# MODELS AND ALGORITHMS FOR DETERMINISTIC AND ROBUST DISCRETE TIME/COST TRADE-OFF PROBLEMS

A Ph.D. Dissertation

by
ÖNCÜ HAZIR

Department of Management
Bilkent University
Ankara
May 2008

*To my dearest family;*
*My mother; Gülsen,*
*My father; İsmail,*
*and My sister; Özgü*

# MODELS AND ALGORITHMS FOR DETERMINISTIC AND ROBUST DISCRETE TIME/COST TRADE-OFF PROBLEMS

The Institute of Economics and Social Sciences
of
Bilkent University

by

ÖNCÜ HAZIR

In Partial Fulfillment of the Requirements for the Degree of
DOCTOR OF PHILOSOPHY

in

THE DEPARTMENT OF
MANAGEMENT
BİLKENT UNIVERSITY
ANKARA

May 2008

I certify that I have read this thesis and have found that it is fully adequate, in scope and in quality, as a thesis for the degree of Doctor of Philosophy in Management.

--------------------------------
Professor Erdal Erel
Supervisor

I certify that I have read this thesis and have found that it is fully adequate, in scope and in quality, as a thesis for the degree of Doctor of Philosophy in Management.

--------------------------------
Professor İhsan Sabuncuoğlu
Examining Committee Member

I certify that I have read this thesis and have found that it is fully adequate, in scope and in quality, as a thesis for the degree of Doctor of Philosophy in Management.

--------------------------------
Professor Mohamed Haouari
Examining Committee Member

I certify that I have read this thesis and have found that it is fully adequate, in scope and in quality, as a thesis for the degree of Doctor of Philosophy in Management.

--------------------------------
Assistant Professor Yavuz Günalay
Examining Committee Member

I certify that I have read this thesis and have found that it is fully adequate, in scope and in quality, as a thesis for the degree of Doctor of Philosophy in Management.

--------------------------------
Assistant Professor Ayşe Kocabıyıkoğlu
Examining Committee Member

Approval of the Institute of Economics and Social Sciences

--------------------------------
Professor Erdal Erel
Director

## ABSTRACT

MODELS AND ALGORITHMS FOR DETERMINISTIC AND ROBUST DISCRETE TIME/COST TRADE-OFF PROBLEMS

Hazır, Öncü
Ph.D., Department of Management
Supervisor: Prof. Dr. Erdal Erel

May 2008

Projects are subject to various sources of uncertainties that often negatively impact activity durations and costs. Therefore, it is of crucial importance to develop effective approaches to generate robust project schedules that are less vulnerable to disruptions caused by uncontrollable factors. This dissertation concentrates on robust scheduling in project environments; specifically, we address the discrete time/cost trade-off problem (DTCTP).

Firstly, Benders Decomposition based exact algorithms to solve the deadline and the budget versions of the deterministic DTCTP of realistic sizes are proposed. We have included several features to accelerate the convergence and solve large instances to optimality. Secondly, we incorporate uncertainty in activity costs. We formulate robust DTCTP using three alternative models. We develop exact and heuristic algorithms to solve the robust models in which uncertainty is modeled via interval costs. The main contribution is the incorporation of uncertainty into a practically relevant project scheduling problem and developing problem specific solution approaches. To the best of our knowledge, this research is the first application of robust optimization to DTCTP.

Finally, we introduce some surrogate measures that aim at providing an accurate estimate of the schedule robustness. The pertinence of proposed measures is assessed through computational experiments. Using the insight revealed by the computational study, we propose a two-stage robust scheduling algorithm. Furthermore, we provide evidence that the proposed approach can be extended to solve a scheduling problem with tardiness penalties and earliness rewards.

Keywords: Project Scheduling, Time/Cost Trade-off, Robust Optimization, Benders Decomposition.

## ÖZET

DETERMİNİSTİK VE GÜRBÜZ KESİKLİ ZAMAN/MALİYET ÖDÜNLEŞİM PROBLEMLERİ İÇİN MODELLER VE ALGORİTMALAR

Hazır, Öncü
Doktora, İşletme Bölümü
Tez Yöneticisi: Prof. Dr. Erdal Erel

Mayıs 2008

Proje çizelgeleri, projenin ne zaman tamamlanacağını, hangi faaliyetlerin ne zaman yapılacağını ve kaynakların faaliyetlere nasıl atanacağını belirtir. Mevcut proje çizelgeleme yöntemlerinin büyük çoğunluğu proje çizelgelerinin öngörüldüğü şekilde uygulanabileceğini varsaymaktadır. Fakat pratikte projeler, kaynak kullanımındaki, faktör fiyatlarındaki, nakit akışlarındaki değişkenliklerden, nitelik problemleri sebebiyle işlerin tekrarlanması ve buna benzer diğer belirsizlik kaynaklarından etkilenmektedirler. Bu çalışmada proje çizelgeleme modellerinde belirsizlik göz önüne alınmış ve belirsizliğin proje amaçlarına ulaşılmasına etkisinin en aza indirgenmesi için gürbüz çizelgeleme yöntemlerinin geliştirilmesi hedeflenmiştir. Proje ortamı olarak gerçek proje uygulamalarını iyi yansıtan ve literatürde iyi bilinen kesikli zaman/maliyet ödünleşim problemi (KZMÖP) incelenmiştir.

İlk olarak, iki temel belirgin KZMÖP türü incelenmiştir: vade problemi ve bütçe problemi. Vade probleminde proje süresi belirlenen vadeyi geçmeyecek şekilde proje bütçesi enazlanmaktadır. Bütçe probleminde ise proje bütçesi belirlenen miktarı geçmeyecek şekilde proje süresi enazlanmaktadır. Her iki tür için de büyük ölçekli proje çizelgeleme problemlerini kesin olarak çözebilmek için Benders ayrıştırması uygulanmış, probleme özgü hızlandırma mekanizmaları öne sürülmüştür.

Daha sonra maliyetlerdeki belirsizlik göz önüne alınmış ve aktivite maliyetlerinin belirli aralıklar dahilinde gerçekleştiği varsayılmıştır. Bu şartlar altında gürbüz KZMÖP için üç farklı model öne sürülmüş ve bu modellerin etkinliği karşılaştırılmıştır. Modellerin çözümü için kesin ve sezgisel yöntemler öne sürülmüştür.

Son olarak belirsizlik ortamından kaynaklanan risklere karşı çizelgenin direncini, dayanıklılığını, nesnel olarak değerlendirebilmek için ölçü birimleri tasarlanmış ve proje risklerini göz önüne alan iki aşamalı gürbüz proje çizelgeleme yöntemi geliştirilmiştir. Ayrıca, önerilen yaklaşımın gecikme cezası ve erken bitirme kazancı olan bir karmaşık çizelgeleme problemine de dönüştürülebileceği gösterilmiştir.

Anahtar Kelimeler: Proje Çizelgeme, Zaman/Maliyet Ödünleşimi, Gürbüz Eniyileme, Benders Ayrıştırması.

# ACKNOWLEDGEMENTS

I would like to express my sincere gratitude to Prof. Erdal Erel for his continuous support, guidance and inspiration. Since I began studying in the Ph.D. program, he has always been encouraging and supportive.

