,242,1188) | 27.891 |
| 50 | 535.5139428 | (185,230,240,1191) | 29.703 |
| 60 | 532.1877261 | (182,229,239,1185) | 31.406 |
| 70 | 530.7847692 | (182,226,233,1184) | 33.219 |
| 80 | 529.5208749 | (179,226,236,1183) | 34.625 |
| 90 | 528.8169818 | (185,226,235,1175) | 35.766 |
| 100 | 523.9747252 | (174,217,233,1180) | 38 |
| 110 | 518.0829201 | (169,214,219,1171) | 39.547 |
| 120 | 514.9184746 | (165,211,224,1168) | 41.188 |
| 130 | 513.642679 | (157,206,221,1177) | 43.391 |
| 140 | 512.9200614 | (170,206,221,1162) | 44.641 |
| 150 | 511.965459 | (164,205,221,1166) | 46.219 |
| 160 | 509.2121688 | (166,207,219,1154) | 47.5 |
| 170 | 505.0344576 | (152,202,204,1161) | 49.984 |
| 180 | 253.3154538 | (197,246,260,310) | 50.562 |
| 190 | 252.1086803 | (199,247,252,308) | 52.031 |
| 200 | 250.5753003 | (195,242,256,308) | 53.578 |
| 210 | 250.3620819 | (196,246,256,304) | 55 |
| 220 | 249.2305877 | (199,242,252,302) | 56.75 |
| 230 | 248.816502 | (193,242,255,305) | 58.094 |
| 240 | 247.5396364 | (194,244,253,300) | 59.5 |
| 250 | 245.9663191 | (192,241,249,301) | 61.75 |
| 260 | 245.6831758 | (189,240,252,302) | 63.656 |
| 270 | 243.3182704 | (195,238,248,292) | 64.938 |
| 280 | 243.1573761 | (194,241,244,293) | 66.719 |
| 290 | 242.0005165 | (202,238,246,282) | 68.156 |
| 300 | 241.0005187 | (191,238,244,291) | 69.234 |
| 310 | 237.4526992 | (186,231,242,290) | 71.375 |
| 320 | 237.356588 | (196,235,241,278) | 72.75 |
| 330 | 237.0005274 | (177,230,244,297) | 73.734 |
| 340 | 235.9513472 | (179,231,237,295) | 76.141 |
| 350 | 233.6301639 | (202,229,237,266) | 77.828 |
| 360 | 232.6082051 | (181,227,235,286) | 79.859 |
| 370 | 231.50054 | (183,227,236,280) | 81.797 |
| 380 | 231.3709114 | (175,225,239,287) | 83.438 |
| 390 | 229.7591632 | (179,222.235,282) | 84.562 |
| 400 | 229.0005459 | (172,223,235,286) | 86.734 |
| 410 | 228.6407576 | (171,223,233,287) | 88.234 |
| 420 | 219.5052437 | (166,220,221,272) | 89.594 |
| 430 | 228.1241104 | (178,220,235,279) | 91.25 |
| 440 | 226.0005531 | (172,221,231,280) | 92.75 |
| 450 | 247.4346127 | (192,238,255,304) | 94.312 |

| Generations | Project completion time (Fuzzy ranking index) | Fuzzy project completion time | CPU time (sec) |
|---|---|---|---|
| 460 | 231.6210517 | (176,222,240,288) | 95.969 |
| 470 | 246.9909813 | (196,245,247,299) | 96.906 |
| 480 | 231.0005411 | (184,226,236,278) | 99.641 |
| 490 | 240.8606387 | (189,237,246,292) | 101.109 |
| 500 | 242.4671811 | (194,238,245,292) | 101.688 |
| 510 | 236.7578364 | (187,229,242,288) | 104.812 |
| 520 | 255.5004892 | (204,248,263,307) | 105.984 |
| 530 | 233.1338687 | (184,227,238,283) | 107.531 |
| 540 | 234.6831997 | (178,229,241,291) | 108.578 |
| 550 | 235.0005319 | (183,231,239,287) | 110.203 |

