DISCRETE TIME/COST TRADE-OFF PROJECT SCHEDULING WITH A NONRENEWABLE RESOURCE

A THESIS SUBMITTED TO

THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES

OF

MIDDLE EAST TECHNICAL UNIVERSITY

BY

SELİN KIRBIYIK

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

FOR

THE DEGREE OF MASTER OF SCIENCE

IN

INDUSTRIAL ENGINEERING

NOVEMBER 2009

Approval of the thesis:

DISCRETE TIME/COST TRADE-OFF PROJECT SCHEDULING WITH A NONRENEWABLE RESOURCE

submitted by **SELİN KIRBIYIK** in partial fulfillment of the requirements for the degree of **Master of Science in Industrial Engineering Department, Middle East Technical University** by,

Prof. Dr. Canan Özgen _____
Dean, Graduate School of **Natural and Applied Sciences**

Prof. Dr. Nur Evin Özdemirel _____
Head of Department, **Industrial Engineering**

Prof. Dr. Meral Azizoğlu _____
Supervisor, **Industrial Engineering Dept., METU**

Asst. Prof. Dr. Ferda Can Çetinkaya _____
Co-Supervisor, **Industrial Engineering Dept., Çankaya University**

Examining Committee Members:
Prof. Dr. Ömer Kırca _____
Industrial Engineering Dept., METU

Prof. Dr. Meral Azizoğlu _____
Industrial Engineering Dept., METU

Asst. Prof. Dr. Ferda Can Çetinkaya _____
Industrial Engineering Dept., Çankaya University

Prof. Dr. Sencer Yeralan _____
Agricultural and Biological Engineering Dept., University of Florida, USA

Vedat Aydemir, M.Sc. _____
TPAO, Exploration District Manager

Date: *25 November 2009*

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Last name: Selin Kırbıyık

Signature:

ABSTRACT

DISCRETE TIME/COST TRADE-OFF PROJECT SCHEDULING WITH A NONRENEWABLE RESOURCE

Kırbıyık, Selin
M.Sc., Department Industrial Engineering
Supervisor : Prof. Dr. Meral Azizoğlu
Co-Supervisor : Asst. Prof. Dr. Ferda Can Çetinkaya

November 2009, 71 pages

In this thesis, we consider a discrete time/cost trade-off problem with a single nonrenewable resource. We assume the resource is released at some prespecified time points and at some prespecified quantities. We also assume that the costs due to the activities are incurred at their completions. Our aim is to minimize total project completion time.

We formulate the problem as a pure integer programming model. We show that the problem is strongly $\mathcal{NP}$-hard. We find lower bounds by pure linear programming and mixed integer linear programming relaxations of the model. We develop three heuristic procedures using the optimal solutions of mixed integer linear program and pure linear program.

The results of our computational study reveal the satisfactory performance of our heuristic procedures.

Keywords: Project scheduling, discrete time/cost trade-off, nonrenewable resource.

ÖZ

YENİLENEMEYEN KAYNAKLA KESİKLİ ZAMAN/MALİYET ÖDÜNLEŞİM PROJE ÇİZELGELEMESİ

Kırbıyık,Selin
Yüksek Lisans, Endüstri Mühendisliği Bölümü
Tez Yöneticisi : Prof. Dr. Meral Azizoğlu
Ortak Tez Yöneticisi: Yrd. Doç. Dr. Ferda Can Çetinkaya

Kasım 2009, 71 sayfa

Bu tezde, yenilenemeyen kaynakla kesikli zaman/maliyet ödünleşim problemi ele alınmıştır. Kaynağın önceden belirlenmiş zamanlarda ve miktarlarda açığa çıktığını varsayıyoruz. Aynı zamanda, maliyetlerin de aktivite tamamlanma zamanlarında gerçekleştiğini varsayıyoruz. Amacımız, toplam proje tamamlanma süresini minimuma indirmektir.

Problem salt tamsayı programlama şeklinde formüle edilmiştir. Problemin $\mathcal{NP}$-zor olduğunu gösterdik. Alt sınırlar, salt tamsayı programlama ve modelin karma tamsayı programlamasının gevşetilmesiyle elde edildi. Karma tamsayı lineer programlamanın optimal sonuçlarıyla iki sezgisel yöntem geliştirilmiştir.

Elde ettiğimiz sonuçlar, yaptığımız sezgisel prosedürlerin performanslarının tatmin edici olduğunu göstermektedirler.

Anahtar Kelimeler: Proje Çizelgeleme, kesikli zaman/maliyet ödünleşimi, yenilenemeyen kaynak.

To my family

ACKNOWLEDGEMENTS

I would like to express my deep and sincere gratitude to my supervisor, Professor Meral Azizoğlu, METU and my co-supervisor, Asst. Professor Ferda Can Çetinkaya, Çankaya University. Their wide knowledge and their logical way of thinking have been of great value for me. Their understanding, encouraging and personal guidance have provided a good basis for my thesis.

I would like to express my sincere appreciation to the examining committee members for their valuable comments.

I would like to thank the Scientific and Technological Research Council of Turkey (TÜBİTAK), for the encouragements and financial support.

I kindly express my special gratitude to TPAO and my colleagues for their kind encouragements, especially Mr. Recep Atalay who supported me throughout my studies and encouraged me.

I am grateful to H. Özgün Polat, who shared his computer skills with me during my thesis and his invaluable help.

I wish to express my warm and sincere gratitude to my dearest friends who were there whenever I needed them and helped me through the difficult times that I encountered whilst completing my thesis.

I owe my loving thanks to my family without whom it would have been impossible for me to precede my work and I am very grateful for their love and moral support they gave me.

TABLE OF CONTENTS

LIST OF TABLES

TABLES

LIST OF FIGURES

FIGURES

CHAPTER 1

INTRODUCTION

A project is a set of interrelated activities to be conducted to achieve a prespecified goal. Project management is planning, organizing, directing, scheduling and controlling the resources, the budget and achieving a satisfactory performance from them. With the current advances in business structures and enterprises, project management concept gained considerable importance in the last decades. Project scheduling defines the start and completion time of the activities with precedence relations and resource constraints. Project scheduling is the core of project management.

In project scheduling, several resources need to be considered. The studies that involve resource are mainly of two types: time/cost trade-off problems and resource constrained problems.

Resource-constrained project scheduling problems apply when there are concerns on the availability of the resources. More resources can be dedicated to the activities in consideration to minimize the total duration of the project. Resource leveling problem arises when the objective is to keep the resource amount at a certain level. Resource allocation minimizes the total duration subject to a limited resource.

In a project there might be many limited resources that are ready to be used; either in *renewable* or *nonrenewable* forms. Manpower, machines, equipments can be

considered as renewable resources which are available at each time period whereas money is a limited nonrenewable resource.

Time/Cost Trade-off models form an important body of the project scheduling problems. These problems do not consider any limit on the availability of the resources; but assume different alternatives for processing an activity. They assume that the time to perform an activity can be reduced if extra cost is paid. These problems are referred to as discrete time/cost trade-off problems if there are a defined number of alternatives to perform each task. These alternatives are defined such that a smaller duration alternative has higher cost.

Time/Cost trade-off problems can be of three types; deadline, budget and curve. The deadline problem aims to minimize the project cost while constrained to finish the project at a certain deadline. The budget problem minimizes the completion time of the project within a given budget. The Time/Cost curve problem considers both time and cost aspects and finds a set of nondominated solutions from a set of feasible time-cost pairs.

In practice, Time/Cost trade-offs and resource constraints may also appear together in the same environment. In this study, we consider such an environment. We assume that there are discrete alternatives to perform each task and there is a single nonrenewable resource that is released at some specified time points.

In project scheduling problems, generally it is assumed that the total budget of the project is given at the beginning in advance or at the end of the project as a total payment. If all budget becomes available at the beginning then the problem reduces to the budget problem. If the total payment is received at the completion then the deadline problem that forces the earliest possible project completion time becomes a good fit. In many practical situations, the payments are received progressively at certain time points, but not entirely at the beginning or end of the project. Generally these payment points are the end of the pre-defined milestone

activities. Also, all projects have budget consumption according to the work that are done by their activities. These budget consumptions should also be optimized to secure the completion of the project i.e., the cost incurred by the activities or total payment to the project should not exceed the total reserved budget.

The time period between two consecutive milestone events is referred to as an interval. The points at each which payments are received are referred to as milestone events. We assume the resources are nonrenewable in the sense that if the amount released at the beginning of the interval is not entirely used in the interval, the unused amount is transferred to the next interval. Hence the unused amounts accumulate, not lost, unlike the renewable resource case. Our aim is to minimize the project completion time in such a way that at each interval total budget available is no less than total budget consumed by the assigned activities. The decision is to select the time/cost pair for each activity, among the select of specified modes.

The rest of the thesis is organized as follows; in Chapter 2, the related literature survey is discussed In Chapter 3, problem definition together with the mathematical model is given. In Chapter 4, modification to the model, its linear programming relaxation and heuristic procedures are discussed. Chapter 5 presents our computational study. In the last chapter, Chapter 6, conclusions and suggestions for further studies are given.

CHAPTER 2

PROJECT SCHEDULING: BASICS AND LITERATURE SURVEY

In this chapter, the basics of the project scheduling will be reviewed and related literature is discussed. We will give the mathematical models for different types of discrete time/cost trade-off and resource- constrained problems.

2.1 Project Management in General

Project Management Institute defines project as a temporary endeavor undertaken to create a product or service**.** Project management is planning, organizing, directing, scheduling and controlling the resources, the budget and achieving a satisfactory performance from them.

2.2 Project Scheduling in General

Project Management means planning, controlling and organizing a project. Project Scheduling is the core of the Project Management and specifies the start and ending times of the activities. The precedence relations, durations, costs are the characteristics of the activities (Fulkerson, 1961, İçmeli *et al.*, 1993). An activity has immediate predecessor(s) which are to be completed before the current could start, similarly, immediate successor(s) which could start only if the current activity is completed (Demeulemeester and Herroelen, 2002, Weglarz, 1998). Project network depicts these precedence relations.

There are two types of networks used to present the activities and precedence relations; AoA (activity on arc) and AoN (activity on node). In AoA representation, activities are represented by arcs and in AoN, activities are represented by nodes. When using an AoA representation for a network, two dummy nodes are needed to initiate and finish the project; i.e. the beginning and the end. For a sample project whose data are given in Table 2.1, we give the AoN and AoA network representations in Figure 2.1 and Figure 2.2 respectively (Elmaghraby, 1970).

Table 2.1 The Activities of a Sample Network

Activity	Duration	Cost	Immediate Predecessor(s)	Immediate Successor(s)
A	2	5	-	B,C
B	4	3	A	D
C	5	3	A	D
D	3	7	B,C	E
E	4	2	D	-

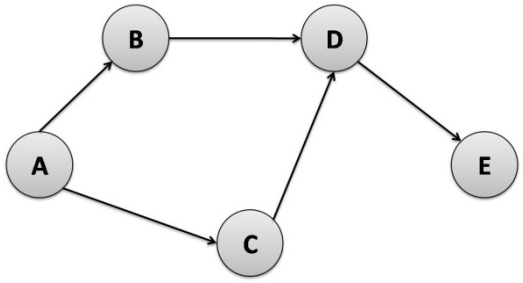

Figure 2.1 AoN Representation of the Network

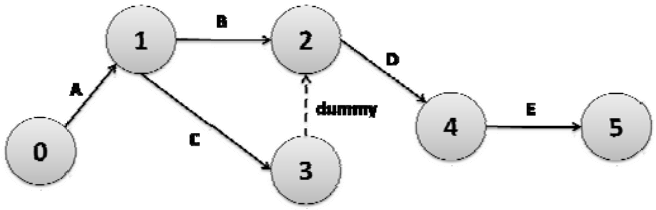

Figure 2.2 AoA Representation of the Network

Note that in Figure 2.2, in AoA representation we have a dummy arc for connecting activity C to D.