I am also indebted to Prof. İhsan Sabuncuoğlu, Dr. Yavuz Günalay, Prof. Mohamed Haouari, Prof. Levent Kandiller, Dr. Ayşe Kocabıyıkoğlu and Dr. Nagihan Çömez for their invaluable suggestions and assistance. I have greatly benefited from their suggestions, constructive criticisms and comments during my Ph.D. study.

I also would like to thank to Çilem Selin Akay for her precious support and encouragement. Finally, I express my appreciation to my family İsmail, Gülsen and Özgü for their never ending love, support and drive.

# TABLE OF CONTENTS

## LIST OF TABLES

# LIST OF FIGURES

## LIST OF ABBREVIATIONS AND NOTATION

$A$: Set of arcs.

$a_i$ : Intercept of linear segment $i$.

$\alpha$: Opportunity cost per unit time.

$B$: Weighting factor for the budget.

$b_i$ : Slope of linear segment $i$.

$B_0$: Predefined project budget.

$c_{jm}$: Activity cost of activity $j$ when processed at mode $m$.

$C_j$: *C*ompletion time of activity $j$.

$CR$: Set of potentially critical activities.

$\delta$: Project due date.

$d_{jm}$ : The difference between the upper bound and nominal costs, $d_{jm} = \overline{c_{jm}} - c_{jm}$.

DTCTP: The Discrete Time/Cost Trade-off Problem.

DTCTP - B: The budget version of DTCTP.

DTCTP - D: The deadline version of DTCTP.

$\varepsilon$: *R*elative optimality tolerance level.

$E_j$: Earliest finishing time of activity $j$.

ESS: Early start schedule.

$\eta$: Budget amplification factor.

FS: Free slack.

$\gamma$: Auxiliary variables that are used for feasibility checking.

$\Gamma$: The parameter that reflects the risk attitude of the decision makers, i.e. only $\Gamma$ of activity cost parameters involves random behavior.

$M$: A large positive number.

$M_j$ : Set of executable modes for activity j.

MP: Master problem.

$n$: Number of non-dummy activities.

$N$: Set of nodes.

$P$: Polyhedron

Pr(i): Set of immediate successors of activity $i$

$\phi$ : Interest paid for budget overruns.

$p_{jm}$ : Activity processing time of activity $j$ when processed at mode $m$.

PM: Performance measures.

$\psi$: Uncertainty factor ,i.e. $d_{jm} = \psi.c_{jm}$.

RM: Robustness measures.

$\rho$: Extra revenue gained per unit earliness.

SDR: Slack duration ratio.

SP: Subproblem.

Su(i): Set of immediate successors of activity $i$

$\theta$: The parameter to reflect tightness of deadline or restrictiveness of the budget

TS: Total slack.

$u$: Binary variable to identify the activities with cost deviations that influence the objective most.

$\xi$: Slack/Duration threshold.

$x_{jm}$: Binary variable showing whether mode $m$ is assigned to activity $j$ or not.

$\zeta$: Tardiness penalty per unit time.

# CHAPTER 1

# INTRODUCTION

Projects are one of the most important components of today's organizations. In almost any type and size of organization, it is common to organize tasks as projects. This may be perceived as one consequence of the contemporary management practices, which have transformed from a hierarchical nature to a more flat one. As the organizations have receded from a hierarchical and isolated nature, projects have become the medium for interdepartmental or even inter-organizational activities. Another factor that has affected the private enterprises has been the increasing competitive pressure. Competition, becoming fierce day by day, leads the enterprises to seek excellence in accomplishing the tasks. Hence, monitoring the performance of tasks regarding both the schedule and the cost has gained increasing importance for the realization of the organizational goals.

Meredith and Mantel (2005) claim that organizing tasks as projects serve to focus responsibility and authority in order to achieve the organizational goals. In this way, organizations experience better control, coordination, communication, and customer relations. Due to these advantages, organizations are becoming more project-driven and project management is becoming crucial.

Project management primarily involves defining, planning, monitoring, and controlling functions. Scheduling, as a part of the planning function, is concerned with determining the start and finish times of activities and allocation of scarce resources to these activities. Since the amount of information is limited in practice, the quality of the scheduling lies in the way the uncertainties are handled.

When the literature on project scheduling is examined, we observe that the majority of the work assumes knowledge of complete information and a deterministic environment. Differently, in this dissertation we concentrate on project scheduling under uncertainty. In particular, we study robust project scheduling which aims to generate schedules that are protected against project disruptions. We address two major issues: the former is how to generate robust project schedules and the latter is how to assess robustness of project schedules. In this introductory chapter, we introduce some basic definitions, terminology of project management and scheduling and give an outline of the dissertation.

## 1.1. Definitions and Terminology

A *project* is a collection of interrelated activities that must be completed within some time limits to achieve predefined objectives. Projects are temporary and unique; they have a finite duration and distinguishing characteristics. An *activity* is a work element of a project that consumes time and requires resources during project execution. There exist precedence relationships among project activities due to factors such as technological requirements, economic necessities or legislative requirements. *Precedence relationships* define the processing order of the activities.

Moreover, the activities may have resource relationships among each other; in other words, they may share the same resources.

Each project has specific goals to be achieved. *Goals* define the outcomes to be realized. *Objectives* are operational definitions of the goals. The achievement of the objectives determines the performance of a project. Therefore, the objectives must be concrete and hence measurable. According to Meredith and Mantel (2005), project duration, project cost and specifications set by the customers are the three prime project objectives. Among these, the *project duration* refers to the time committed to complete the project. It is also called the *project makespan* and is the most common objective addressed in the scheduling literature. In addition to the above mentioned project objectives, some others such as completing the project with maximum net present value of cash flows, or with maximum quality are also used in the literature. In addition to optimizing with respect to a single objective, several objectives could be simultaneously sought using multi criteria approaches. We refer the reader to Kolisch and Padman (2001) for a classification of the literature in terms of the objectives addressed.

*Project management* is the management discipline that develops and applies various tools and methods to ensure that project objectives are achieved. Each project passes through conceptual design, definition, planning, monitoring and controlling, and termination phases during its life time. Each phase requires a different set of management techniques. *Conceptual design* identifies the needs for the projects and sets the basic principles that will serve as a reference in the definition phase. In this phase, the problem definition is fuzzy. However, feasibility and risk analysis are performed to decide on whether to start the project or not. Once a project is conceptually designed, the objectives, scope and strategy of a project should be clearly defined. Furthermore, a *budget*, which represents a financial plan of the project, is allocated to the project at this stage. After the definition phase, the *planning* function creates a concrete plan to reach the predefined project objectives. In this phase the work content of the project is divided into work packages that comprise activities. For each activity time, resource and cost requirements are estimated.

*Project scheduling*, which produces time plans, called project schedules, is an important part of the planning function. *Project schedules* define activity start and finish times, and also allocate resources to the activities. *Monitoring* function collects and prepares information that is required to evaluate project performance. *Controlling* function verifies that actual performance matches the planned performance and corrective actions are taken if needed. Accomplishment of the project goals is evaluated and a final report is prepared in the *termination* phase. Furthermore, the project organization is dissolved. In this dissertation, we focus on the planning function and specifically on project scheduling.