(B) The evaluation of the tabu size for fuzzy GA with tabu in five projects

(a) A project with 25 activities

| Tabu size | Project completion time (Fuzzy ranking index) | Fuzzy project completion time | CPU time (sec) |
|---|---|---|---|
| 2 | 81.3348765 | (61,82,101) | 28.057 |
| 4 | 80.66821944 | (63,81,98) | 28.193 |
| 6 | 78.668259 | (60,79,97) | 28.281 |
| 8 | 79.66823897 | (63,80,96) | 28.37 |
| 10 | 78.66826907 | (64,80,92) | 28.38 |
| 12 | 78.66826907 | (57,80,99) | 28.53 |
| 14 | 79.66823897 | (59,80,100) | 28.595 |
| 16 | 77.66826918 | (53,77,103) | 28.645 |
| 18 | 78.33492566 | (58,78,99) | 28.663 |
| 20 | 77.66827954 | (59,78,96) | 28.81 |
| 22 | 78.33492566 | (58,78,99) | 28.826 |
| 24 | 77.33495666 | (60,78,94) | 28.854 |
| 26 | 77.33494621 | (52,77,103) | 28.856 |
| 28 | 76.33496729 | (61,76,92) | 28.857 |
| 30 | 76.33495654 | (52,75,102) | 28.957 |
| 32 | 76.00165552 | (51,77,100) | 28.998 |
| 34 | 76.00164472 | (61,76,91) | 29.004 |
| 36 | 76.00164472 | (57,76,95) | 29.015 |
| 38 | 76.00164472 | (61,76,91) | 29.197 |
| 40 | 75.66832227 | (52,76,99) | 30.63 |
| 42 | 75.33499994 | (55,76,95) | 30.64 |
| 44 | 75.00167774 | (57,76,92) | 30.646 |
| 46 | 75.00166665 | (50,75,100) | 30.651 |
| 48 | 75.00166665 | (55,75,95) | 30.667 |
| 50 | 65.335227 | (51,64,81) | 30.678 |
| 52 | 66.66853225 | (46,66,88) | 31.086 |
| 54 | 71.66841489 | (51,72,92) | 31.682 |
| 56 | 74.66832202 | (57,73,94) | 32.213 |
| 58 | 67.66851848 | (42,68,93) | 33.265 |

| Tabu size | Project completion time (Fuzzy ranking index) | Fuzzy project completion time | CPU time (sec) |
|---|---|---|---|
| 60 | 74.00170056 | (58,75,89) | 33.765 |
| 62 | 73.6683788 | (53,75,93) | 34.276 |
| 64 | 71.66841489 | (50,72,93) | 34.848 |
| 66 | 74.00170056 | (55,75,92) | 35.364 |
| 68 | 72.66839077 | (49,73,96) | 36.185 |
| 70 | 71.66839049 | (47,70,98) | 36.911 |
| 72 | 72.66837893 | (55,72,91) | 37.511 |
| 74 | 69.33513186 | (47,69,92) | 37.916 |
| 76 | 73.3350456 | (51,74,95) | 38.532 |
| 78 | 71.66841489 | (55,72,88) | 39.258 |
| 80 | 74.66833327 | (55,74,95) | 39.958 |
| 82 | 71.33508155 | (55,71,88) | 40.52 |
| 84 | 74.66833327 | (52,74,98) | 41.026 |
| 86 | 72.66839077 | (54,73,91) | 41.803 |
| 88 | 74.66833327 | (53,74,97) | 42.46 |
| 90 | 68.66847818 | (53,68,85) | 43.078 |
| 92 | 73.33505717 | (55,75,90) | 43.75 |
| 94 | 67.00187954 | (46,68,87) | 44.119 |
| 96 | 69.33513186 | (49,69,90) | 44.713 |
| 98 | 68.66851783 | (48,71,87) | 45.145 |
| 100 | 66.66853225 | (42,66,92) | 46.25 |