2.2.1 Project Scheduling with No Resource Constraints (Time Based)

When there are no resource constraints in a project scheduling problem, then this problem is called as a time based problem and can be solved by using critical path method (CPM) if it is deterministic or by project evaluation review technique (PERT) if it is stochastic.

CPM method is generated to find the best schedule that minimizes the total cost of the project (Moder, *et al*., 1983). A project can have both direct and indirect costs, where direct costs indicate the activity costs and indirect costs indicate the costs incurred throughout the entire project. There are several advantages of CPM which are;

- Planning: Defining the objectives and providing basis to those objectives.
- Communication: Documenting and associating time and cost relations.
- Psychological: Positive outcome to the project team if applied properly.

- Control: Defines the distinctive points in a project and makes it easier to control.
- Training: Gives a good insight for the team and the managers on how to control the project and develop for the other projects.

To perform CPM and PERT, every task should be well-defined with a definite start and ending point. The relations and sequence of the tasks should determined beforehand (Wiest and Levy, 1969).

After determining the characteristics, the longest path is found through the network. This is called the critical path of the network which is made up of critical activities and the completion time of the last activity determines the total length of the project.

2.2.2 Project Scheduling with Resource Constraints

A project scheduling problem with resource constraints is called a Resource-Constrained Project Scheduling Problem (RCPSP). The resource-constrained project scheduling problems, a problem may be activity based or project based (Klein, 1999).

2.2.2.1 Time/Cost Trade-off (Activity Based)

Time/Cost Trade-off Problem can be defined as an activity based resource-constrained project scheduling problem. A time/cost trade-off problem may have either continuous alternatives or discrete alternatives.

2.2.2.1.1 Continuous Alternatives

If a time/cost trade-off problem has discrete alternatives, then there are no single modes of time and cost that can be used directly. Instead a function is specified for the time and cost pairs for a problem.

2.2.2.1.2 Discrete Alternatives

In many real life applications, there are several alternatives to conduct a job in terms of time and cost. In project scheduling literature these problems are referred as time/cost trade-off problems. The alternatives are referred to as modes in discrete time/cost trade-off problems (DTCTP). (De *et al.,* 1995, Kolisch and Padman, 2001, Demeulemeester *et al*., 1996).

Despite its practical importance, the discrete time/cost modes can be used to represent a continuous time/cost curve as mentioned by Elmaghraby (1977).

Table 2.2 tabulates an example of time-cost alternatives for the activities of a project. In this table the second column shows the activity modes. Activity A has three modes with different duration and cost values. In DTCTP, only one mode is to be selected for each activity.

Table 2.2 Sample Modes of Activities in a Project

Activity	Mode	Duration	Cost	Immediate Predecessor(s)	Immediate Successor(s)
A	1	1	10	-	B,C
	2	2	5		
	3	3	4		
	4	5	2		
	5	10	1		
B	1	2	15	A	D
	2	4	3		
	3	6	2		
C	1	1	15	A	D
	2	3	6		
	3	4	4		
	4	5	3		
D	1	3	7	B,C	E
	2	10	2		
E	1	2	8	D	-
	2	4	2		
	3	7	1		

There are three types of discrete time/cost trade-off problems (DTCTP) studied. A project that is to finish within a certain time interval (due date) is said to be a *deadline* problem while minimizing the total cost of the project. On the other hand, a *budget* problem focuses on minimizing the completion time of the project within a given budget. A *time/cost curve* problem in DTCTP problems is simultaneously selecting the non-dominated pairs of time and total cost values.

Below are the mathematical formulations of the problems.

Indices:

i: activity index $\quad i = 1,2, \ldots . , N + 1$

j: mode index $\quad j = 1, \ldots . , m_i$

$N + 1$ is the dummy activity indicating the finish of the project

Parameters:

$m_i =$ number of modes of the activity i

$c_{ij} =$ cost of activity i at mode j

$d_{ij} =$ duration of activity i at mode j

$P_i =$ set of immediate predecessors of activity i

Decision Variables:

$CT_i =$ Completion time of activity i

$$x_{ij} = \begin{cases} 1, & \text{if activity } i \text{ is assigned to mode } j \\ 0, & \text{otherwise} \end{cases}$$

CT_{N+1} = Total project completion time

Constraints:

$$\sum_{j}^{m_i} x_{ij} = 1 \qquad i = 1, \dots, N+1 \qquad (2.1)$$

$$CT_{\boldsymbol{i}} \geq CT_{\boldsymbol{k}} + \sum_{j=1}^{m_i} d_{ij} x_{ij} \qquad i = 1, \dots, N+1 \qquad \forall k \in P_i \qquad (2.2)$$

$$CT_{\boldsymbol{i}} \geq 0 \qquad (2.3)$$

$$x_{ij} \in \{0,1\} \qquad i = 1, \dots, N \;;\; j = 1, \dots, m_i \qquad (2.4)$$

Equation set (*2.1*) indicates that each activity should be conducted at exactly one mode. Second equation set (*2.2*) gives the immediate precedence relations of the activities given in set P_i, i.e. activity *k*'s immediate successor is activity *i*. The next equation set (*2.3*) states that the completion time of the activities is nonnegative. x_{ij} is the binary variable of the model.

There exists $\sum_{i=1}^{N} m_i$ number of x_{ij} binary variables, $N+1$ number of CT_i variables. Totally there are $N + \sum_{i=1}^{N} |P_i|$ number of constraints.

Pre-mentioned types of DTCTP's are based on the above decision variables and constraints. Below are the formulations of these problems.

Deadline Problem:

The deadline problem minimizes the total cost of the project subject to a given deadline.

The additional constraint set states that the total project completion time cannot exceed the deadline of the project (*T*) (Equation (*2.5*)).

$$CT_{N+1} \leq T \qquad (2.5)$$

The objective function given in (2.6) minimizes the total cost.

$$\sum_{i=1}^{N} \sum_{j=1}^{m_i} c_{ij} x_{ij} \qquad (2.6)$$

Demeulemeester (1998) and Hafızoğlu and Azizoğlu (2008) are some researchers that study the deadline problem.

Mathematical Model of Deadline Problem

$$min \ \sum_{i=1}^{N} \sum_{j=1}^{m_i} c_{ij} x_{ij}$$

Subject to (2.1), (2.2), (2.3), (2.4), (2.5).

Budget Problem

The budget problem minimizes the completion time of the project subject to a given budget.

The additional constraint set states that the cost incurred by all activities cannot exceed the budget of the project (*B*) (Equation *2.7*).

$$\sum_{i=1}^{N} \sum_{j=1}^{m_i} c_{ij} x_{ij} \leq B \qquad (2.7)$$

The objective function of the budget problem is given in (*2.8*) and minimizes the completion time of the project.

$$min \ CT_{N+1} \qquad (2.8)$$

Hazır *et al.* (2009) and Değirmenci and Azizoğlu (2008) are some researchers that study the budget problem.

Mathematical Model of Deadline Problem

$min \quad CT_{N+1}$

Subject to (*2.1*), (*2.2*), (*2.3*), (*2.4*), (*2.7*).

Time/Cost Curve Problem

Time/Cost Curve Problem finds the set of non-dominated solutions with respect to total cost and completion time criteria. A solution *S* is said to be non-dominated if there does not exist any other solution *S'* such that $C_{N+1}(S') \leq C_{N+1}(S)$ and $\sum\sum c_{ij}x_{ij}(S') \leq \sum\sum c_{ij}x_{ij}(S)$ with strict inequality holding at least once. Finding the time/cost curve is important as the optimal solution of a nondecreasing function of total cost and completion time is in the nondominated solution set.

Demeulemeester (1998) use the deadline problem to construct the time/cost trade-off curve. Alternatively the budget problem can be used to construct this curve.

2.2.2.2 Resource Constrained Project Scheduling Problem

Several approaches like linear and integer programming, dynamic programming, implicit enumeration, bounded enumeration and heuristic programming are used to solve resource-constrained project scheduling problems *(RCPSP)* (Herroelen, 1972).

The integer programming (0-1) model that was suggested by Gonguet minimizes the completion time of the project where the binary decision variables control the start time of each activity. At each time interval, demand of each type of resource should not exceed its available amount.

Pritsker *et al.* (1969) allow preemptions and assume different arrival times and due dates for the activities. They propose integer programming (0-1) model for three possible objective functions; minimizing the completion time, minimizing total lateness and minimizing the lateness penalty. According to their model, the

time periods are determined based on the activities' arrival times, precedence relations and due dates.

Herroelen *et al.* (1998) state that in most of the cases the objective in project scheduling problems is minimizing the completion time of the project, which is a *regular* objective function. However in recent years, focus is shifted to non-regular objective functions such as maximizing the net present value (NPV). A non-regular objective function can be described as a function that does not worsen even if there is a delay on some activities. Maximizing NPV is also called *payment scheduling problem* in the literature. There is no constraint regarding resource usage and activities have fixed durations which involve cash flow payments and receipts.

In discrete time/resource trade-off problems, it is assumed that all the activities are conducted on one single mode and on one single time period with a restriction of *renewable* resource(s) in a single period and subject to precedence relations. Below is the formulation of the discrete time/resource trade-off resource-constrained problem. (Herroelen *et al.*,1998)

Indices:

i: activity index $\quad i = 1,2, \ldots, n$

j: mode index $\quad j = 1, \ldots, m_i$

t: time index $\quad t = 0,1, \ldots, T$

$N + 1$ is the dummy activity indicating the finish of the project

Parameters:

$m_i =$ number of modes of the activity i

r_{ij} = resource usage of activity i at mode j

d_{ij} = duration of activity i at mode j

P_i = $s$et of immediate predecessors of activity i

e_i = critical path based earliest start time of activity

i based on the modes with the smallest duration.

l_i = critical path based latest start time of

activity i based on the modes with the smallest duration

a = constant renewable resource availability per period

Decision Variables:

$$x_{ijt} = \begin{cases} 1, & \text{if activity } i \text{ is assigned to mode } j \text{ started at time } t \\ 0, & \text{otherwise} \end{cases}$$

Mathematical Model:

$$Min \sum_{t=e_n}^{l_n} t\, x_{n1t} \qquad (2.9)$$

Subject to

$$\sum_{j=1}^{m_i} \sum_{t=e_i}^{l_i} x_{ijt} = 1 \qquad (2.10)$$

$$\sum_{j=1}^{m_i} \sum_{t=e_i}^{l_i} (t + d_{ij}) x_{ijt} \leq \sum_{j=1}^{m_k} \sum_{t=e_k}^{l_k} t x_{kjt} \qquad \forall k \in P_i \qquad (2.11)$$

$$\sum_{i=1}^{N+1} \sum_{j=1}^{m_i} r_{ij} \sum_{s=max(t-d_{ij},e_i)}^{min\,(t-1,l_i)} x_{ijs} \leq a \qquad (2.12)$$

$x_{ijt} \in \{0,1\}$ (2.13)

Equation set (*2.9*) minimizes the completion time of the project subject to equation sets (*2.10*), (*2.11*), (*2.12*) and (*2.13*) which assigns activities to one mode at one time, arranges precedence relations of activities, maintains the renewable resource usage per period of time and forces the decision variables to be binary, respectively.