Project schedules are prepared before the project execution and this pre-execution plan is called the *baseline* or the *predictive schedule*. Generating a good baseline schedule is important, because this schedule is used to plan and coordinate many activities, such as procurement of materials, planning equipment and staff, etc. Moreover, in practice due dates are usually set utilizing this schedule.

Project activities and their relationships among them are usually displayed by graphical tools called project network diagrams. *Project network diagrams* are schematic tools to display the relationships among the project activities. They are drawn from left to right to reflect the time sequence. A network diagram typically consists of elements called arcs and nodes. Two alternative network representations are possible: activity-on-arc representation and activity-on-node representation. In an *activity-on-node (AON)* representation, nodes represent the activities and arcs define the precedence relationships among the nodes. In an a*ctivity-on-arc (AOA)* representation, the nodes represent events such as the completion of activities, whereas the arcs correspond to the activities.

An activity that must be completed before the beginning of another activity is called *a predecessor*. The activity that can start after the completion of another activity is called *a successor*. *Gating activities* are activities with no predecessors. The duration and the sequence of activities can be represented by a time scaled bar line called a *Gantt chart* that was proposed first by Henry Gantt in 1917. This chart still continues to be the most frequently used method to present schedules.

The Critical Path Method (CPM) and The Program Evaluation and Review Technique (PERT) are network analysis methods, which are widely used in industry. *Critical path* is the longest path in project networks that determines the earliest completion time of the project. The activities on the critical path are called critical activities. CPM is used to define the time schedule and find the critical activities. It was introduced in the late 50's in the United States. Du Pont Chemical Company was the first user of CPM; they applied CPM for planning and maintenance of chemical plants. Kelley and Walker (1959) express the historical development of CPM.

*Earliest start* (*ES*) and *latest start* (*LS*) of an activity define the earliest and the latest points in time related to the starting of the activity. Similarly, *earliest finish* (*EF*) and *latest finish* (*LF*) define the earliest and the latest finish times. Critical path is determined by performing forward and backward passes through the project network. The ES and the EF times for each activity are calculated in the forward pass, the LS and LF times are computed in the backward pass.

In project management literature, two types of slacks are widely used; the total slack and free slack. *Total slack* (*TS*) is the amount of time by which the completion time of an activity can exceed its earliest completion time without delaying the project completion time. *Free slack* (*FS*) is the amount of time by which the completion time of an activity could be delayed without affecting the earliest start time of its immediate successors in the project. Total slack is computed as the difference between ES and LS for each activity, whereas free slack is the difference between EF of an activity and minimum ES of its immediate successors. *Critical activities* have zero TS. Therefore, any delay in these activities will lead to a delay in project completion.

PERT is another commonly used network analysis technique that estimates the project completion time and the starting time of each activity. PERT was developed for the POLARIS missile program of U.S. Navy in 1958 (see Kerzner 2006). It is the first method incorporating uncertainty into activity durations and is seen as a stochastic alternative of CPM. Due to the stochastic characteristic, expected completion times and probabilities of on-time completion are calculated in PERT. It assumes beta distribution for activity durations and requires three time estimates of activity durations: most likely, optimistic and pessimistic estimates. However, beta distribution assumption has been criticized as it may not always be a good approximation to the actual distribution (Maccrimmon and Ryavec, 1964). Furthermore, like CPM, PERT also assumes infinite resource availability. As real life projects have limited resource availability, this unrealistic assumption renders its usability since the activities compete for scarce resources in every project.

Resources may be grouped as renewable, nonrenewable and doubly constrained. A *renewable resource* is available at a constant amount in every instance of the planning period. Machines, equipment and staff are the classical examples of renewable resources. *Nonrenewable resources* are consumable; the available quantities of these resources decrease with consumption. Money is a good example of nonrenewable resources. *Doubly constrained resources* are the resources that have limited availability in every period of the planning horizon and have constrained total availability. As project budgets control the consumption of money both in every period and also over the duration of the complete project, they are typical examples to doubly constrained resources.

*Resource Constrained Project Scheduling Problem (RCPSP)* incorporates resource constraints with constant resource availability assumption. In this problem, the project completion time is minimized. Both precedence relationships among activities and constant resource availability constraints are considered. In resource constrained project scheduling, the concept of critical chain is important. The *critical chain* is the sequence of both precedence and resource dependent activities which determines the minimum completion time of the project. Unlike the critical path, it considers the resource relationships as well.

RCPSP is shown to be NP-hard by Blazewicz et al. (1983). It has been extensively studied in the literature and many different versions of the basic problem have been addressed. Brucker et al. (1999), Herroelen et al. (1998) and Kolisch and Padman (2001) discuss the exact and approximate solution strategies and review the literature comprehensively.

While RCPSP assumes a single execution possibility for an activity, in practice, project activities can be executed in various processing alternatives. Each alternative represents processing with a different technology or with a different resource assignment. In scheduling literature each execution alternative, which is characterized with a fixed duration and a fixed resource allocation, is called a *mode* of the activity. The extension of RCPSP to multi-mode setting is called the *Multi-Mode Resource Constrained Project Scheduling Problem (MRCPSP).* This problem deals with assigning one of the possible modes to each activity so that project completion time is minimized while precedence and resource constraints are satisfied. MRCPSP models the use of renewable, nonrenewable and doubly constrained resources. A special case of MRCPSP that utilizes only one single nonrenewable resource (money) is called *discrete time/cost trade-off problem* (*DTCTP*). It is a well-known project scheduling problem with practical implications. We focus on this problem in this dissertation.

## 1.2. Problems Addressed

Among the three versions of DTCTP, this dissertation addresses the deadline problem (DTCTP-D) and the budget problem (DTCTP-B). DTCTP-D assigns modes to each activity so that the total cost is minimized. On the contrary, given the budget, the budget problem minimizes the project duration. Both of these multi-mode scheduling problems have practical implications as they model the time/cost relationship in processing activities. In practice, project managers often allocate more resources to accelerate the activities and each resource allocation defines an execution mode in scheduling. In real life projects, usually multiple alternatives exist to execute an activity.

These problems are difficult to solve for large scale project networks; both of these DTCTP versions have been shown to be strongly NP-hard optimization problems for general activity networks by De et al. (1997). Firstly, we examine the deterministic deadline and budget problems and propose Benders Decomposition based solution algorithms.

Moreover, in order to represent the real life project environments more realistically in project scheduling models, we relax the complete information and deterministic environment assumptions and incorporate uncertainty into the problems. We introduce robust DTCTP models. In these models, we address the uncertainty in the activity costs and in the durations. We develop algorithms to generate robust schedules, which are less sensitive to these uncertainty sources.

## 1.3. Dissertation Outline

The organization of this dissertation is as follows. In Chapter 2, we discuss optimization and project scheduling under uncertainty in detail and the literature is reviewed comprehensively. In Chapter 3, the discrete time/cost trade-off problems are introduced; Benders Decomposition based solution algorithms for solving the deterministic deadline and budget problems are proposed. In chapter 4, we propose three robust optimization models for DTCTP. In these models, the uncertainty lies in activity costs and is represented via intervals. In order to solve the models, exact and heuristic algorithms are introduced. The schedules that have been generated with these models are compared on the basis of robustness. In Chapter 5, the uncertainty is assumed to lie in activity durations and some surrogate robustness measures are proposed.  We test these measures using simulation and a scheduling algorithm which uses the selected robustness measure is presented. Finally, in Chapter 6 we summarize the contributions of this dissertation to the literature and discuss future research areas.