(b) A project with 50 activities

| Tabu size | Project completion time (Fuzzy ranking index) | Fuzzy project completion time | CPU time (sec) |
|---|---|---|---|
| 5 | 97.69219081 | (74,94,102,121) | 93.086 |
| 10 | 95.17675425 | (68,94,97,122) | 93.898 |
| 15 | 95.00131578 | (70,94,96,120) | 94.648 |
| 20 | 94.254422 | (70,92,99,117) | 95.586 |
| 25 | 94.17820038 | (71,93,96,117) | 96.172 |
| 30 | 92.61426747 | (64,91,98,119) | 96.266 |
| 35 | 92.09225414 | (68,88,93,118) | 96.484 |
| 40 | 91.86446088 | (66,90,95,117) | 96.5 |
| 45 | 91.19184944 | (64,87,96,118) | 96.82 |
| 50 | 88.68562266 | (63,86,92,114) | 96.828 |
| 55 | 89.00140448 | (66,86,92,112) | 98.125 |
| 60 | 91.06260471 | (69,89,95,112) | 98.272 |
| 65 | 90.65216748 | (71,89,91,111) | 98.914 |
| 70 | 89.81957092 | (65,87,92,115) | 98.966 |
| 75 | 89.68922699 | (62,86,94,117) | 99.296 |
| 80 | 90.68765541 | (68,88,94,113) | 99.406 |
| 85 | 89.43998627 | (65,85,92,115) | 99.541 |
| 90 | 89.50139664 | (68,85,94,111) | 99.82 |
| 95 | 90.51251246 | (63,89,97,115) | 100.625 |
| 100 | 89.31512243 | (67,86,92,112) | 101.75 |

(c) A project with 100 activities

| Tabu size | Project completion time (Fuzzy ranking index) | Fuzzy project completion time | CPU time (sec) |
|---|---|---|---|
| 5 | 947.1031789 | (780,930,973,1109) | 1027.445 |
| 10 | 915.5445257 | (733,902,943,1090) | 1029.235 |
| 15 | 906.5223316 | (733,891,934,1073) | 1030.864 |
| 20 | 901.4915035 | (729,883,930,1068) | 1032.325 |
| 25 | 893.0423344 | (716,874,919,1066) | 1040.357 |
| 30 | 883.9845978 | (711,866,912,1051) | 1042.897 |
| 35 | 770.41364 | (602,750,792,934) | 1046.025 |
| 40 | 764.9300296 | (577,742,791,951) | 1050.951 |
| 45 | 878.4916679 | (721,860,902,1033) | 1065.264 |
| 50 | 877.1269719 | (697,853,905,1055) | 1070.875 |
| 55 | 871.1268441 | (699,847,894,1044) | 1078.092 |
| 60 | 871.2122652 | (692,855,900,1043) | 1186.235 |
| 65 | 847.8991648 | (679,832,872,1012) | 1192.368 |
| 70 | 855.9260721 | (683,836,879,1027) | 1209.483 |
| 75 | 847.8088646 | (664,827,878,1026) | 1234.273 |
| 80 | 840.6866259 | (670,818,864,1011) | 1258.357 |
| 85 | 839.2467247 | (672,826,862,1001) | 1266.873 |
| 90 | 787.402458 | (617,771,812,953) | 1290.964 |
| 95 | 792.0250085 | (616,776,820,961) | 1304.981 |
| 100 | 772.9791264 | (588,753,798,955) | 1318.954 |

(d) A project with 150 activities

| Tabu size | Project completion time (Fuzzy ranking index) | Fuzzy project completion time | CPU time (min) |
|---|---|---|---|
| 5 | 206.2719219 | (149,180,186,229) | 33 |
| 10 | 205.643464 | (159,195,204,242) | 33.05 |
| 15 | 203.6810266 | (148,190,198,241) | 33.29 |
| 20 | 203.3298324 | (168,201,209,246) | 33.79 |
| 25 | 186.8611269 | (164,202,204,243) | 34.24 |
| 30 | 190.3506578 | (154,191,198,240) | 36.11 |
| 35 | 191.865236 | (162,198,201,240) | 37.16 |
| 40 | 195.8150708 | (152,189,193,228) | 38.26 |
| 45 | 201.1378753 | (146,188,196,240) | 39.44 |
| 50 | 199.1031893 | (148,188,197,235) | 40.63 |
| 55 | 194.3207747 | (143,188,197,241) | 40.99 |
| 60 | 200.4821034 | (144,184,188,235) | 41.43 |
| 65 | 192.1813364 | (146,189,195,236) | 42.08 |
| 70 | 196.1045774 | (158,194,201,242) | 43.11 |
| 75 | 191.354821 | (159,200,208,248) | 44.18 |
| 80 | 200.1346815 | (155,196,205,248) | 45.65 |
| 85 | 190.6860144 | (167,202,208,245) | 46.73 |
| 90 | 191.285204 | (143,185,197,238) | 47.77 |
| 95 | 192.6411706 | (153,190,201,239) | 48.91 |
| 100 | 188.2848703 | (154,187,193,230) | 50.57 |