The above formulation can be extended by adding nonrenewable and doubly-constrained resource constraints with different types of objective functions. When a resource constrained project scheduling problem involves multiple renewable, nonrenewable and doubly-constrained resource restrictions with trade-offs, then the associated problem is called *Multi Mode Resource-Constrained Project Scheduling Problem (MRCPSP)* (Herroelen *et al.*, 1998).

In 1982, Talbot proposed a non-preemptive case for RCPSP with time/cost trade-offs. In non-preemptive cases, an activity in the whole project cannot be split and proceed by the successor activity. Talbot (1982) uses two types of objectives; minimizing the project completion time and minimizing the overall project costs. Three types of resources are considered in this model; renewable, nonrenewable and doubly-constrained. Here renewable and nonrenewable resources are available to a certain extent for the whole project. Below is the formulation of the model:

Indices:

i: activity index $i = 1,2, \ldots, n$

j: mode index $j = 1, \ldots, m_i$

t: time index $t = 0,1, \ldots, T$

Parameters:

$m_i =$ number of modes of the activity i

$r_{ijr} =$ renewable resource r usage of activity i at mode j

$w_{ijn} =$ nonrenewable resource n usage of activity i at mode j

$d_{ij} =$ duration of activity i at mode j

$P_i =$ set of immediate predecessors of activity i

$e_i =$ critical path based earliest start time of activity i based on the modes with the smallest duration.

$l_i =$ critical path based latest start time of activity i based on the modes with the smallest duration

$R_{rt} =$ available renewable resource of r at time t

$W_n =$ available nonrenewable resource of n at time t

Decision Variables:

$$x_{ijt} = \begin{cases} 1, & \text{if activity } i \text{ is assigned to mode } j \text{ started at time } t \\ 0, & \text{otherwise} \end{cases}$$

Mathematical Model:

$$Min \sum_{j=1}^{m_n} \sum_{t=e_n}^{l_n} t\, x_{njt} \qquad (2.14)$$

Subject to

$$\sum_{j=1}^{m_i} \sum_{t=e_i}^{l_i} x_{ijt} = 1 \tag{2.15}$$

$$-\sum_{j=1}^{m_i} \sum_{t=e_i}^{l_i} t x_{ijt} + \sum_{j=1}^{m_k} \sum_{t=e_k}^{l_k} (t - d_{kj}) x_{kjt} \qquad \forall k \in P_i \tag{2.16}$$

$$\sum_{i=1}^{N} \sum_{j=1}^{m_i} \sum_{q=t}^{t+d_{ij}-1} r_{ijr} x_{ijq} \leq R_{rt} \tag{2.17}$$

$$\sum_{i=1}^{N} \sum_{j=1}^{m_i} \sum_{t=e_i}^{l_i} w_{ijn} x_{ijt} \leq W_n \tag{2.18}$$

$$x_{ijt} \in \{0,1\} \tag{2.19}$$

The objective function minimizes the project completion time as stated in Equation set (*2.14*). Equation set (*2.15*) ensures that an activity is completed once at one time in one mode. The next equation set (*2.16*) gives the precedence relations of the activities. Resource constraints are given by the equation sets (*2.17*) and (*2.18*). The first one, regarding renewable resources ensures the availability of the resources at each time period. The second equation which considers nonrenewable resources constrains the limit of the resource for the entire project. Doubly constrained resources may be given as a combination of both resource constraints i.e., $r_{ijr} = w_{ijn}/d_{ij}$ where *r* and *n* can be considered as the same type of resources. Equation set (*2.19*) forces the decision variables to be binary.

2.3 Most Closely Related Problems

Nonrenewable Resource Constraints

Carlier and Rinooy Kan (1982) show that a single nonrenewable resource problem with the following settings is polynomially solvable. The single resource is released at prespecified time points at prespecified quantities.

Hence our problem with fixed modes is polynomially solvable.

Budget Problem

When all units of the single resources are released at time zero at prespecified quantity B, our problem reduces to the budget problem in discrete time/cost trade-off scheduling.

CHAPTER 3

THE MODEL

In this chapter, firstly we define the problem. Then the chapter continues with the construction of the model for our problem.

3.1 Problem Definition

In the literature, mostly renewable resources are used as the resource type, which includes manpower, equipments or machines. When nonrenewable resources are considered, like the capital budget, it is assumed that the resource is available as a lump sum in the lifetime of the project. Thus theoretically, it is assumed that the total budget of a project is available either in the beginning or at the end of the project. However, this is not common in practice. Mostly, dedicated budget is released at certain time intervals corresponding to the either completion or beginning of an activity or regular time periods. These certain time instants that the budget is released are said to be *milestones*. Thus at each milestone event, there is resource inflow.

If the budget releases are at time instants t_1, t_2, and t_3, then the budget released at t_2 covers the costs of the completed activities in interval $(t_1, t_2]$.

In the model, only one nonrenewable resource, say budget, is used. After each time instant if the total budget released is not entirely used by the activities, then the remaining amount is transferred to the next time interval to be used.

In our model, we assume a cost is incurred whenever an activity is complete and money is received at certain time instants, called milestones.

The model constructed takes the mode selection decisions into account in terms of time/cost trade-off. There are several modes with different time and cost pairs. We assume the activities can be shortened whenever additional money is spent.

In this model it is important to define the time intervals, as the activity selections are also based on time, *t*. Completion time of an activity should lay between two or more consecutive time instants.

In the following section, mathematical representation of the problem described above is given.

3.2 The Model

We first define the indices, parameters, decision variables and constraints.

Indices:

i: activity index

j: mode index

t: time index

Parameters:

$n =$ number of activities in the project

$m_i =$ number of modes of the activity i $\qquad i = 1,2, \dots, n+1$

$TI =$ number of time instants at which money is released

c_{ij} = cost of activity i at mode j

$i = 1,2, \ldots, n+1, \quad j = 1, \ldots, m_i$

p_{ij} = processing time of activity i at mode j

Note that $P_{ij_1} > P_{ij_2}$ implies $c_{ij_1} < c_{ij_2}$.

B_t = Budget released at time t $\quad t = 0,1, \ldots, T$

P_i = set of immediate predecessors of activity i $\quad i = 1,2, \ldots, n+1$

If activity k is immediate predecessor of activity i, then P_i includes k.

I_t = time instant I at which t$^{\text{th}}$ payment is recieved $\quad t = 0,1, \ldots, T$

Decision Variables:

$$x_{ijt} = \begin{cases} 1, & \text{if activity } i \text{ is assigned to mode } j \text{ started at time } t \\ 0, & \text{otherwise} \end{cases}$$

$CT_i =$ completion time of activity i

$k \rightarrow i$ activity i is successor of k

Mathematical Model:

$$Min\ CT_{n+1} \tag{3.1}$$

$$\sum_{j=1}^{m_i} \sum_{t=1}^{T} x_{ijt} = 1 \qquad \forall i \tag{3.2}$$

$$CT_i \geq CT_k + \sum_{j=1}^{m_i} \sum_{t=1}^{T} p_{ij} x_{ijt} \qquad \forall i, k \in P_i \tag{3.3}$$

$$\sum_{i=1}^{n} \sum_{j=1}^{m_i} \sum_{t=1}^{\tau} c_{ij} x_{ijt} \leq \sum_{t=1}^{\tau} B_t \qquad \tau = 1 \tag{3.4}$$

$$CT_i \leq \sum_{j=1}^{m_i} \sum_{t=1}^{T} I_t x_{ijt} \quad (3.5)$$

$$CT_i \geq \sum_{j=1}^{m_i} \sum_{t=1}^{T} (I_{t-1} + 1)\, x_{ijt} \quad (3.6)$$

$$CT_0 = 0 \quad (3.7)$$

$$CT_i \geq 0 \quad (3.8)$$

$$x_{ijt} \geq 0 \text{ and integer} \quad (3.9)$$

The model has three indices; *i, j* and *t,* which identifies activities $i = 0,1, \dots\dots, n+1$; modes $j = 1, \dots\dots, m_i$ and time interval $t = 0, \dots\dots, T$.

There are *n* activities with one sink (dummy) node, $n+1$, at the end to identify the project completion time. The number of modes is limited to the activities' maximum mode number, m_i.Time index *t*, starts from *0* to *T* where *T* is the last time interval's end.

Minimizing the total completion time is the objective which Equation (*3.1*) below indicates.

There are several constraints to be satisfied while minimizing the project duration. Equation set (*3.2*) ensures that each activity *i* is conducted exactly at one mode and at one time interval.

Constraint set (*3.3*) satisfies the precedence relations. The completion time of activity *i* is no smaller the processing time of activity *i* added to the predecessor activity *k*'s completion time, i.e., the start time of a successor activity cannot be greater than the completion time of the predecessor.

Constraint set (*3.4*) guarantees that the total amount needed by all activities that are assigned to intervals *1,…,τ* cannot exceed to the total amount released at time intervals *1,…,τ*.

Constraint sets (*3.5*) and (*3.6*) define the interval at which activity i is completed.

Constraint (*3.7*) states that the project starts at time zero. That is the completion time of the dummy activity is zero.

Constraint set (*3.8*) is sign constraint. However the set is redundant as there is a single source activity with start time of zero.

Constraint set (*3.9*) is for binary variables. Note that we do not include the upper bound on x_{ijt} values, as equation set (*3.2*) satisfies $x_{ijt} \leq 1$ relation.

CHAPTER 4

SOLUTION PROCEDURES

In this chapter, we first discuss our lower bounding procedures and then present the heuristic procedures. All our procedures are based on linear programming relaxation (LPR) of the original model.

4.1 Lower Bounds

We develop two lower bounding procedures one is based on pure LP relaxation and the other one is based on strengthened LP relaxation of the model. The latter model has integer variables, but fewer in the number, when compared with the original model.

4.1.1 Lower Bound 1

We simply relax the integrality constraints on the x_{ijt} values and solve the resulting linear program. In doing so, we replace $x_{ijt} \in \{0, 1\}$ with constraints $x_{ijt} \leq 1$ and $x_{ijt} \geq 0$ for all *i ,j, t*. The optimal project completion time value of the resulting LPR model is a lower bound on the optimal project completion time. We let this lower bound be LB_1.

In the LB_1 solution, one or both of the following cases may be observed.

Case 1. An activity may be assigned to more than one period, i.e., $x_{ijt} > 0$ for more than one *t*.

Case 2. An activity can be assigned to more than one mode, i.e., $x_{ijt} > 0$ for more than one *j*.

It is possible to strengthen the lower bound, by preventing the occurrence of any one of the cases. For example the occurrence of Case 1 can be avoided by imposing a constraint that forces the completion of each activity to be in exactly one period. We next explain our strengthened lower bound that bases on this imposed constraint.