# CHAPTER 2

# MODELS AND APPROACHES FOR PROJECT SCHEDULING UNDER UNCERTAINTY: LITERATURE REVIEW

In this chapter, we discuss the project scheduling models in detail and present a comprehensive review of the literature. First, we review optimization techniques to hedge against uncertainty, and then concentrate on project scheduling under uncertainty.

## 2.1. Uncertainty and Optimization under Uncertainty

Meredith and Mantel (2005) define *uncertainty* as "having only partial information about the situation or outcomes". Uncertainty is an inevitable part of decision making and strategies to hedge against uncertainty should be developed. To manage uncertainties we consider o*ptimization under uncertainty*, which is the branch of optimization where there are uncertainties involved in the data. Consider the following mathematical programming problem:

$$\text{Min } \{f(x) \colon x \in X \subset R^n \} \qquad (2.1)$$

$$\text{where } X = \{ x \in R^n \colon g_i(x) \geq 0,\ i = 1,\dots,m \} \qquad (2.2)$$

In (2.1) and (2.2), $f$: $R^n \rightarrow R$, and $g_i$: $R^n \rightarrow R$, $i = 1,\ldots,m$ define the objective function, and the constraints, respectively. Throughout the dissertation $f$ and $g_i$ are used to denote functions. Traditionally, in optimization literature all the parameters are usually assumed to be deterministic. However, in optimization under uncertainty, this assumption is relaxed and uncertainty in the parameters is incorporated into the model.

We formally state a mathematical programming model under uncertainty as follows:

Min $\{f(x,u) : x \in X(u),\ u \in U\}$, where $X(u) = \{x \in R^n : g_i(x, u) \geq 0,\ i = 1,\ldots,m\}$ (2.3)

In the above function, $u$ defines the uncertain parameter vector and $U$ characterizes the set of uncertain data, i.e. $u \in U$. Note that in (2.3) the feasible region is dependent on the uncertain parameter set and is represented with $X(u)$. The difference between the models in (2.3) and in (2.1) is that parameter vector is not exactly known and incorporated into the model as uncertain.

The first issue in optimization under uncertainty is how the uncertain data, defined by set $U$, are represented; they might be modeled as being either discrete or continuous. Scenario generation is a widely used approach to model the discrete case. *Scenarios* refer to the realization of the uncertain variables such as activity durations or costs. However, the scenario-based methods face the following difficulties: disruption scenarios cannot be defined easily or identified beforehand, and there may be too many scenarios to consider. Continuous data might be assumed to lie in some pre-specified intervals (*interval uncertainty*) or ellipsoidal sets or various convex sets. In this dissertation, interval uncertainty will be used for the continuous models. For both discrete and continuous cases, one common approach to represent unknown data is using random variables. This approach will be further investigated in Section 2.1.1.

Stochastic programming, robust optimization, sensitivity analysis, parametric programming and fuzzy programming are the fundamental optimization paradigms under uncertainty. In the next section, we concentrate on stochastic programming and robust optimization, which are the most commonly applied paradigms in the literature, and then briefly mention the other paradigms.

### 2.1.1. Stochastic Programming

*Stochastic programming* is a powerful modeling framework that uses probabilistic models to describe the uncertain data in terms of probability distributions. Typically, the average performance of the system is examined and expectation over the assumed probability distribution is taken. In this case, (2.3) could be reformulated as:

$$\text{Min } \{E_u[f(x, u)] : x \in X(u)\}, \tag{2.4}$$

In the above formulation, $u$ refers to a random vector and $E_u[]$ refers to the expectation over all the possible values of the random vector $u$.

The fundamental idea behind stochastic programming is the notion of *recourse*, which is the ability to take corrective action following a random event. *Two-stage stochastic recourse programming* is the most frequently used model in stochastic programming. The decision variables are partitioned into two sets: the first set consists of variables that are set prior to the realization of the uncertain event. The second set includes the recourse variables that represent the response to the first-stage decision and to realized uncertainty. The sum of the first-stage decision cost and the expected cost of optimal second-stage recourse decisions are minimized. The following model, which is based on Birge and Louveaux (1997), expresses the basic two-stage linear programming.

$$\min_{x_1 \in R^{n_1}} \{c_1 x_1 + E_u [Q(x_1, u)] : A_1 x_1 = b_1, x_1 \geq 0\}, \tag{2.5}$$

$$Q(x_1, u) = \min_{x_2 \in R^{n_2}} \{c_2(u)\, x_2 : A_2 x_2 = b_2(u) - D(u)\, x_1, x_2 \geq 0\}. \tag{2.6}$$

In (2.5) and (2.6), $c_1, c_2$ are the objective vectors of sizes $n_1$ x $1$ and $n_2$ x $1$; $A_1$, $A_2$ are the $m_1$ x $n_1$ and $m_2$ x $n_2$ constraint matrices and $b_1$, $b_2$ are the vectors of right-hand side of the constraints with sizes $m_1$ x $1$ and $m_2$ x $1$. Second stage parameters, $c_2(u)$, $b_2(u)$ and $D(u)$, are dependent on random vector $u$ and they become known when $u$ is realized. Uncertainty is modeled by the use of the random vector, $u$ and note that uncertain data is a function of the random vector $u$. The vectors $x_1$ and $x_2$ define first and second stage decision variables, respectively. When the first stage decisions are taken, uncertainty is present in the system; however in the second stage actual values of unknown vector $u$, becomes known and corrective actions are taken by the use of the second stage decision variables. Two-stage programming can naturally be extended to multiple stages.

An application of two-stage stochastic recourse programming is production planning under uncertain demand. This could be modeled as a linear program (LP) as follows. The first-stage variables specify production levels that are determined in the presence of uncertain demand. Once the demand is known, the second-stage variables take recourse in deciding how to do deal with excess or shortage quantities.

Total expected costs comprising shortage penalties and excess costs are minimized. For other applications of stochastic programming, we refer the readers to Birge and Louveaux (1997).

Mulvey et al. (1995) capture the notion of risk in stochastic programming; they propose to integrate a variability measure, typically the variance, on the second-stage costs in the objective function. They describe the uncertainty with scenarios and assume that the probability distributions could be accurately identified. In their scenario based approach, a solution is allowed to violate the constraints; however these violations are penalized.

An alternative stochastic approach is chance constrained programming proposed by Charnes and Cooper (1959). It includes constraints which do not always need to be satisfied; they could be satisfied with some given probabilities. The probabilistic or chance constraints have the following generic form:

$$P(g_i(x, u) \geq 0) \geq p_i \qquad i = 1,\dots,m;\ x \in X(u) \tag{2.7}$$

In this formulation, $P(.)$ refers to a probability distribution associated with uncertain vector $u,$ and $p_i$ is a threshold probability level for constraint $i,$ which is defined by $g_i(.)$.

Stochastic programming is a proper technique when accurate probabilistic description of the randomness is available; however, in many real-life applications the decision-maker either does not have or cannot access this information. In these cases, robust optimization is more appropriate. Furthermore, data requirements of stochastic programming are generally high and it is often computationally demanding.