(e) A project with 200 activities

| Tabu size | Project completion time (Fuzzy ranking index) | Fuzzy project completion time | CPU time (min) |
|---|---|---|---|
| 5 | 219.5675809 | (177,214,227,261) | 41.61 |
| 10 | 219.2792568 | (164,213,223,276) | 42.41 |
| 15 | 217.2826248 | (158,211,221,278) | 43.53 |
| 20 | 193.6778389 | (149,191,197,238) | 44.59 |
| 25 | 195.5006394 | (142,192,199,249) | 46.55 |
| 30 | 208.6037714 | (161,203,211,258) | 48.39 |
| 35 | 213.6287893 | (168,207,219,260) | 50.62 |
| 40 | 214.6831224 | (167,210,220,262) | 53.44 |
| 45 | 213.0005869 | (160,211,215,266) | 55.77 |
| 50 | 210.9484172 | (160,204,216,263) | 57.93 |
| 55 | 214.6827532 | (156,209,221,273) | 60.21 |
| 60 | 210.0005952 | (157,204,216,263) | 63.11 |
| 65 | 210.5005938 | (156,206,215,265) | 66.16 |
| 70 | 214.3282181 | (158,212,216,271) | 69.7 |
| 75 | 213.0532197 | (153,207,221,272) | 72.83 |
| 80 | 201.1786154 | (153,198,205,249) | 76.93 |
| 85 | 199.0006281 | (156,196,202,242) | 80.28 |
| 90 | 202.0006188 | (146,199,205,258) | 85.62 |
| 95 | 201.3660064 | (154,197,207,248) | 90.4 |
| 100 | 194.3696918 | (144,189,201,244) | 106.31 |

(C) The evaluation of the Markov Chain length for fuzzy SA in five projects

(a) A project with 25 activities

| Markov chain length | Project completion time (Fuzzy ranking index) | Fuzzy project completion time | CPU time (msec) |
|---|---|---|---|
| 10 | 105.0012018 | (79,107,129) | 773.4375 |
| 20 | 103.6678744 | (88,104,119) | 1507.8125 |
| 30 | 87.00142022 | (68,89,108) | 2304.6875 |
| 40 | 87.33472996 | (72,87,107) | 2929.6875 |
| 50 | 95.33458331 | (77,97,123) | 3664.0625 |
| 60 | 94.00134389 | (72,96,114) | 4484.375 |
| 70 | 97.66795524 | (75,99,119) | 5406.25 |
| 80 | 96.33463539 | (81,97,111) | 6148.4375 |
| 90 | 94.3345895 | (75,96,124) | 6820.3125 |
| 100 | 93.00134408 | (73,93,113) | 7351.5625 |

(b) A project with 50 activities

| Markov chain length | Project completion time (Fuzzy ranking index) | Fuzzy project completion time | CPU time (msec) |
|---|---|---|---|
| 10 | 117.0607739 | (87,114,122,146) | 1773.4375 |
| 20 | 117.0418889 | (94,116,120,139) | 3375 |
| 30 | 114.6307296 | (90,113,120,137) | 5109.375 |
| 40 | 114.5010917 | (89,113,116,140) | 7117.1875 |
| 50 | 95.00131578 | (72,93,97,118) | 8343.75 |
| 60 | 100.4522186 | (78,97,102,124) | 10421.875 |
| 70 | 110.7154266 | (86,112,114,133) | 12320.3125 |
| 80 | 109.0011468 | (85,106,112,133) | 14320.3125 |
| 90 | 106.6908123 | (80,109,111,136) | 16000 |
| 100 | 108.1261667 | (86,107,113,128) | 17273.4375 |
| 110 | 107.6995805 | (87,108,110,127) | 19000 |
| 120 | 107.1950652 | (85,105,113,132) | 21179.6875 |
| 130 | 103.0207187 | (101,125,130,154) | 22554.6875 |
| 140 | 101.8794361 | (79,99,106,124) | 24375 |
| 150 | 101.8182191 | (79,99,104,125) | 26039.0625 |