4.1.2 Lower Bound 2

Consider the following decision variable:

$$y_{it} = \begin{cases} 1, & \text{if activity } i \text{ is completed in period } t \\ 0, & \text{otherwise} \end{cases}$$

The following two constraint sets are included to the LP relaxation together with new decision variables y_{it}s.

$$\sum_t y_{it} = 1 \qquad \forall i \qquad (4.1)$$

$$\sum_j x_{ijt} = y_{it} \qquad \forall i, t \qquad (4.2)$$

Constraint set (*4.1*) requires each activity's completion is assigned to at most one period. Constraint set (*4.2*) relates the continuous x_{ijt} values to new integer variables y_{it}'s.

The resulting model is a mixed integer linear program (MILP). We let the optimal completion time of the MILP be LB_2. Note that LB_2 provides a lower bound on the optimal project completion as it does not avoid multimode assignment i.e.,

Case 2 can still occur. However, it is easier to solve as it includes much fewer binary decision variables due to the fact that there are $n \times T$ binary variables y_{it}'s in MILP, and there are $(\sum_{i=1}^{n} m_i) \times T$ binary variables x_{ijt}'s in the original model.

$LB_2 \geq LB_1$ as it puts an additional restriction to the linear program. However it is harder to solve the mixed integer linear program as it includes binary decision variables.

Alternately, Case 2 might be avoided by introducing the following decision variables and constraint sets (*4.3*) and (*4.4*).

$$z_{it} = \begin{cases} 1, & \text{if activity } i \text{ is assigned to its } j^{th} \text{ mode} \\ 0, & \text{otherwise} \end{cases}$$

where $i = 1,\ldots, n$ and $j = 1,\ldots\ldots, m_i$

$$\sum_{j=1}^{m_i} z_{ij} = 1 \qquad \forall i \qquad (4.3)$$

$$\sum_{t} x_{ijt} = z_{ij} \qquad \forall i, j \qquad (4.4)$$

This model requires $\sum_{i=1}^{n} m_i$ binary decision variables which is much more than $n \times T$ in particular when $n>T$.

We did not try for this alternative, as our experiments have shown that the MILP to find LB_2 results with many binary variables and few continuous variables, hence there are too few mode splitting.

4.2 Heuristic Procedures

In this section, we propose three heuristic procedures, i.e., upper bounds. All upper bounds use the optimal solutions of the MILP that allows mode splitting. The first heuristic makes the mode assignments using the partial or full mode

assignments made by MILP. Using these modes it finds an optimal solution considering the budget releases and usages. The second heuristic takes the full mode assignments from the MILP solutions and fixes these activities to the modes assigned by the MILP. The unassigned modes are the ones that receive partial assignments by the MILP. These mode assignments are done optimally by using the original IP. This step is done in exponential time, however, at reduced problem. The third heuristic uses pure LP relaxation results to make partial mode assignments. Using new modes it finds an optimal solution considering the budget releases and usages.

Below are the detailed descriptions of the heuristic procedures.

4.2.1 Heuristic 1

If the mode assignments are known, the budget allocation problem reduces to the nonrenewable resource constrained time problem. This problem can be solved in polynomial time using the algorithm proposed by Carlier and Rinooy Kan (1982). For the sake of completeness we state the steps of this polynomial time algorithm. We need the following notation to state the algorithm.

P_i = time to perform activity *i* at fixed mode.

A_t= total amount of resource released till the end of the period *t*.

Resource release profile plots *t* versus A_t.

R_t = total amount of resource required till the end of period *t* when the activities are scheduled according to the latest start schedule.

Resource requirement profile plots *t* versus R_t.

Algorithm Y:

Step 1. If $R_t \leq A_t$ for all periods *t*, then the resource requirement profile is beneath the resource release profile, and then stop. The latest start schedule is optimal.

Step 2. Shift the resource requirement profile to the right until all points of the profile lie on or below the resource release profile.

Shifting the line is equivalent to increasing the project completion time. With minimum shift, the project completion time is minimized subject to budget constraints.

Our heuristic procedure sets the activity modes using the optimal MILP solution. We mention the fractional mode assignments such that the budget constraint is not violated. Hence, we increase the fractional processing time till it fits to defined mode. We make the increase as small as possible to favor our objective of minimizing the total project completion time. We take the full mode assignments, if any made by MILP. After all activities are set to one mode, we apply Algorithm Y to find the optimal completion times subject to given mode assignments.

After we obtain a schedule by implementing Algorithm Y, we shorten the project duration by reducing the task times of some activities. The activities that could reduce the project duration are the critical ones that are not already assigned to their shortest duration mode and that would not violate the budget constraints when assigned to its next shorter mode. We let the set of such activities as Set CP. We select the activity having the longest completion time in Set CP and assign it to its next shorter mode. After the reassignment, we shift the other activities to the left of the schedule; therefore reduce their completion times, as long as budget constraint permits. We stop when there is no further room for improvement, i.e., Set CP is empty or when the new schedule's total completion time is found out to be higher than that of original schedule's.

Below is the algorithmic description of our first heuristic procedure.

Step 1. Let t_{iLP} be the processing time of activity i returned by the MILP.

Set activity i to mode j_i such that $t_{i,j_i-1} < t_{iLP} \leq t_{ij_i}$.

i.e., the mode having the smallest processing time that is no smaller than t_{iLP}.

Step 2. Let $t_i = t_{ij_i} \quad \forall_i$

Find the project completion time using Algorithm Y with fixed t_i values.

Step 3. Define the critical activities.

Let CP be the set of critical activities whose durations can be reduced without violating the budget constraint and increasing the project duration.

Stop if CP is empty.

Step 4. Let r be the task in CP that has the highest completion time.

Let j_r be the current mode of task r. Assign task r to its next shortest duration mode, i.e., mode j_r+1.

Shift all activities that follow, i.e., all successors of task r to the right of the schedule as much as possible (as long as budget constraint permits).

Go to Step 3.

In Step 1, we solve the MILP heuristically. The heuristic solution to MILP is found by giving 10% optimality gap to our IP-solver. We use 10% gap, as our initial experimentation has revealed that with this gap we still find satisfactory solutions for our heuristics as we observe that attaining optimal solutions is not easy. Moreover, optimal solutions are not essential as they are eventually used to produce approximate solutions.

Step 2 runs in polynomial time as Algorithm Y is polynomial-time. The improvement steps, i.e., Steps 3 and 4 iterate at most $\sum_j (m_j - 1)$ times, as at worst case all activities may be at their longest modes and can be shifted to their shortest modes. Hence, the improvement step runs in polynomial time.

4.2.2 Heuristic 2

The heuristic runs in two steps. In the first step, the MILP with mode splitting is solved and the full mode assignments are fixed. In the second step, the mode assignments (that are partial in the MILP solutions) are made optimally using the IP. Below is the stepwise description of Heuristic 2.

Algorithm X:

Step 1. Solve the MILP, by allowing α % gap allowance

Let S_{LP} be the set of activities that are fully assigned to a single mode. Let k_i be the mode index at which $x_{ik_i} = 1$ for $i \in S_{LP}$.

Step 2. Set $\sum_t x_{ik_{it}} = 1$ for each $i \in S_{LP}$ and find the mode assignments for each $i \notin S_{LP}$ by solving the original IP.

Heuristic 2, without improvement steps (i.e., Steps 3 and 4) dominate Heuristic 1 as it does the mode assignments for all $i \notin S_{LP}$ optimally whereas Heuristic 1

does those assignments heuristically. In other words, Heuristic 1 pre-sets the mode assignments whereas Heuristic 2 decides on their values.

After the improvement steps, there is no domination between Heuristic 1 and Heuristic 3.

4.2.3 Heuristic 3

Note that Heuristics 1 and 2 run in exponential time. A polynomial time heuristic is also generated by implementing Heuristic 1 with pure linear program. In doing so, the t_i values are found from LP using the found expression.

$$t_i = \sum_{t=1}^{T} \sum_{j=1}^{m_i} t_{ij} x_{ijt} \quad \forall i$$

where x_{ijt} is the optimal values of the assignment variables found by LP.

For the sake of completeness, we give the stepwise description of Heuristic 3.

Algorithm Z:

Step 1. Let t_{iLP} be the processing time of activity *i* returned by the LP.

Set activity *i* to mode j_i such that

$$t_{ij-1} < t_{iLP} \leq t_{ij_i}$$

i.e., to a mode having smallest processing time that is no smaller than t_{iLP}.

Step 2. Let $t_i = t_{ij_i} \quad \forall i$

Using t_i values, find the project completion time using Algorithm Y.

Step 3. Define the critical activities.

Let CP be the set of critical activities whose durations can be reduced without violating the budget constraint and increasing the project duration.

Stop if CP is empty.

Step 4. Let r be the task in CP that has the highest completion time.

Let j_r be the current mode of task r. Assign task r to its next shortest duration mode, i.e., mode j_r+1.

Shift all activities that follow, i.e., all successors of task r to the right of the schedule as much as possible (as long as budget constraint permits).

Go to Step 3.

Note that all steps of the heuristic run in polynomial time unlike Heuristic 2 that runs in exponential time.

Figure 4-1 demonstrates the data flowchart of the solution procedures described above.

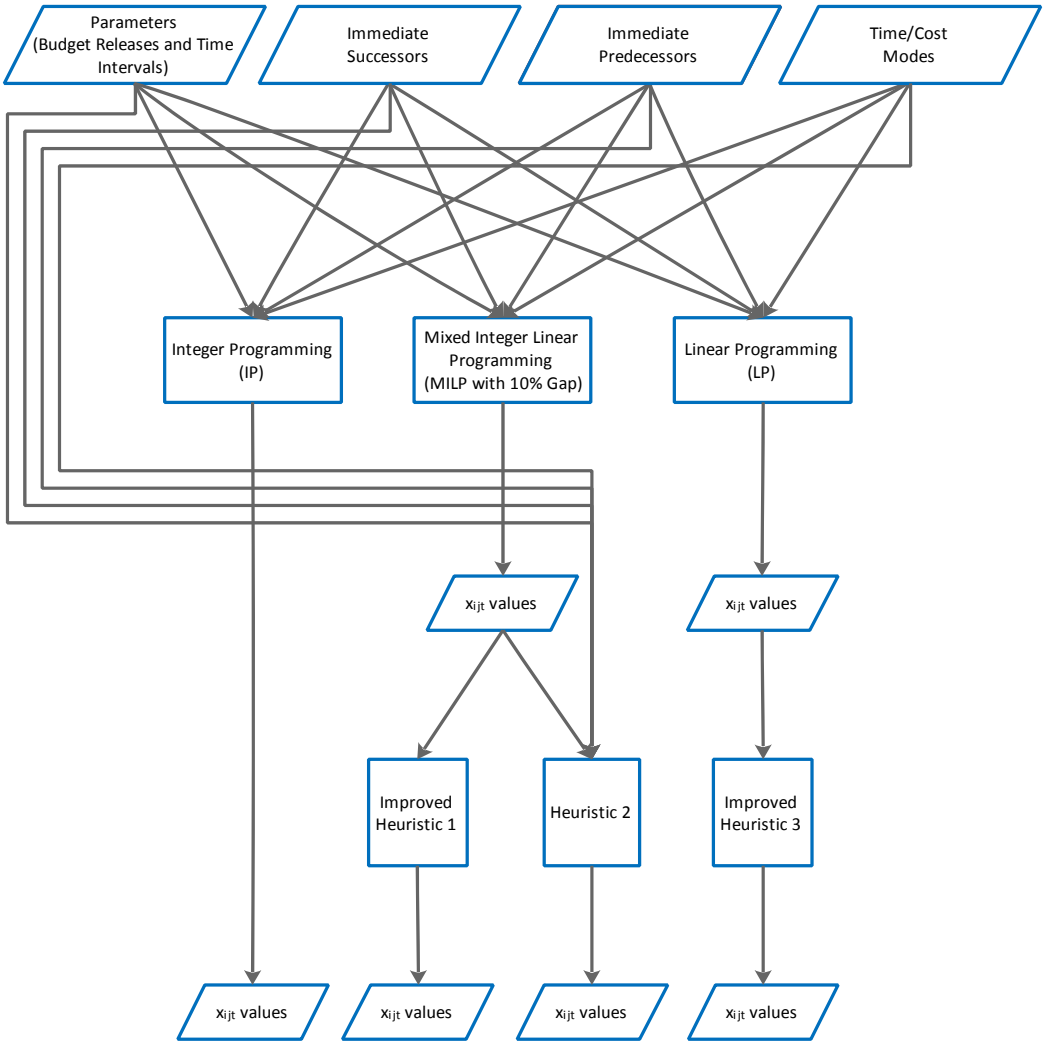

Figure 4.1 Flowchart Diagram of Data Flow and Solution Procedures

4.3 An Illustrative Example

In this section, an illustrative example will be solved to demonstrate the solution procedures.