### 2.1.2. Robust Optimization

*Robust optimization* is a modeling approach to generate a plan that is insensitive to data uncertainty. Generally, the worst-case performance of the system is optimized and plans that perform well under worst-case scenarios are sought. Since it is worst-case oriented, it is a conservative methodology.

The most widely studied robust optimization models are minmax and minmax regret models. Kouvelis and Yu (1997) discuss these models comprehensively and apply them to a wide range of combinatorial optimization problems. The *minmax* models minimize the maximum cost across all scenarios. A general formulation of these modes is:

$$\text{Min}\left\{\underset{u\in U}{\text{Max}}\left\{f(x,u)\right\} : x \in X(u)\right\} \tag{2.8}$$

This modeling approach is extremely pessimistic and might therefore result in poor solutions under many scenarios. They are most suitable for circumstances in which the system is expected to perform well even in the worst-case.

The *regret* of a solution in a given scenario is the difference between the cost of the solution and the cost of the optimal solution for that scenario. Note that scenarios define the realization of uncertain vector *u*. Models that seek to minimize the maximum regret across all scenarios are called *minmax regret* models. The regret for the vector *x* under realization *u* is:

$$r(x,u) = f(x,u) - \text{Min}\left\{f(x,u) : x \in X(u)\right\} \tag{2.9}$$

Using (2.9), the minmax regret model could be formulated as:

$$\mathrm{Min}\left\{\underset{u\in U}{\mathrm{Max}}\left\{r(x,u)\right\} : x\in X(u)\right\} \tag{2.10}$$

Minmax regret models have been employed to model the robust versions of some well-known combinatorial optimization problems in the literature (for shortest path problem see Karaşan et al. (2001), Montemanni and Gambardella (2004); Yaman et al. (2001) and Montemanni and Gambardella (2005) for minimum spanning tree etc.). Ben-Tal and Nemirovski (1999, 2000) follow a worst-case oriented approach and reformulate (2.3) as follows:

$$\mathrm{Min}\ \{z{:}\, f(x,u)\ \le\ z\ ,\ x\ \in\ X(u),\ \forall\ u\ \in\ U\} \tag{2.11}$$

They call the model "robust counterpart" of an optimization under uncertainty model. Note that (2.11) is conservative since $x$ is feasible only if all the constraints for all possible values of $u \in U$ are satisfied. They use interval or ellipsoid uncertainty sets to model $U$ in their models. We elucidate their approach by using the following LP:

$$\mathrm{Max}\ \{\ cx : \sum_j a_{ij}x_j \le b_i,\ i=1,\dots,m\ \} \tag{2.12}$$

Ben-Tal and Nemirovski (2000) assume a row-wise uncertainty and each coefficient $a_{ij}$, $\forall j \in M_i$ are uncertain and bounded with the interval $\left[\hat{a}_{ij} - d_{ij}, \hat{a}_{ij} + d_{ij}\right]$, which is centered at the nominal value, $\hat{a}_{ij}$, and is usually approximated with the mean. $M_i$ refers to the set of coefficients that are subject to parameter uncertainty in row $i$. The parameter $d_{ij}$ is the half length of the interval and defines the precision level of the estimate. They propose the following robust counterpart of LP expressed in (2.12):

Max $cx$

subject to

$$\sum_{j} \hat{a}_{ij} x_j + \sum_{j \in M_i} d_{ij} y_{ij} + \Psi_i \sqrt{\sum_{j \in M_i} d_{ij}^{2} w_{ij}^{2}} \le b_i \qquad i = 1,\dots,m \tag{2.13}$$

$$-y_{ij} \le x_j - w_{ij} \le y_{ij} \qquad i = 1,\dots,m, \qquad \forall j \in M_i \tag{2.14}$$

$$y_{ij} \ge 0 \qquad i = 1,\dots,m, \qquad \forall j \in M_i \tag{2.15}$$

In the above formulation, the parameter $\Psi_i$ sets a safety level. Ben-Tal and Nemirovski (2000) show that probability of violating any constraint in the optimization model (2.12) is bounded by $\exp(-\Psi_i^2/2)$. Some new set of variables, *y* and *w*, are required for modeling the row uncertainty. The above formulation can be solved using conic quadratic programming and due to computational complexity, it is not appropriate for discrete optimization problems.

In this approach, all variables represent decisions that must be made before the realization of uncertain parameters. On the contrary, Ben-Tal and Nemirovski (2004) propose the Adjustable Robust Counterpart (ARC) in which some of the variables should be determined before the realization of the uncertain parameters (non-adjustable variables), while the other variables could be decided after the realization (adjustable variables).

As an alternative to the work by Ben-Tal and Nemirovski (2004), Bertsimas and Sim (2003, 2004) recommend a restricted uncertainty approach in which only a subset of coefficients are driven to their upper bounds. They propose the following robust counterpart of (2.12):

Max $cx$

subject to

$$\sum_{j} \hat{a}_{ij} x_j + g_i(x, \Gamma_i) \le b_i \qquad i = 1, \ldots, m \tag{2.16}$$

$$g_i(x, \Gamma_i) = \text{Max}\left\{ \sum_{j \in O_i} d_{ij} y_j + (\Gamma_i - \lfloor \Gamma_i \rfloor) d_{it_i} y_{t_i} : O_i \subset M_i, |O_i| = \lfloor \Gamma_i \rfloor, t_i \in M_i \setminus O_i \right\}$$

$$-y_j \le x_j \le y_j \qquad \forall j \tag{2.17}$$

$$y_j \ge 0 \qquad \forall j \tag{2.18}$$

In the above model, for each row $i = 1, \ldots, m$, the set $O_i$, which is a subset of $M_i$ with cardinality $\lfloor \Gamma_i \rfloor$, is identified so that the total deviation occurs at maximum level. In this model, $\Gamma_i$ adjusts the robustness level and it could be fractional. One of the coefficients, which has an index of $t_i$ for row $i$, deviates with an amount of $(\Gamma_i - \lfloor \Gamma_i \rfloor)\, d_{it_i}$.

Their approach has the advantage of applicability to discrete optimization problems. Besides, the robust problem maintains the structure of the deterministic problem, i.e. if the deterministic model is an LP, then the robust model is also an LP. Their approach has been applied to define the robust versions of some well-known combinatorial optimization problems such as the shortest path problem and the knapsack problem; however applicability of Bertsimas and Sim's approach in project scheduling has not been shown, yet.

On the whole, the major advantage of robust optimization over stochastic programming is that the system performance remains under control even in the worst-case conditions. Furthermore, no assumptions regarding the underlying probability distribution of the uncertain data are required. It is most appropriate if there exist "hard constraints", which must be always satisfied no matter what the realization of the data is (see Ben-Tal and Nemirovski, 1999), or if the solution is sensitive to small data perturbations (see Ben-Tal and Nemirovski, 2000). Furthermore, robust optimization should be preferred to model circumstances in which the system should perform well even in the worst cases, such as deciding on the location of fire stations. On the other hand, as worst-case conditions are emphasized in robust optimization, in some cases the expected performance of the generated solutions might be worse when compared to the solutions generated using stochastic programming.