(c) A project with 100 activities

| Markov chain length | Project completion time (Fuzzy ranking index) | Fuzzy project completion time | CPU time (sec) |
|---|---|---|---|
| 10 | 958.1365687 | (778,936,989,1133) | 5.546875 |
| 20 | 949.6909049 | (773,927,978,1123) | 9.109375 |
| 30 | 944.3679487 | (764,923,972,1121) | 11.0859375 |
| 40 | 941.7475012 | (774,920,966,1108) | 14.21101563 |
| 50 | 911.0001372 | (746,888,934,1076) | 17.6679 |
| 60 | 905.7991206 | (730,887,934,1076) | 22.179 |
| 70 | 888.185632 | (727,868,909,1049) | 27.554 |
| 80 | 876.2497198 | (703,853,902,1048) | 31.46 |
| 90 | 862.3139227 | (697,840,884,1028) | 35.695 |
| 100 | 770.7416953 | (604,751,793,936) | 39.203 |
| 110 | 772.9529855 | (608,756,798,939) | 46.12 |
| 120 | 861.7738681 | (676,844,889,1042) | 52.125 |
| 130 | 859.865676 | (703,842,884,1013) | 60.671875 |
| 140 | 854.4776942 | (681,837,882,1022) | 65.242 |
| 150 | 860.9069178 | (690,844,886,1027) | 70.429 |
| 160 | 844.8188013 | (669,830,871,1014) | 75.507 |
| 170 | 812.8057785 | (634,791,839,989) | 86.421 |
| 180 | 837.2223721 | (648,819,868,1019) | 89.664 |
| 190 | 806.5430125 | (634,788,832,975) | 95.648 |
| 200 | 789.056145 | (617,769,816,957) | 99.71 |

(d) A project with 150 activities

| Markov Chain length | Project completion time (Fuzzy ranking index) | Fuzzy project completion time | CPU time (sec) |
|---|---|---|---|
| 10 | 470.3966908 | (136,168,173,1107) | 7.742 |
| 20 | 253.0004941 | (207,250,256,299) | 15.609 |
| 30 | 233.3605364 | (187,230,238,279) | 23.726 |
| 40 | 228.1352989 | (186,223,232,271) | 31.671 |
| 50 | 227.3237818 | (181,224,230,274) | 40.046 |
| 60 | 223.3595348 | (175,220,228,271) | 48.14 |
| 70 | 223.2297283 | (179,220,229,266) | 55.671 |
| 80 | 220.6369293 | (171,215,225,271) | 63.078 |
| 90 | 220.119612 | (181,216,219,262) | 72.406 |
| 100 | 219.9423146 | (175,213,225,266) | 77.898 |
| 110 | 219.6066263 | (179,215,219,263) | 84.203 |
| 120 | 218.3189242 | (178,214,222,259) | 94.57 |
| 130 | 217.677994 | (174,215,221,261) | 101.937 |
| 140 | 217.4568842 | (171,212,221,265) | 108.07 |
| 150 | 208.6802111 | (161,205,213,256) | 116.64 |
| 160 | 209.3198949 | (166,205,213,253) | 127.32 |
| 170 | 213.0426879 | (171,210,220,258) | 131.984 |
| 180 | 213.5649431 | (169,208,221,257) | 139.804 |
| 190 | 212.8210997 | (171,209,216,255) | 151.671 |
| 200 | 212.1452424 | (163,208,215,262) | 159.125 |
| 210 | 211.6332428 | (168,206,216,256) | 163.46 |
| 220 | 212.1241516 | (162,204,219,263) | 177.593 |
| 230 | 211.9882426 | (173,210,212,252) | 183.984 |
| 240 | 210.6464258 | (166,207,213,256) | 192.929 |
| 250 | 209.9575832 | (168,205,213,253) | 198.89 |