4.3.1 Problem Data

Our example problem (P) has 15 activities with alternate time/cost modes. Table 4.1 shows the predecessor – successor relations of P, and Figure 4.1 shows the associated network.

Table 4.1 Predecessor and Successor Relations of P

Activity	Immediate Predecessor(s)	Immediate Successor(s)
1	-	3,4,5
2	-	6,7
3	1	9,10
4	1	13
5	1	8
6	2	12
7	2	11,14
8	5	13
9	3	13
10	3	15
11	7	12
12	6,11	15
13	4,8,9	15
14	7	15
15	10,12,13,14	-

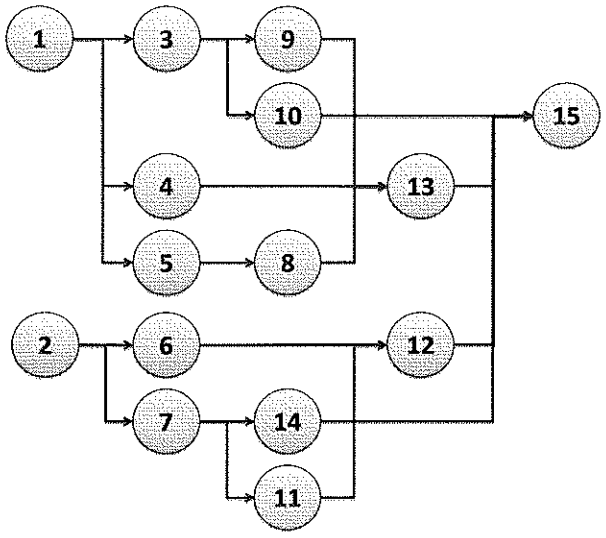

Figure 4.2 Network Diagram of P

Each activity has at most 9 modes and least 1 mode. Tables 4.2 and 4.3 tabulate the times and costs for the activity modes.

Table 4.2 Times for the Activity Modes of P

Activity	Mode 1	Mode 2	Mode 3	Mode 4	Mode 5	Mode 6	Mode 7	Mode 8	Mode 9
1	16	13	12	11	10	7	4	2	-
2	13	10	7	5	3	-	-	-	-
3	7	4	3	-	-	-	-	-	-
4	13	10	7	5	3	-	-	-	-
5	17	15	13	10	8	7	5	3	-
6	11	10	7	6	4	1	-	-	-
7	10	7	5	2	-	-	-	-	-
8	12	10	7	6	4	1	-	-	-
9	15	14	11	8	5	4	2	-	-
10	7	4	3	-	-	-	-	-	-
11	4	1	-	-	-	-	-	-	-
12	17	15	14	11	8	5	2	-	-
13	2	-	-	-	-	-	-	-	-
14	19	17	15	12	10	7	6	3	-
15	21	18	15	14	11	8	5	4	2

Table 4.3 Costs for Activity Modes of P

Activity	Mode 1	Mode 2	Mode 3	Mode 4	Mode 5	Mode 6	Mode 7	Mode 8	Mode 9
1	2	13	16	17	31	35	47	70	-
2	6	38	45	52	96	-	-	-	-
3	38	71	85	-	-	-	-	-	-
4	6	38	45	52	96	-	-	-	-
5	10	24	28	39	40	77	94	98	-
6	7	19	35	57	67	71	-	-	-
7	15	16	25	75	-	-	-	-	-
8	11	27	35	65	81	90	-	-	-
9	13	29	41	54	75	77	97	-	-
10	28	71	85	-	-	-	-	-	-
11	52	73	-	-	-	-	-	-	-
12	22	27	30	35	36	37	99	-	-
13	32	-	-	-	-	-	-	-	-
14	39	55	57	63	69	72	86	97	-
15	8	19	22	30	33	42	62	73	76

The problem is solved four times for the following cases which are generated randomly:

Case 1. Narrow time interval, low budget

Case 2. Narrow time interval, high budget

Case 3. Wide time interval, high budget

Case 4. Wide time interval, low budget

Table 4.4 tabulates the values used in each of the four cases showing time interval and budget release values

Table 4.4 Parameters of P

Case 1	time intervals	budget release
t1	4	77
t2	28	85
t3	35	79
t4	42	89
t5	60	61

Case 3	time intervals	budget release
t1	8	154
t2	56	170
t3	70	158
t4	84	178
t5	120	122

Case 2	time intervals	budget release
t1	4	154
t2	28	170
t3	35	158
t4	42	178
t5	60	122

Case 4	time intervals	budget release
t1	8	77
t2	56	85
t3	70	79
t4	84	89
t5	120	61

4.3.2 Solution Procedures

The problem is solved firstly as an integer programming, and then its LP relaxation is solved. After that, the MILP model that requires the execution of the activity within the same time interval is solved.

Our first heuristic procedure takes the solutions of MILP and allocates the nonrenewable resource based on the activity durations found by the MILP.

Our second heuristic procedure takes the integer variables of the optimal mixed integer program, and solved the IP for the continuous variables.

Third heuristic uses the same logic with Heuristic 1, where instead of the mixed integer program's results; it uses the LP's.

4.3.3 Results

We first solve the IP. The optimal solution for each case found by IP solution is given in Table 4.5.

Table 4.5 IP Solution of P

IP solution	
	z*
Case 1	43
Case 2	31
Case 3	55
Case 4	79

Then, solve the MILP. The optimal solution (0% gap) for each case is given in Table 4.6 together with the # of fractional activities and # of integers.

Table 4.6 MILP Solution of P

MILP		
z	# of fractional activities	# of integers
43	4	11
31	3	12
54.667	1	14
77.765	1	14

Note that according to this solution, the following activities are to be fixed (Table 4.7).

Table 4.7 Fixed Activities of MILP

Case 1		
Activity	Mode	Time
2	1	2
3	1	2
4	1	3
6	1	3
8	1	4
9	1	3
10	1	4
11	1	3
13	1	4
14	1	4
15	9	5

Case 3		
Activity	Mode	Time
2	1	2
3	1	2
4	1	2
5	1	2
6	1	2
7	1	2
8	1	2
9	1	2
10	1	2
11	1	2
12	1	2
13	1	2
14	1	2
15	6	2

Case 2		
Activity	Mode	Time
2	1	2
3	1	2
4	1	2
6	1	2
7	3	2
8	1	2
9	1	2
10	1	3
11	1	2
12	6	3
13	1	3
15	9	3

Case 4		
Activity	Mode	Time
1	1	2
2	1	2
3	1	2
4	1	2
5	1	3
6	1	2
7	1	2
8	1	3
9	1	2
10	1	3
11	1	2
12	1	2
13	1	4
14	1	4

The remaining activities are set to the following modes when implementing Heuristic 1 (Table 4.8).

Table 4.8 Assigned Modes of the Remaining Activities

Case 1			
Activity	Cost	Duration	Chosen Mode
1	13	13	2
5	24	15	2
7	16	7	2
12	30	14	3

Case 3			
Activity	Cost	Duration	Chosen Mode
1	2	16	1

Case 2			
Activity	Cost	Duration	Chosen Mode
1	47	4	7
5	28	13	3
14	69	10	5

Case 4			
Activity	Cost	Duration	Chosen Mode
15	42	8	6

Hence the problem is reduced to the single mode nonrenewable resource allocation problem. Shifting algorithm (Heuristic 1) produces the following task completion times (Table 4.9).

Table 4.9 Completion Times of the Activities by Heuristic 1

Case 1		Case 2		Case 3		Case 4	
Activity	Late Finish	Activity	Late Finish	Activity	Late Finish	Activity	Late Finish
1	14	1	5	1	16	1	40
2	17	2	16	2	16	2	40
3	26	3	15	3	30	3	54
4	41	4	30	4	45	4	69
5	29	5	18	5	33	5	57
6	29	6	27	6	30	6	54
7	24	7	22	7	26	7	50
8	41	8	30	8	45	8	69
9	41	9	30	9	45	9	69
10	43	10	32	10	47	10	71
11	29	11	27	11	30	11	54
12	43	12	32	12	47	12	71
13	43	13	32	13	47	13	71
14	43	14	32	14	47	14	71
15	45	15	34	15	55	15	79

The total completion time of the project is given below for each case found by implementing Heuristic 1 (Table 4.10).

Table 4.10 Completion Times of the Project by Heuristic 1

Cases	Total Completion Time
Case 1	45
Case 2	34
Case 3	55
Case 4	79

When implementing Heuristic 2, solution of the MILP is used to find the project completion time. Like Heuristic 1, binary activities found by MILP are fixed and remaining activities' modes are found optimally by solving IP. The completion time of the project found by Heuristic 2 is given below (Table 4.11).

Table 4.11 Completion Times of the Project by Heuristic 2

Cases	Total Completion Time
Case 1	43
Case 2	32
Case 3	55
Case 4	79

Heuristic 3 uses LP relaxation solutions to implement the shifting algorithm. Table 4.12 shows the completion times of the activities produced by Heuristic 3.

Table 4.12 Completion Times of the Activities by Heuristic 3

Case 1		Case 2		Case 3		Case 4	
Activity	Latest Finish	Activity	Latest Finish	Activity	Latest Finish	Activity	Latest Finish
1	32	1	24	1	49	1	90
2	33	2	26	2	54	2	88
3	42	3	31	3	58	3	101
4	57	4	39	4	69	4	115
5	45	5	33	5	59	5	105
6	45	6	36	6	66	6	106
7	40	7	32	7	61	7	98
8	57	8	39	8	69	8	115
9	57	9	39	9	69	9	115
10	59	10	41	10	71	10	117
11	45	11	36	11	66	11	106
12	59	12	41	12	71	12	117
13	59	13	41	13	71	13	117
14	59	14	41	14	71	14	117
15	61	15	43	15	73	15	121

The total completion times of the project found by Heuristic 3 for all cases are given below (Table 4.13).