When accurate distributional information is available, stochastic programming has the advantage of incorporating this available distributional information; however stochastic programming models are usually computationally more demanding. In this dissertation, we assume that project managers do not have accurate information about the distribution of random activity durations or costs. Therefore, we employ robust optimization methodology to formulate robust project scheduling models.

### 2.1.3. Other Techniques

Sensitivity analysis, parameter programming and fuzzy programming are the alternative paradigms to address optimization problems under uncertainty. In this section, we discuss them briefly.

In sensitivity analysis, the dependence of model output on input parameters is investigated; generally the effect of small perturbations on the optimal solution is analyzed. Sensitivity analysis is distinctively different from stochastic programming and robust optimization since it is reactive in nature; it does not address uncertainty in the modeling phase.

Parametric programming solves a set of mathematical models over the parameter vector. It is proactive since uncertainty is integrated into the model as a function of the parameter vector. Sensitivity analysis and parametric programming are usually studied together in the literature. The major difference is that: sensitivity analysis models the discrete changes in problem parameters, whereas parametric programming addresses continuous changes. We refer the readers to Gall and Greenberg (1997) for  a detailed examination of sensitivity analysis and parametric programming.

Fuzzy programming has attracted attention of the researchers as an alternative paradigm to address optimization problems under uncertainty since the pioneering work of Bellman and Zadeh (1970). Instead of using random variables, uncertain parameters are modeled as fuzzy numbers and the constraints are defined with the use of fuzzy sets and membership functions. Membership functions might allow some constraint violations and measure the degree of satisfaction of the constraints. For details of the theory and applications of fuzzy programming, we refer the readers to Zimmerman (2001).

In the next section, models and algorithms to hedge against uncertainty for project scheduling problems will be discussed, and applications of the above mentioned techniques in project scheduling will be reviewed.

## 2.2. Project Scheduling under Uncertainty

Deterministic project scheduling assumes that the baseline schedule can be executed as planned. However, during the project execution, both the activity durations and resources are subject to uncertainties. Machine failures, inaccurate time estimates, quality problems and arrival of urgent jobs are common situations that prevent the baseline schedule from being executed as planned. Therefore, how to manage projects in the presence of uncertainty is a crucial question in project management. According to De Meyer et al. (2002), there exist four types of uncertainty in projects and each type requires a different managerial approach. This uncertainty classification is given below.

1. *Variation*: Variation refers to the random deviation in production systems. This type of uncertainty is the most common type in production systems. Machine breakdowns, quality problems and flu epidemics are classical examples. Robust scheduling techniques such as inserting buffers between the activities are used to decrease the effect of variation on project performance.

2. *Foreseen Uncertainty*: Foreseen uncertainty refers to the cases where possible sources of uncertainty are identifiable. To give an example, possible side effects in a drug development project may be predicted before project execution. To model foreseen uncertainty, decision tree based techniques are generally used.

3. *Unforeseen Uncertainty*: Unlike the foreseen uncertainty, sources of uncertainty are not known. To give an example, the famous drug "Viagra" was developed to prevent heart attacks. However, it is widely used for other problems. In drug development phase, nobody could have predicted this application area and high sale figures. Scenario planning is commonly used to model unforeseen uncertainty.

4. *Chaos*: This is the hardest case to manage since project structure may change radically. Crisis management techniques are applied in chaotic situations. Learning and experience become more important than planning. Natural disasters such as earthquake and hurricanes are the typical examples of chaotic situations.

In this dissertation, we focus on uncertainties of the variation type. De Meyer et al. (2002) emphasize the importance of planning and accounting for variation during planning in projects at which variation dominates.

During project execution, validity of the baseline schedule becomes questionable due to structural changes caused by disruptions. The baseline schedule is prepared under the assumption that activity durations are deterministic and resource availability is constant. The realized activity durations and resource availability may differ from the planned values. As a result, actual project performance may vary significantly from the expected performance.

To minimize the effect of unexpected events on project performance, five fundamental scheduling approaches have been discussed in the literature: stochastic scheduling, fuzzy scheduling, sensitivity analysis, reactive scheduling, and robust (proactive) scheduling (Herroelen and Leus, 2005). This classification is depicted in Figure 1. Details of each approach will be given in the following subsections.

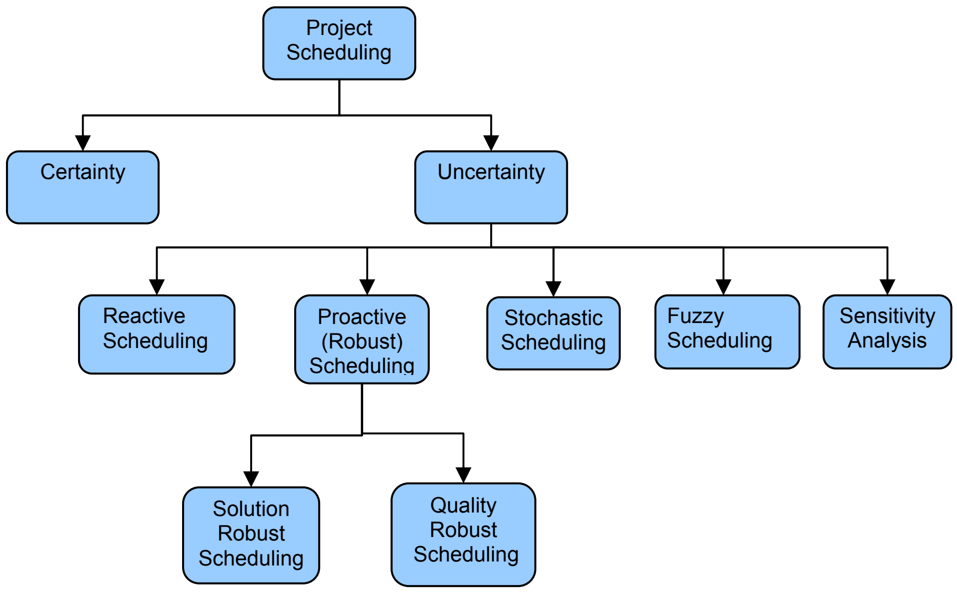

**Figure 1. Taxonomy Based on Herroelen and Leus (2005)**

### 2.2.1. Reactive Scheduling

Modifying or re-optimizing a schedule in the face of disruptions is called reactive scheduling. If a baseline schedule is prepared before execution, this approach is known as predictive-reactive scheduling, on the other hand, the schedule could dynamically be constructed, and this is called dynamic scheduling.

When and how to reschedule are the major questions in reactive scheduling. For timing, two approaches exist: In event driven scheduling, rescheduling is performed when an unexpected event is observed; whereas in the periodic policy, rescheduling is performed at the beginning of each period. Corrective action in the case of disruptions may be taken as either full or partial rescheduling. All the available tasks are rescheduled in full scheduling, whereas in partial scheduling only a part of the current schedule is updated. For further discussion of these approaches and a comprehensive review of applications in machine scheduling the readers are referred to Sabuncuoğlu and Bayız (2000). Even though there are a large number of reactive machine scheduling applications in literature, the reactive scheduling applications in project management are scarce.