(e) A project with 200 activities

| Markov chain length | Project completion time (Fuzzy ranking index) | Fuzzy project completion time | CPU time (sec) |
|---|---|---|---|
| 10 | 517.1670005 | (174,214,224,1163) | 20.549 |
| 20 | 243.0005144 | (189,237,249,297) | 41.96 |
| 30 | 239.7782976 | (191,234,243,290) | 62.825 |
| 40 | 239.6797199 | (178,235,245,301) | 84.671 |
| 50 | 239.6429451 | (189,235,243,291) | 103.068 |
| 60 | 238.7373672 | (187,234,246,289) | 123.983 |
| 70 | 238.0321052 | (193,236,242,282) | 144.312 |
| 80 | 237.1285026 | (188,230,243,287) | 164.967 |
| 90 | 236.1433857 | (178,231,240,295) | 182.562 |
| 100 | 235.0005319 | (188,231,239,282) | 206.982 |
| 110 | 234.9304672 | (168,223,245,303) | 229.635 |
| 120 | 234.500533 | (183,233,236,286) | 246.239 |
| 130 | 234.4684219 | (184,230,237,286) | 265.491 |
| 140 | 234.1793959 | (177,230,239,291) | 287.929 |
| 150 | 234.0464068 | (184,230,240,283) | 301.538 |

| Markov Chain length | Project completion time (Fuzzy ranking index) | Fuzzy project completion time | CPU time (sec) |
|---|---|---|---|
| 160 | 233.8179255 | (182,228,239,286) | 330.4 |
| 170 | 233.0005365 | (172,231,235,294) | 359.476 |
| 180 | 232.0005388 | (184,229,235,280) | 362.476 |
| 190 | 231.3661324 | (175,226,238,287) | 393.475 |
| 200 | 210.5005938 | (157,205,216,264) | 411.426 |
| 210 | 214.0005841 | (160,208,220,268) | 429.487 |
| 220 | 230.6697184 | (170,226,241,288) | 457.813 |
| 230 | 229.8810477 | (184,223,238,275) | 469.65 |
| 240 | 229.5971811 | (176,223,233,285) | 485.928 |
| 250 | 226.8712414 | (175,221,234,278) | 501.481 |
| 260 | 227.5005494 | (177,223,232,278) | 523.202 |
| 270 | 223.2998753 | (177,219,221,273) | 560.12 |
| 280 | 226.8135588 | (173,219,234,281) | 561.897 |
| 290 | 219.1831794 | (167,214,225,271) | 569.744 |
| 300 | 217.6213673 | (167,213,219,270) | 609.626 |

(D) The investigation of the cooling ratio for fuzzy SA with five projects

(a) A project with 25 activities

| Cooling ratio | Project completion time (Fuzzy ranking index) | Fuzzy project completion time | CPU time (msec) |
|---|---|---|---|
| 0.1 | 104.6678685 | (85,106,123) | 1484.375 |
| 0.2 | 101.0012437 | (78,102,123) | 1667 |
| 0.3 | 100.6679043 | (82,100,120) | 1700 |
| 0.4 | 100.0012562 | (83,101,116) | 1885.9375 |
| 0.5 | 97.6679487 | (77,98,118) | 2820.3125 |
| 0.6 | 95.33465595 | (71,97,118) | 3245 |
| 0.7 | 94.00133684 | (71,95,116) | 5484.375 |
| 0.8 | 85.3347867 | (68,84,104) | 6250 |
| 0.9 | 88.33474575 | (71,88,106) | 7625 |
| 0.95 | 88.0014285 | (70,89,105) | 9734.375 |

(b) A project with 50 activities

| Cooling ratio | Solutions (Fuzzy ranking index) | Fuzzy project completion time | CPU time (sec) |
|---|---|---|---|
| 0.1 | 129.576716 | (106,125,131,155) | 2.062 |
| 0.2 | 121.3194369 | (91,118,124,152) | 2.718 |
| 0.3 | 117.8854216 | (94,118,121,140) | 3.109 |
| 0.4 | 117.7510676 | (96,116,122,138) | 4.843 |
| 0.5 | 112.5611204 | (88,113,116,135) | 5.89 |
| 0.6 | 112.0011161 | (89,111,113,135) | 7.89 |
| 0.7 | 110.0011364 | (87,106,114,133) | 10.937 |
| 0.8 | 105.8238744 | (84,107,113,132) | 17.859 |
| 0.9 | 107.0881347 | (85,107,111,127) | 36.937 |
| 0.95 | 102.4086379 | (81,102,106,122) | 73.953 |