Table 4.13 Completion Times of the Project by Heuristic 3

Cases	Total Completion Time
Case 1	61
Case 2	43
Case 3	73
Case 4	121

The task completion times are then reduced to the following values by the Improvement Algorithm (Table 4.14).

Table 4.14 Completion Times of the Heuristics after Improvement Algorithm

Cases	Improved Heuristic 1	Heuristic 2 (10% Gap)	Improved Heuristic 3
Case 1	45	45	61
Case 2	34	32	43
Case 3	55	55	73
Case 4	79	79	121

Since the number of activities is very small, it is found out that the solutions are already giving the best possible results, so the effect of the improvement algorithm cannot be observed from this illustrative example.

CHAPTER 5

COMPUTATIONAL RESULTS

In this chapter, we design an experiment to test the performances of our heuristic procedures. We first discuss our data generation scheme, state our performance measures and then discuss the results of our experiment.

5.1 Data Generation

We take our data basically from Akkan *et al.* (2005). According to their scheme, the durations are generated from uniform discrete distribution between 3 and 123. The durations are then sorted in their non-increasing order such that t_k refers to the k^{th} smallest duration. The minimum cost, c_m is generated from discrete uniform distribution between 5 and 15. Thereafter, c_{k-1} is set to $(c_k + s_k) \times (t_k - t_{k-1})$ where $s_{k-1} \epsilon \cup [s_k, s_k + 3]$ or $s_{k-1} \epsilon \cup [\max(1, s_k - 3), s_k]$.

We take 15 test problems from the below ranges of the number of the activities (Table 5.1).

Table 5.1 Range of the Problem and Number of Instances

Range	# of instances
[30,35]	5
[40,45]	5
[85,90]	5

For each activity, the number of modes is generated from uniform discrete distribution between 1 and 10.

In our experiments, we analyze the effects of interval lengths and budget values on the performances of the algorithms. We generate interval lengths to represent two cases:

Case 1. Narrow Time Intervals

Case 2. Wide Time Intervals

To assign interval lengths, the average and minimum processing time of each activity is taken. In order not to generate trivial-to-solve problems, for narrow intervals, approximately 1/3 of the sum of average of all activities' processing time is set as the last interval. Remaining time interval lengths are randomly generated. For wide intervals; two folds of the narrow interval values are used.

For each interval length, we generate the budget values to represent the following two cases:

Case 1. High Budget Releases

Case 2. Low Budget Releases

To assign budget release amounts, the average and minimum costs of each activity is taken. In order not to generate trivial-to-solve problems, for low amounts, approximately 1/3 of the sum of average of all activities' cost is set as the total cost through the project. Each budget release is randomly generated ensuring the total cost remains as determined. For high amounts; two folds of the low amount budget release values are used.

Two cases for interval lengths and two cases for budget values together give four different combinations. For each of the 15 problem instances, we consider these 4 cases. Hence we generate and solve a total of 60 problem instances.

5.2 Performance Measures

In this section, we describe our performance measures used to evaluate the efficiency of our IP to find optimal solution, MILP and pure LP to find lower bounds and heuristic approaches.

For IP we use maximum and average Central Processing Unit (CPU) time in seconds as performance measures.

We evaluate the lower bounds by the following measures:

i. CPU times in seconds (average, maximum)
ii. Percent Deviation from the Optimal Solution *(% DEV*) (average, maximum).

    We calculate the *% DEV* of problem instance *i* as follows;

    $$\%\ DEV_i = \frac{OPT_i - LB_i}{OPT_i} \times 100$$

    where

    LB_i = Lower bound value found by MILP or LP for problem instance *i*

    OPT_i = Optimal solution value of problem instance *i*

iii. Number of fractional activities and percent of integer variables (average, maximum)

We evaluate the heuristic procedures by the following measures:

i. CPU times in seconds (average, maximum)
ii. Percent Deviation from the Optimal Solution (*% DEV*) (average, maximum).

    We calculate the *% DEV* of problem instance *i* as follows:

$$\% \, DEV_i = \frac{H_i - OPT_i}{OPT_i} \times 100$$

    where

    H_i = Solution value found by the heuristic for problem instance *i*

iii. Number of times (out of 5 problem instances) the optimal solution is reached.

All mathematical models are solved by the GAMS Base Module version 23.0.2 WIN 9396.9411 VIS x86/MS Windows with CPLEX solver. The algorithms used in the heuristics are coded in C programming language.

The instance runs are performed on the Intel® Core ™2 Duo CPU 2.26 GHz and 3.45 GB of RAM computer.

5.3 Analysis of Results

We present the performances of the integer program (IP) and lower bounds (MILP, LP) in Table 5.2. The table reports the average and maximum CPU times to achieve the optimal solutions to IP, MILP and LP by using GAMS with CPLEX solver.

As can be observed from the Table 5.2, LP can be solved in negligible time. It takes less than 1 second to get the lower bound, in majority of the combinations. The performance of LP does not deteriorate with an increase in the problem sizes. For example when *n* increases from 40 to 85, the average CPU times increase from 0.47 to 0.91 seconds when budget releases are low and the interval lengths are wide. The same results hold for other combinations. However, for MILP and IP, the CPU times increase drastically with an increase in problem size. The difference becomes much pronounced when *n* gets larger. For example when n increases from 30 to 40, the average CPU times by IP increase from 8.22 to 88.14 seconds when the budget releases are low and interval length is narrow. For this combination, when *n* increases from 40 to 85, the average CPU times increase from 88.144 to 371.85 seconds. When budget releases are low and interval lengths are wide, 1 problem instance out of 5 remain unsolved in one hour for *n*=85, thus it dominates the other values when finding the average CPU time of 824.72 seconds. These observations are in line with MILP results. For example when *n* increases from 30 to 40, the average CPU times by MILP increase from 14.43 to 29.94 seconds when the budget releases are low and interval length is narrow. For this combination, when *n* increases from 40 to 85, the average CPU times increase from 29.94 to 1020.49 seconds. 1 problem instance out of 5 remains unsolved in one hour and this instance plays a dominant role in high average CPU time.

The majority of the instances are solved easier by MILP than by IP. However there are some expectations like when *n*=85 the budget releases are and high interval length is narrow, and the average CPU time of IP is 14.68 seconds whereas the average CPU time of MILP is 22.27 seconds. This is due to the fact that the MILP solution tries to reach the optimal value by searching every possibility before reaching to it. For example for some instances, although it has found the optimal solution in MILP, it didn't stop until it finished its search around the optimal, which is counted as CPU time at the end.

Table 5.2 The CPU Times (seconds) of the IP, MILP and LP Models

Budget Release	Problem Size	IP				MILP				LP			
		Narrow Interval		Wide Interval		Narrow Interval		Wide Interval		Narrow Interval		Wide Interval	
		Average	Max	Average	Max	Average	Max	Average	Max	Average	Max	Average	Max
Low	30-35	8.22	14.44	21.59	85.42	14.43	22.14	13.13	25.48	0.43	0.47	0.41	0.48
	40-45	88.14	356.66	21.96	37.39	29.94	72.03	14.89	25.39	0.53	0.63	0.47	0.49
	85-90	371.85	897.32	824.72	3600	1020.49	3601	77.54	119.98	1.06	1.22	0.91	1.09
High	30-35	11.94	31.81	12.19	47.34	17.5	36.13	7.15	13.58	0.47	0.56	0.38	0.45
	40-45	38.11	125.49	30.17	117	20.64	80.95	21.41	69.19	0.45	0.49	0.45	0.47
	85-90	14.68	36.51	43.15	112.59	22.27	62.42	589.36	2605.5	1.01	1.16	0.98	1.11

Note that the easily solvable problem instances are observed when budget releases are lower and the intervals are wider. The lower the budget releases, the easier the solutions, as the search of the solutions begin from the first modes of the time-cost pair. Thus, iterations of the solution generate lower number of decision variables compared to the higher budget releases. For example when budget releases are low and time intervals are wide, for n=85, average CPU time is 77. 54 seconds, whereas when the budget releases increase, average CPU times increase drastically to 589.36 seconds in wider intervals. Similarly, wider intervals also generate easier solutions, since the construction logic of MILP is based on conducting the activity within a single time interval. Thus, the wider the time interval, the shorter the solution time would be. For example for n=40, when the budget is low and intervals are wide, the average CPU time is 14.89 seconds, whereas when the intervals are narrower, the average CPU time increases to 29.94 second. However this is based on both the intuition and outcomes of the solution instances. Thus, there may be some exceptions.

Table 5.3 reports the average and maximum deviations of the lower bounds from the optimal solutions, as the percentage of the optimal solution.

As can be observed from the table, the deviations by LPs are high; the majority of the deviations are between 20% and 35%. The LP deviations do not deteriorate with an increase in problem sizes. The wider interval instances return slightly bigger deviations, as the search of the solutions begin from the first modes of the time-cost pair. First modes have lower costs and higher durations. Thus, the program finds a solution easily but with a worse deviation. For example, for n=85, when the budget releases are high and interval lengths are narrow, deviation is 10.09% whereas when the intervals are wide, it increases to 17.63%. The deviations of MILP are very low, all deviations are below 8%, and they are consistent over all problem combinations. This supports that the MILP can be used to estimate the optimal solution.

Table 5.3 The Percent Deviation of MILP and IP Solutions

Budget Release	Problem Size	MILP				LP			
		Narrow Interval		Wide Interval		Narrow Interval		Wide Interval	
		Average	Max	Average	Max	Average	Max	Average	Max
Low	30-35	1.81	3.79	1.06	1.65	21.59	23.43	21.75	24.51
	40-45	3.25	5.86	3.84	5.62	24.09	29.47	25.25	32.61
	85-90	3.49	4.86	3.57	5.06	24.44	34.16	26.18	35.98
High	30-35	2.09	3.05	1.16	2.05	21.81	25.92	24.82	28.74
	40-45	4.09	7.88	3.81	5.1	18.71	31.53	19.27	28.64
	85-90	2.52	6.23	4.22	7.38	10.09	21.27	17.63	26.16

Tables 5.4 and 5.5 report the percentages of integer variables and fractional activities of the optimal relaxations returned by MILP and LP, respectively.

As can be observed from Table 5.4, percentages of integer variables of the MILP are mostly above 50 whereas the percentages of integer variables of LP are either 0 or very close to 0. Similarly, percentages of fractional activities are mostly less than 30 whereas the percentages of integer variables of LP are close to 100. Tables 5.4 and 5.5 complement each other. As the number of fractional activities increases, the number of integer variables decreases. Almost all the activities are fractional in LP solutions, which leads to a worse approximation compared to the MILP solutions. As stated above, wider intervals in the MILP solutions give smaller percentages of fractional activities as the construction idea of the MILP is based on conducting the activity within a single time interval. This follows, when the interval is wide, the percentage of integer variables would be higher based on our intuition which is also supported by our experiments. For example when budget releases are low and interval lengths are wide, for n=85, percentage of integer variables is 76.43 for the MILP, whereas the percentage is 0 for LP solution. These results have motivated us to the MILP as a starting point of our heuristic procedures.