Simulation is the most commonly used approach in reactive project scheduling literature. In simulation studies, effects of problem characteristics on performance are tested and impact of rescheduling on performance is analyzed. Full rescheduling is compared with simple repair mechanisms, such as right shifting. Yang (1996) performs a simulation experiment to explore the advantages of rescheduling on makespan minimization. He uses a simulated annealing (SA) based heuristic to generate schedules and shows that SA-based heuristic performs much better than simple dispatching rules. He also demonstrates that the frequency of rescheduling affects the project completion time and that the effect of rescheduling depends on project structure.

Herroelen and Leus (2001) use simulation to show the weaknesses and strengths of critical chain management. They also demonstrate that rescheduling has positive effects on project makespan. They test the effect of scheduling mechanism on makespan. To schedule and reschedule, branch-and-bound and latest finishing time heuristic are used. Performance difference between these two methods is significant; therefore they suggest using the branch-and-bound method.

Van de Vonder et al. (2007b) offer four different predictive-reactive resource-constrained project scheduling procedures. Through simulation, they evaluate these procedures under the combined objective of maximizing the schedule stability and the timely project completion probability. In another study, Van de Vonder et al. (2007a) propose heuristics for repairing resource-constrained project baseline schedules. The effect of multiple activity duration disruptions during project execution on stability is minimized. They also apply to simulation to compare the performances of the heuristics.

Zhu et al. (2005) follow a mathematical programming approach to model uncertainty. They formulate an integer linear program for recovering the project disruptions. They model various disruption alternatives including the disruptions in activity durations, in the network structure, and in resource availabilities are considered. Various recovery options are modeled. In addition to rescheduling, their model allows altering activity modes and increasing resource availabilities. However, these alternatives are costly. They optimize a composite objective function, which is a function of project makespan and stability. The model is solved with a hybrid mixed-inter programming/constraint programming procedure after relaxing some of the constraints. Their model is flexible, as different recovery options are considered; however, it is computationally demanding to solve.

### 2.2.2. Robust Scheduling

In *proactive* or *robust scheduling,* variability is incorporated into the models, and schedules that are less vulnerable to disruptions are sought. Herroelen and Leus (2005) divide schedule robustness into two groups: solution robustness (stability) and quality robustness. We use this classification in this dissertation. The *solution robustness* is defined as the insensitivity of the activity start times with respect to variations in the input data. On the other hand, *quality robustness* is defined as insensitivity of schedule performance such as project makespan with respect to disruptions. Quality robust scheduling aims to construct schedules in such a way that the value of the performance measure is affected as little as possible by disruptions. The total slack concept is closely related to quality robustness, whereas free slack to the stability of a schedule.

The most popular approach of project management aiming for quality robustness is *critical chain project management* (CCPM) that has been introduced by Goldratt (1997) who applied of the theory of constraints (TOC) to project management. TOC is a management philosophy introduced by Goldratt and Cox (1984). It emphasizes identifying and controlling system constraints so as to improve the performance of the overall system. Basic properties of CCPM can be summarized as follows:

1. *Multi-tasking*, or performing multiple tasks in the same time frame, is discouraged in order to minimize total flow time.
2. CCPM tries to eliminate due-date focused behavior. Defining and communicating *project milestones*, which are particularly important project events, are eliminated.
3. CCPM controls buffer usages to monitor project performance. *Buffers* are protection mechanisms against uncertainty in the duration of activities.
4. Safety factors are eliminated from individual activities and aggregated at the end as a project buffer. Aggressive time estimates are used and in this way, staff is forced to increase productivity.

CCPM is also called *Critical Chain Scheduling and Buffer Management*. *Buffer Management* aims to plan and control the buffers. It is an emerging field in project management. CCPM defines three types of buffers:

1. *Project buffer*: This type of buffer is added to the end of the critical chain to prevent possible project delays.
2. *Feeding buffer*: This buffer is added to the end of the paths merging into the critical chain, thus it prevents any possible delay on feeding paths to affect the start time of critical tasks.
3. *Resource buffer*: This buffer works as a warning mechanism to assure that the resources are ready when they are demanded by critical activities.

In the CCPM literature two buffer sizing methods are common: the *50% rule* and the *Root Square Error Method* (*RSEM*). In the 50% rule, half of the total duration of the chain is calculated with safe estimates and taken as the buffer size. The 50% rule is also known as the "*cut and paste method*" in the literature. On the other hand, the RSEM uses two estimates for each task on the feeding chain: the safe estimate and the average estimate. The difference is assumed to be equal to twice the standard deviation of the activity duration. Assuming the independence of task durations in the chain, standard deviation of the sum of activity times is calculated. Twice the standard deviation is used as the buffer size.

In these two methods, buffer size does not depend on project characteristics related to resource availability and network structure. Tukel et al. (2006) propose two heuristics that take into account the number of precedence relationships and resource tightness in buffer sizing. Their heuristics create smaller buffers than both of the two well known methods. However, for large projects with high uncertainty levels, their method results in lower probability of meeting the planned completion times. Therefore, their method may not be accepted as a better method than the other two methods.

Some of the limitations of the CCPM will also be mentioned. The use of aggressive time estimates creates pressure on staff to increase productivity, which may lead to quality problems. Moreover, CCPM does not give attention to resource constrained scheduling; simple heuristic rules are used to determine the critical chain in CCPM software packages. It also does not provide algorithms to solve the resource conflicts that may occur after inserting the buffers into the project baseline. However, Herroelen and Leus (2001) demonstrate that the choice of scheduling and rescheduling algorithms may significantly affect the final makespan.

CCPM focuses on minimizing the makespan and as a second objective it minimizes work in process inventory. During execution, all the activities other than gating activities, that is activities which do not have predecessors, are started as early as possible. Late start rule is only applied to gating tasks. However, scheduling non-gating tasks as early as possible may not be the cost-minimizing strategy. Right shifting some of the non-gating tasks may have positive effects on costs without affecting the robustness. A comprehensive critical examination of the CCPM is given by Raz et al. (2003).

Herroelen and Leus (2003) propose some mathematical programming models to construct stable project schedules. They develop an LP model and some benchmark heuristics. Their LP model allows a single activity disruption, which is duration increase in one activity, during the schedule execution. Leus (2003) extends the model and considers multiple disruption possibilities. Leus and Herroelen (2004) adapt the stability model to the resource constrained networks using resource flow networks. These networks model the number resource units transferred among the activities as a resource flow. In their model, only a single resource type is considered and branch-and-bound method is used to solve the problem.

Van de Vonder et al. (2005, 2006) analyze the trade-off between the quality robustness and solution robustness. They use a scheduling mechanism that is adapted from the float factor model of Tavares (1998). The factor float model shifts activity start times from the earliest start times with the same proportion of the slack values for all the activities. Van de Vonder et al. (2005) relax the resource constraints and concentrate on stability. However their model results in quality robustness as well in the cases where the dummy ending activity has a high weight. Van de Vonder et al. (2006) extend the activity dependent float factor model to the resource constrained environment. In these studies, the quality robustness is measured by the probability that the project will end by the project due date. Lambrechts et al. (2008a) and Van de Vonder et al. (2008) propose heuristics for solution robust scheduling and compared the performances of proposed heuristics using simulation. Al Fawzan and Haouari (2005) develop a bi-objective model for the RCPSP and optimize the solution robustness and makespan. They use a tabu search algorithm for generating the set of efficient solutions.