(c) A project with 100 activities

| Cooling ratio | Project completion time (Fuzzy ranking index) | Fuzzy project completion time | CPU time (sec) |
|---|---|---|---|
| 0.1 | 981.7987564 | (808,962,1006,1153) | 11.14 |
| 0.2 | 968.626538 | (787,943,993,1151) | 15.453 |
| 0.3 | 957.791696 | (773,938,982,1140) | 22.062 |
| 0.4 | 949.8633797 | (776,930,976,1120) | 26.281 |
| 0.5 | 903.3475073 | (732,887,931,1068) | 37.015 |
| 0.6 | 879.5463166 | (710,861,905,1045) | 49.781 |
| 0.7 | 860.9240079 | (700,843,882,1020) | 65.109 |
| 0.8 | 769.9233037 | (591,750,793,947) | 107.89 |
| 0.9 | 843.5873213 | (680,825,874,1000) | 222.109 |
| 0.95 | 842.9304001 | (660,822,872,1021) | 448.031 |

(d) A project with 150 activities

| Cooling ratio | Project completion time (Fuzzy ranking index) | Fuzzy project completion time | CPU time (sec) |
|---|---|---|---|
| 0.1 | 476.01281 | (139,175,176,1114) | 23.109 |
| 0.2 | 228.4791784 | (173,225,230,285) | 37.625 |
| 0.3 | 225.0005556 | (176,219,231,274) | 44.25 |
| 0.4 | 225.0005556 | (181,223,227,269) | 60.609 |
| 0.5 | 220.3204322 | (175,216,224,266) | 76.875 |
| 0.6 | 219.9638697 | (170,215,223,271) | 106.046 |
| 0.7 | 217.9689932 | (174,214,220,263) | 155.578 |
| 0.8 | 204.369407 | (162,200,210,246) | 233.359 |
| 0.9 | 217.5970635 | (175,212,220,262) | 482.25 |
| 0.95 | 204.5442788 | (165,199,218,259) | 1028 |

(e) A project with 200 activities

| Cooling ration | Project completion time (Fuzzy ranking index) | Fuzzy project completion time | CPU time (sec) |
|---|---|---|---|
| 0.1 | 508.4369841 | (160,206,212,1159) | 57.312 |
| 0.2 | 260.6402639 | (204,255,265,318) | 72.062 |
| 0.3 | 258.4956283 | (208,257,258,310) | 87.125 |
| 0.4 | 255.1349922 | (204,249,260,307) | 119.343 |
| 0.5 | 248.7782779 | (200,244,249,300) | 160.765 |
| 0.6 | 241.2597759 | (194,234,246,290) | 196.218 |
| 0.7 | 239.8838522 | (188,231,245,294) | 298.703 |
| 0.8 | 214.0005841 | (160,208,220,268) | 459.562 |
| 0.9 | 234.1893247 | (185,227,242,283) | 997.671 |
| 0.95 | 217.6213673 | (167,213,219,270) | 1974.593 |

(E) The evaluation of the tabu size for fuzzy SA with tabu with five projects

(a) A project with 25 activities

| Tabu size | Project completion time (Fuzzy ranking index) | Fuzzy project completion time | CPU time (sec) |
|---|---|---|---|
| 10 | 81.00155271 | (62,82,99) | 32.4062 |
| 20 | 79.66822912 | (63,79,97) | 33.282 |
| 30 | 78.66824889 | (60,78,98) | 34.25 |
| 40 | 76.66827931 | (60,75,95) | 35.4218 |
| 50 | 62.66853225 | (46,62,88) | 36.4375 |
| 60 | 62.335227 | (51,62,81) | 37.4171 |
| 70 | 75.66832227 | (57,76,94) | 40.39 |
| 80 | 74.66833327 | (52,74,98) | 43.75 |
| 90 | 74.66832202 | (57,73,94) | 50 |
| 100 | 71.66841489 | (51,72,92) | 58.4375 |

(b) A project with 50 activities

| Tabu size | Project completion time (Fuzzy ranking index) | Fuzzy project completion time | CPU time (sec) |
|---|---|---|---|
| 10 | 94.75950528 | (70,95,99,117) | 101.25 |
| 20 | 93.91043377 | (68,93,98,118) | 102.7359 |
| 30 | 92.56499407 | (68,90,97,116) | 103.1925 |
| 40 | 91.74211316 | (67,90,96,115) | 103.98 |
| 50 | 87.61870517 | (65,83,91,111) | 104.5453 |
| 60 | 87.31589427 | (64,84,90,111) | 104.9312 |
| 70 | 91.69142424 | (67,88,96,116) | 108.656 |
| 80 | 90.86249187 | (63,89,94,118) | 113.015 |
| 90 | 89.69371168 | (64,90,92,114) | 118.68 |
| 100 | 90.61231922 | (62,85,95,119) | 125.089 |
| 110 | 89.68693093 | (66,87,93,113) | 131.785 |
| 120 | 88.13602829 | (65,85,90,112) | 141.8265 |