Table 5.4 The Percentage of Integer Variables of the Optimal Relaxations

Budget Release	Problem Size	MILP				LP			
		Narrow Interval		Wide Interval		Narrow Interval		Wide Interval	
		Average	Max	Average	Max	Average	Max	Average	Max
Low	30-35	70.03	76.47	73.83	76.47	0.27	1.35	0.29	1.43
	40-45	58.81	71.43	65.88	78.72	0	0	0	0
	85-90	70.88	77.08	76.43	82.69	0.69	2.91	0	0
High	30-35	58.89	70.59	73.27	87.5	0	0	0	0
	40-45	44.92	53.57	51.07	58.49	1.76	8.79	0.43	2.13
	85-90	62.36	71.72	66.94	77.36	5.97	16.56	3.73	15

Table 5.5 The Percentage of Fractional Activities of the Optimal Relaxations

Budget Release	Problem Size	MILP				LP			
		Narrow Interval		Wide Interval		Narrow Interval		Wide Interval	
		Average	Max	Average	Max	Average	Max	Average	Max
Low	30-35	17.84	27.59	15.08	17.24	99.33	100	99.31	100
	40-45	26.26	35	20.9	30.95	100	100	100	100
	85-90	17.18	23.53	13.65	17.65	98.59	100	100	100
High	30-35	26.74	48.28	15.82	27.59	100	100	100	100
	40-45	38.48	50	32.55	38.1	94.69	100	99.05	100
	85-90	23.29	27.06	20.24	24.71	88.24	100	92.71	100

We next report on the performance of our MILP and heuristic procedures regarding the gap analysis. Table 5.6 reports the CPU times and percent deviations for the MILP solution. Tables 5.7 and 5.8 report the CPU times and percent deviations for our heuristic procedures.

Note that Heuristics 1 and 2 require the MILP solutions. The MILP can be solved optimally or by allowing some gap of optimality. The solutions with smaller gaps are likely to be more satisfactory, however they are obtained with high computational effort, i.e., in high CPU times.

In our initial experiments, to see the effect of α (percentage gap) on the solution times and quality of the solutions, we considered the problem instances with 85 activities. We tried α=0% (optimal), α=5%, α=10% and α=15% cases. Table 5.6 reports the average and maximum CPU times to solve the MILP for different values of α. The Table 5.6 also reports the time to obtain an optimal solution via IP. As can be observed from the table, the CPU times drop considerably with an increased α value. The CPU times with α=0%, is generally higher than the solution times of IP. Note that when the budget releases are low and time intervals are narrow the average CPU time of MILP is 1020.49 seconds and the average CPU time of IP is 371.86 seconds. For this combination, for α=5%, 10% and 15%, we obtain CPU times of 135.21, 107.47 and 99.45 seconds respectively that are much lower than that of IP (371.86 seconds). For all problem combinations, the CPU times with α=10% and α=15% are very close.

Table 5.6 Performance of MILP for Different Gap Values

Budget Releases			CPU Time (seconds)				% Dev.			
	Time Interval		Narrow		Wide		Narrow		Wide	
	Problem	% Gap (α)	Average	Max	Average	Max	Average	Max	Average	Max
Low	MILP	0	1020.49	3601	77.54	119.98	3.49	4.86	3.57	5.06
		5	135.21	244.41	54.71	77.81	3	4.86	3.55	5.04
		10	107.47	219.69	35.93	53.33	4.16	6.87	2.94	5.04
		15	99.45	217.97	30.41	52.17	4.1	4.1	2.52	5.04
	IP	-	371.85	897.32	304.92	1001	-	-	-	-
High	MILP	0	22.27	62.42	589.36	2605.5	2.52	6.23	4.22	7.38
		5	15.42	39.33	42.04	76.55	2.48	6.07	4.31	7.33
		10	13.2	30.94	29.11	50.25	2.46	5.94	4.05	7.06
		15	10.91	21.5	24.2	49.89	2.24	5.94	1.81	3.84
	IP	-	14.68	36.51	43.15	112.59	-	-	-	-

Table 5.7 Performance of Heuristic 1 for Different Gap Values

Budget Releases			CPU Time (seconds)				% Dev.			
	Time Interval		Narrow		Wide		Narrow		Wide	
	Heuristic	% Gap (α)	Average	Max	Average	Max	Average	Max	Average	Max
Low	H1	0	1020.71	3601.28	77.83	120.29	17.38	21.97	14.02	21.68
		5	135.52	244.77	55.03	78.17	17.37	25.05	14.2	21.68
		10	107.8	220.05	36.24	53.66	21.54	32.03	14.75	21.68
		15	99.61	218.16	30.56	52.33	24.97	37.35	14.89	24.6
High	H1	0	22.56	62.71	589.66	2605.81	35.89	45.19	32.26	40.96
		5	15.74	39.67	42.36	76.89	34.77	40.96	32.81	40.96
		10	13.52	31.28	29.43	50.58	34.77	40.96	32.81	40.96
		15	11.07	21.67	24.36	50.05	34.77	40.96	34.36	46.34

Table 5.8 Performance of Heuristic 2 for Different Gap Values

Budget Releases			CPU Time (seconds)				% Dev.			
	Time Interval		Narrow		Wide		Narrow		Wide	
	Heuristic	% Gap (α)	Average	Max	Average	Max	Average	Max	Average	Max
Low	H2	0	1032.18	3642.81	94.17	173.02	0.76	1.85	0.87	2.19
		5	149.57	286.64	61.79	97.2	1.06	1.85	1	2.19
		10	122.06	261.7	42.82	72.63	2.06	5.67	0.87	1.28
		15	113.15	260	36.55	64.55	2.27	5.67	1.45	4.07
High	H2	0	25.62	67.57	593.58	2611.64	0.76	2.18	0.76	1.46
		5	19.44	45.5	46.03	82.5	0.65	2.18	0.72	1.46
		10	17.05	37.09	33.36	57.16	0.54	2.18	0.57	1.06
		15	14.93	27.72	28.42	56.06	0.54	2.18	2.23	7.32

Tables 5.7 and 5.8 report the average CPU times and percent deviations of the heuristic procedures (Heuristic 1 and Heuristic 2) that use the solutions of MILP with different α values.

As can be observed from the tables the deviations with α=10% are close to the deviations with α=5%. On the other hand, the CPU times with α=5% are higher. Note from the tables that for Heuristic 1, the CPU times of 135.52, 55.03, 15.74 and 42.36 for α=5% are higher than the respective CPU times of 107.80, 36.24, 13.52, and 29.43 for α=10%. The same conclusions can be made for Heuristic 2. We observe that in general, better deviations are due to α=10% gap than α=15% gap for both heuristics. Note that for Heuristic 1, the average deviations of 21.54, 14.75, 34.77 and 32.81% (for combinations 1, 2, 3 and 4) for α=10% are no higher than their respective deviations of 24.98, 14.89, 34.77 and 34.36% for α=15%. These observations are true for Heuristic 2 too.

As α=10% catches the trade-off between the CPU times and percent deviations better than α=5% and α=15%, we choose α=10%.

Table 5.9 tabulates the CPU times of the IP and Heuristics after improvement. CPU time for the Improved Heuristic 1 is the sum of MILP (with 10% gap) solution time and improvement stage, Heuristic 2's CPU time corresponds to the sum of MILP (with 10% gap) solution time and IP solution time, and CPU time for the Improved Heuristic 3 is the sum of LP solution time and the time spent in improvement stage. Improvement stage is performed simply by loops and shifting algorithms in polynomial time. Thus has a negligible CPU time. As can be observed from the table, Heuristic 3 runs the quickest. However, Heuristics 1 and 2 run in exponential time as can be observed from the tabulated values. The CPU times of the Improved Heuristic 1 are lower than those of Heuristic 2. For Improved Heuristic 1 and Heuristic 2, mostly as the number of activities increases, the average CPU time also increases. This is also valid for Improved

Heuristic 3, however the highest average CPU time is of 1.69 seconds, which is still very low.

Table 5.10 tabulates the percent deviations of the Heuristics after improvement. The averaged percent deviations obtained by Improved Heuristic 1 are all smaller than 12, except one instance when n=40, budget release is low and the interval length is wide. For this combination, the average value is dominated by one instance. The 118.93% deviation of this single instance raised the average deviation to about 30%. In Heuristic 2, the percent deviations of the instances are all smaller than 7%. As can be observed from the table, for the instances of n=30 and n=85 majority of the percent deviations are lower than those of n=40. Improved Heuristic 3's percent deviations are still higher those of the other two heuristics. They are mostly smaller than 63% with some exceptions.

Table 5.9 The CPU Times (seconds) of the IP and Heuristics

Budget Release	Problem Size	IP				Improved Heuristic 1			
		Narrow Interval		Wide Interval		Narrow Interval		Wide Interval	
		Average	Max	Average	Max	Average	Max	Average	Max
Low	30-35	8.22	14.44	21.59	85.42	6.76	9.83	9.1	18.78
	40-45	88.14	356.66	21.96	37.39	12.25	20.39	9.2	16.59
	85-90	371.85	897.32	824.72	3600	108.08	220.34	36.54	53.94
High	30-35	11.94	31.81	12.19	47.34	12.36	30.46	6.06	10.84
	40-45	38.11	125.49	30.17	117	6.97	14.05	14.66	46.94
	85-90	14.68	36.51	43.15	112.59	13.81	31.56	29.72	50.89

Budget Release	Problem Size	Heuristic 2				Improved Heuristic 3			
		Narrow Interval		Wide Interval		Narrow Interval		Wide Interval	
		Average	Max	Average	Max	Average	Max	Average	Max
Low	30-35	8.03	32.26	9.78	19.56	0.72	0.77	0.71	0.77
	40-45	13.9	16.14	10.44	17.84	0.92	1	0.87	0.89
	85-90	122.06	34.12	42.82	61.84	1.69	1.86	1.64	1.98
High	30-35	13.14	32.26	6.42	11.08	0.77	0.86	0.69	0.73
	40-45	8.63	16.14	16.37	51.75	0.84	0.89	0.84	0.88
	85-90	17.05	34.12	33.36	54.24	1.62	1.75	1.59	1.72

Table 5.10 The Percent Deviations of the Heuristics

Budget Release	Problem Size	Improved Heuristic 1				Heuristic 2				Improved Heuristic 3			
		Narrow Interval		Wide Interval		Narrow Interval		Wide Interval		Narrow Interval		Wide Interval	
		Average	Max	Average	Max	Average	Max	Average	Max	Average	Max	Average	Max
Low	30-35	7.8	12.24	6.17	14.48	2.56	7.48	0.64	1.1	51.93	75.67	61.26	107.97
	40-45	11.26	19.75	30.34	118.93	1.08	2.56	3.34	12.99	34.65	72.41	77.41	111.14
	85-90	10.01	21.36	8.27	14.08	2.06	5.67	0.87	1.28	48.66	134.29	200.47	741.42
High	30-35	10.47	27.4	9.08	14.26	0.87	2.28	2.68	4.96	30.06	71.46	62.37	140.56
	40-45	10.68	17.41	9.21	15.04	6.66	30.36	5.63	18.7	23.6	37.57	47.36	79.48
	85-90	11.78	21.83	11.33	22.53	0.54	2.18	0.57	1.06	10.99	22.93	40.71	89.68

Tables 5.11 and 5.12 together reveal the improvement procedures improve the performance of heuristic procedures considerably. Tables 5.11 and 5.12 tabulate the CPU times and the percent deviation of Heuristics and Improved Heuristics, respectively.

As can be seen from the Table 5.11, the average CPU times of Improved Heuristics are slightly higher than the original Heuristics. For example, when n=40, the budget is low and the interval length is wide, the average CPU time of Heuristic 1 is 11.97 seconds, whereas the Improved Heuristic 1 has an average CPU time of 12.25 seconds. This indicates an effective approach in terms of CPU time.

As can be observed from Table 5.12, the improved heuristics have lower percentage deviations than those of their original versions. The highest reductions due to the improvements are observed when the budget releases are higher and the interval lengths are narrower. This is due to the fact that the Improvement Algorithm tends to reduce the activity completion times and force them to fit in one interval. For example, when the budget releases are low, the reductions are 3.35%, 26.77% and 22.99% when the interval lengths are narrow, whereas 4%, 18.62% and 21.48% when the interval lengths are wide for $n=30$, $n=40$ and $n=85$ respectively.

Table 5.11 The CPU Times (seconds) of Heuristics and Improved Heuristics (IH)

Budget Release	Problem Size	Narrow Interval				Wide Interval			
		Average		Max		Average		Max	
		H1	IH1	H1	IH1	H1	IH1	H1	IH1
Low	30-35	6.63	6.76	9.66	9.83	8.89	9.1	18.57	18.78
	40-45	11.97	12.25	20.13	20.39	8.9	9.2	16.34	16.59
	85-90	107.8	108.08	220	220.34	36.24	36.54	53.64	53.94
High	30-35	12.19	12.36	30.34	30.46	5.87	6.06	10.62	10.84
	40-45	6.69	6.97	13.78	14.05	14.35	14.66	46.59	46.94
	85-90	13.52	13.81	31.28	31.56	29.43	29.72	50.58	50.89
		H3	IH3	H3	IH3	H3	IH3	H3	IH3
Low	30-35	0.177	0.72	0.218	0.77	0.181	0.71	0.203	0.77
	40-45	0.181	0.92	0.203	1	0.171	0.87	0.218	0.89
	85-90	0.293	1.69	0.312	1.86	0.321	1.64	0.375	1.98
High	30-35	0.175	0.77	0.203	0.86	0.181	0.69	0.203	0.73
	40-45	0.177	0.84	0.187	0.89	0.19	0.84	0.218	0.88
	85-90	0.303	1.62	0.312	1.75	0.312	1.59	0.359	1.72

Table 5.12 The Percent Deviation of Heuristics and Improved Heuristics (IH)

Budget Release	Problem Size	Narrow Interval				Wide Interval			
		Average		Max		Average		Max	
		H1	IH1	H1	IH1	H1	IH1	H1	IH1
Low	30-35	12.10	7.80	19.67	12.24	9.03	6.17	21.29	14.48
	40-45	20.84	11.26	30.18	19.75	33.66	30.34	118.93	118.93
	85-90	21.54	10.01	32.03	21.36	14.75	8.27	21.68	14.08
High	30-35	13.82	10.47	27.40	27.40	13.08	9.08	24.26	14.26
	40-45	37.45	10.68	101.79	17.41	27.83	9.21	61.79	15.04
	85-90	34.77	11.78	40.96	21.83	32.81	11.33	40.96	22.53
		H3	IH3	H3	IH3	H3	IH3	H3	IH3
Low	30-35	51.93	51.93	75.67	75.67	61.26	61.26	107.97	107.97
	40-45	38.30	34.65	72.41	72.41	77.41	77.41	111.14	111.14
	85-90	49.14	48.66	134.29	134.29	204.76	200.47	757.93	741.42
High	30-35	32.61	30.06	79.45	71.46	62.82	62.37	140.56	140.56
	40-45	32.49	23.60	46.43	37.57	51.89	47.36	79.48	79.48
	85-90	31.66	10.99	40.96	22.93	51.04	40.71	89.68	89.68

Table 5.13 gives the number of times each heuristic returns an optimal solution for every instance.

Table 5.13 The Number of Instances that Return Optimal Solution

Budget Release	Problem Size	Improved Heuristic 1		Heuristic 2		Improved Heuristic 3	
		Narrow Interval	Wide Interval	Narrow Interval	Wide Interval	Narrow Interval	Wide Interval
Low	30-35	0	0	1	0	0	0
	40-45	1	0	2	1	0	0
	85-90	0	0	1	0	0	0
High	30-35	0	0	1	1	0	0
	40-45	0	0	3	1	0	0
	85-90	1	1	3	2	2	1

The performances after the improvement are very satisfactory. As can be seen from the Table 5.13, in 3 out of 5 instances, Heuristic 2 finds the optimal solution when the budget releases are high and the interval lengths are narrow for n=40 and n=85. Heuristic 2 returns at least 1 optimal solution out of 5, for all instances except for two combinations when n=30 and n=85, when the budget releases are low and interval lengths are wide. Although the Improved Heuristic 3 returns high CPU times, when n=85 and budget releases are high, it finds optimal solutions in 3 out of 10 problem instances.

Our recommendation to the project managers is to use Heuristic 3, when obtaining quick solutions is more important than the quality of the solutions. If the quality of the solutions is more important, i.e., the quick deliveries are essential then Heuristic 1 should be used.

CHAPTER 6

CONCLUSIONS

In this thesis, we consider a discrete time/cost trade-off problem with a single resource constraint. The single resource is assumed to be nonrenewable, hence can well be represented by monetary units. The resource is released at some prespecified time points at some prespecified quantities. It is assumed that the cost incurred by an activity is charged at the end of its completion. A feasible solution is the one that consumes no more than total amount released at any time point. When all money is released at the beginning of the project's start then the problem reduces to the discrete time/cost trade-off budget problem which is strongly $\mathcal{NP}$-hard. This follows our problem with arbitrary resource release times is strongly $\mathcal{NP}$-hard.

We first formulate the problem as a pure Integer Programming (IP) model. The model decides on the completion time each activity as well as its mode. We observe that the Linear Programming Relaxation (LPR) produces satisfactory results in terms of number of continuous variables. We strengthen the LPR model by introducing an integer variable to prevent the assignment of an activity to more than one completion time. In the resulting Mixed Integer Linear Program (MILP) model we still have continuous decision variables due to splits in mode assignments. We use the optimal objective function value of the MILP model as an under-estimate, i.e., lower bound, on the optimal project completion time. We develop three heuristic procedures that use the approximate solutions of MILP

and LPs. Our first heuristic fixes the assignments made by MILP and solves the remaining problem to optimality. The other heuristics use the solutions of MILP and LP to find mode assignments. Given the mode assignments the completion times are found by shifting algorithm and then reduced by improvement algorithm.

The results of our experiments reveal that the first heuristic finds solutions that are very close to the optimal however with high computational effort. The other heuristics find quick solutions that are reasonably close to the optimal. We also observe that the number activities, budget amounts and interval lengths have significant effects on the performance.

Future research may consider the following areas:

- Renewable resources can be included.
- More than one nonrenewable resource might be studied.

Different payment schedules may be investigated. These schedules may include the payments at the start times of the activities or the payments at the completion/start times of defined milestone activities.

REFERENCES

1. Carlier, J. and Rinnooy Kan, A.H.G., (1982), "Scheduling subject to nonrenewable-resource constraints", Operations Research Letters, 1(2), pp. 52-55.

2. De, P., Dunne, E.J., Ghosh, J.B., and Wells, C.E., (1995), "The discrete time–cost tradeoff problem revisited", European Journal of Operational Research, 81, pp. 225–238.

3. Değirmenci, G. and Azizoğlu M., (2008), "Discrete Time/Cost Trade-off Problem in Project Networks", YA/EM 2008 Yöneylem Araştırması/Endüstri Mühendisliği 28. Ulusal Kongresi, *in submission*.

4. Demeulemeester, E., De Reyck, B., Foubert, B., Herroelen, W., and Vanhoucke, M., (1998), "New computational results on the discrete time/cost trade-off problem in project networks", Journal of the Operational Research Society, 49, pp. 1153–1163.

5. Demeulemeester, and E., Herroelen, W., (2002), "Project Scheduling – A Research Handbook", Kluwer Academic Publishers, Massachusetts.

6. Demeulemeester, E., Herroelen, W., and Elmaghraby, S.E., (1996), "Optimal procedures for the discrete time/cost trade-off problem in project networks", European Journal of Operational Research, 88, pp. 50–68.

7. Elmaghraby, S.E., (1970), "Some Network Models in Management Science", 1. Edition, Springer-Verlag, Germany.

8. Elmaghraby, S.E., (1977), “Activity Networks”, 1. Edition, John Wiley & Sons, Inc., Canada.

9. Fulkerson, D.R., (1961), “A network flow computation for project cost curves”, Management Science, 7, pp. 167–178.

10. Hafızoğlu, B. and Azizoğlu, M., (2008), “LP-Based Procedures for Discrete Time/Cost Trade-off Project Management Problem”, 22nd European Conference on Operational Research, *in printing*.

11. Hazır, Ö., Haouari, M., Erel, E., (2009), “Discrete time/cost trade-off problem: A decomposition based solution algorithm for the budget version”, Computers and Operations Research, In Press, Corrected Proof, Available online 21 June 2009.

12. Herroelen W.S., (1972), “Resource-constrained Project Scheduling – the State of the Art”, Operational Research Quarterly (1970-1977), 23(3), pp. 261-275.

13. Herroelen, W., Reyck, B.D., and Demeulemeester, E., (1998), “Resource-constrained Project Scheduling: A Survey of Recent Developments”, Computers Operations Research, 25(4), pp. 279-302.

14. İçmeli, O., Erengüç, S.S., and Zappe, C.J., (1993), “Project Scheduling Problems: A Survey”, International Journal of Operations & Production Management, 13, pp. 80-91.

15. Klein, R., (1999), “Scheduling of Resource Constrained Projects”, 1. Edition, Kluwer Academic Publishers, Massachusetts.

16. Kolisch, R., and Padman, R., (2001), “An integrated survey of deterministic project scheduling”, The International Journal of Management Science, 29, pp. 249-272.

17. Moder, J.J., Phillips, C.R., Davis, E.W., (1983), “Project Management with CPM, PERT and Precedence Diagramming, 2. Edition, Van Nostrand Reinhold Company Inc., New York.

18. Project Management Institute, 2004, “A guide to the project management body of knowledge: PMBOK® guide. – 3rd ed.”, 3. Edition, Project Management Institute, Inc., Pennsylvania.

19. Pritsker, A.A.B., Waiters, L.J., Wolfe, P.M., (1969), “Multiproject Scheduling with Limited Resources: A Zero-One Programming Approach”, Management Science, 16(1), pp. 93-108.

20. Talbot, F.B., (1982), “Resource-Constrained Project Scheduling with Time-Resource Tradeoffs: The Nonpreemptive Case”, Management Science, 28(10), pp. 1197-1210.

21. Weglarz, J., (1998), “Project Scheduling – Recent Models, Algorithms and Applications”, 1. Edition, Kluwer Academic Publishers, Massachusetts.

22. Wiest, J.D. and Levy F.K., (1969), “A Management Guide to PERT/CPM”, 1. Edition, Prentice-Hall Inc., New Jersey.