(c) A project with 100 activities

| Tabu size | Project completion time (Fuzzy ranking index) | Fuzzy project completion time | CPU time (sec) |
|---|---|---|---|
| 10 | 998.1172426 | (833,980,1025,1158) | 1041.25 |
| 20 | 977.2467266 | (804,955,1002,1149) | 1049.21875 |
| 30 | 977.1705924 | (800,959,1006,1148) | 1059.625 |
| 40 | 759.056145 | (577,729,776,917) | 1061.7578 |
| 50 | 760.056145 | (585,738,784,925) | 1075.25 |
| 60 | 850.9260721 | (683,831,874,1027) | 1193.125 |
| 70 | 878.4916679 | (721,860,902,1033) | 1213.109625 |
| 80 | 896.5980779 | (722,879,923,1066) | 1263.921 |
| 90 | 888.185632 | (727,868,909,1049) | 1293.08156 |
| 100 | 862.3139227 | (697,840,884,1028) | 1321.01 |

| Tabu size | Project completion time (Fuzzy ranking index) | Fuzzy project completion time | CPU time (sec) |
|---|---|---|---|
| 110 | 861.7738681 | (676,844,889,1042) | 1324.109 |
| 120 | 842.9304001 | (660,822,872,1021) | 1327.109 |
| 130 | 837.2223721 | (648,819,868,1019) | 1328.6046 |
| 140 | 812.8057785 | (634,791,839,989) | 1334.75 |
| 150 | 792.0250085 | (616,776,820,961) | 1378.11593 |

(d) A project with 150 activities

| Tabu size | Project completion time (Fuzzy ranking index) | Fuzzy project completion time | CPU time (min) |
|---|---|---|---|
| 10 | 208.000601 | (170,207,209,246) | 33.6703 |
| 20 | 205.8239 | (167,203,208,245) | 34.24538 |
| 30 | 186.077802 | (151,183,189,274) | 36.34 |
| 40 | 181.0392276 | (146,181,184,287) | 38.82 |
| 50 | 194.3207747 | (143,188,197,241) | 40.95 |
| 60 | 200.1346815 | (155,196,205,248) | 41.9512 |
| 70 | 192.1813364 | (146,189,195,236) | 43.56 |
| 80 | 196.1045774 | (158,194,201,242) | 46.02 |
| 90 | 195.068337 | (159,194,199,280) | 48.21 |
| 100 | 198.1031893 | (148,187,197,234) | 51.25 |
| 110 | 200.11031 | (142,182,183,233) | 54.7891 |
| 120 | 191.354821 | (159,200,208,248) | 57.141 |
| 130 | 191.285204 | (143,185,197,238) | 60.734 |
| 140 | 190.6860144 | (167,202,208,245) | 64.297 |
| 150 | 188.2848703 | (154,187,193,230) | 68.12 |
| 160 | 187.4206378 | (148,185,191,269) | 75.275 |

(e) A project with 200 activities

| Tabu size | Project completion time (Fuzzy ranking index) | Fuzzy project completion Time | CPU Time (min) |
|---|---|---|---|
| 10 | 218.1280205 | (170,214,217,269) | 42.4484 |
| 20 | 216.8150688 | (166,210,223,268) | 45.03422 |
| 30 | 206.9898497 | (162,205,207,253) | 49.0359 |
| 40 | 193.4292062 | (146,187,196,242) | 53.81 |
| 50 | 193.819959 | (145,189,198,243) | 58.2625 |
| 60 | 203.1829519 | (150,198,209,256) | 63.315 |
| 70 | 206.1818365 | (159,202,211,253) | 69.84 |
| 80 | 204.9003175 | (152,202,211,256) | 77.15 |
| 90 | 201.0006219 | (144,196,206,258) | 86.01 |
| 100 | 199.6412501 | (141,194,204,259) | 106.45 |
| 120 | 199.14646 | (147,195,202,252) | 126.49 |
| 130 | 198.03733 | (147,195,203,248) | 149.6578 |