Proactive-reactive scheduling protects against disruptions by combining a proactive scheduling procedure and a reactive improvement procedure. It may be applied to a project network as follows: The baseline schedule is created by the maximization of a robustness measure so that it involves sufficient safety time to absorb the effects of the disruptions. Although, this baseline schedule will be less sensitive to the disruptions, all possible disruptions may not be anticipated. For this reason, it is better to incorporate reactive scheduling as the second protection mechanism to prevent large performance deviations due to disruptions. We refer the readers to Lambrechts et al. (2008b) for a nice application of this approach.

### 2.2.3. Stochastic Project Scheduling

In *stochastic project scheduling*, the activity durations are modeled as random variables and probability distributions are used. *Stochastic resource-constrained project scheduling problem* (SRCPSP) is the stochastic extension of RCPSP. Stochastic dynamic programming is used to solve the problem. No baseline schedule is created. Scheduling policies (or scheduling strategies) dynamically make scheduling decisions at decision points corresponding to the start time of activities. Stork (2001) proposes exact algorithms, while Golenko-Ginzburg and Gonik (1997, 1998), Tsai and Gemmill (1998) and Ballestin (2008) propose heuristic algorithms for solving the stochastic RCPSP. Fernandez et al. (1998) stress that the previous simulation based approaches unrealistically ignores non-anticipativity as they assumed perfect information for activity durations. They model the problem as a multi-stage decision process. On the other hand, Zhu et al. (2008) define a new problem with uncertain activity durations and use a cost- based objective function. They model the problem using two-stage stochastic programming. The first stage fixes the activity times based on known probabilistic information. The second phase generates the schedule by minimizing the total penalty of deviating from the planned times. In this paper, the authors make an analogy between the project scheduling problem under uncertainty and the traditional newsvendor problem. In project scheduling, the cost of early and late completions is balanced, and a similar trade-off exists between the excess and shortage costs in the traditional newsvendor problem.

Gutjahr et al. (2000) formulate a stochastic multi-mode project scheduling problem by allowing crashing of the modes with some additional costs and modeling activity durations as random variables. They minimize the total expected cost that includes expected tardiness penalty and crashing cost.

### 2.2.4. Fuzzy Project Scheduling

Instead of probability distributions, *fuzzy project scheduling* uses fuzzy membership functions to model activity durations. The advocates of the fuzzy activity duration approach claim that probability distributions for the activity durations are usually unknown due to reasons such as lack of accurate historical data. They also believe that activity durations estimated by human experts are potentially inaccurate. Hapke and Slowinski (1994, 1996) generate a set of schedules applying twelve dispatching rules and select the schedule with the least fuzzy makespan. Wang (2002, 2004) concentrated on product development projects. They use fuzzy set theory to generate robust schedules.

Applications of sensitivity analysis to scheduling problems are very limited in the literature. For a good discussion and relevant examples, the readers are referred to Hall and Posner (2004). We summarize the literature review on project scheduling under uncertainty in Table 1. As illustrated in the table, fuzzy scheduling has rarely been addressed, whereas stochastic scheduling has widely been studied between mid 90's and 2000, and then reactive and robust scheduling has started to be addressed in the literature. In the last years, reactive and robust scheduling have increasing popularity among the researchers.

In the next chapter, we formulate the discrete time/cost trade-off problems, explain its versions in detail and develop solution algorithms to solve the problems exactly.

**Table 1. Summary of Literature on Project Scheduling under Uncertainty**

| Author(s) | Reactive Scheduling | Robust Scheduling | | Stochastic Scheduling | Fuzzy Scheduling |
|---|---|---|---|---|---|
| | | Stability | Quality Robustness | | |
| Hapke and Slowinski (1994, 1996) | | | | | x |
| Yang (1996) | x | | | | |
| Golenko-Ginzburg and Gonik (1997,1998) | | | | x | |
| Fernandez et al. (1998) | | | | x | |
| Tsai and Gemmil (1998) | | | | x | |
| Valls et al. (1998) | | | x | | |
| Gutjahr et al. (2000) | | | | x | |
| Stork (2001) | | | | x | |
| Herroelen and Leus (2001) | | | x | | |
| Wang (2002, 2004) | | | | | x |
| Herroelen and Leus (2003) | | x | | | |
| Leus and Herroelen (2004) | | x | | | |
| Al-Fawzan and Haouari (2005) | | | x | | |
| Zhu et al. (2005) | x | | | | |
| Van de Vonder et al. (2005, 2006) | x | x | | | |
| Tukel et al. (2006) | | | x | | |
| Van de Vonder et al. (2007a) | x | | | | |
| Van de Vonder et al. (2007b) | x | x | | | |
| Ballestin (2008) | | | | x | |
| Chtourou and Haouari. (2008). | | | x | | |
| Cohen et al. (2008) | | | x | | |
| Lambrechts et al. (2008a) | x | x | | | |
| Lambrechts et al. (2008b) | x | | | | |
| Van de Vonder et al. (2008) | | x | | | |
| Yamashita et al. (2008) | | | x | | |
| Zhu et al. (2008) | | | | x | |

# CHAPTER 3

# MODELS AND EXACT APPROACHES FOR THE DETERMINISTIC DISCRETE TIME/COST TRADE-OFF PROBLEMS

In project management, it is often possible to reduce the duration of some of the activities and therefore expedite the project duration with additional costs. This time/cost trade-off has been widely studied in the literature since the critical path method (CPM) was developed in 1950s. The majority of these studies address the linear, continuous time/cost relationships (Icmeli et al., 1993). In this dissertation, we consider the discrete version of the problem, or the discrete time/cost trade-off problem (DTCTP).

Three versions of the DTCTP have been studied in the literature: the deadline problem (DTCTP-D), the budget problem (DTCTP-B) and the efficiency problem, (DTCTP-E). In DTCTP-D, given a set of time/cost pairs (mode) and a project deadline, each activity is assigned to one of the possible modes in such a way that the total cost is minimized. Conversely, the budget problem minimizes the project duration while meeting a given budget. On the other hand, DTCTP-E is the problem of constructing efficient time/cost solutions over the set of feasible project durations. This study concentrates on the deadline and the budget problems.

Despite its importance in practice, the research in the discrete version of the problem, DTCTP, is rather new due to its inherent computational complexity; it has been shown to be strongly NP-hard for general activity networks by De et al. (1997). In their comprehensive review paper, De et al. (1995) discuss the exact and approximate solution strategies. The readers are referred to the studies of Robinson (1975), Hindelang and Muth (1979), Harvey and Patterson (1979), and Demeulemeester et al. (1996,1998) for exact algorithms and to Skutella (1998), Akkan et al. (2005), and Vanhoucke and Debels (2008) for approximate algorithms to solve DTCTP.

In the next subsection, we formally define an extension of the DTCTP and present the model formulation. Moreover some special cases of the model are examined. Firstly, we investigate a special case that is suitable to model Build-Operate-Transfer (BOT) projects. Detailed information about the characteristics of this type of projects is given. Furthermore, we examine the two well-known special cases, the deadline and the budget problems, in Sections 3.2 and 3.3 respectively. In these sections, we propose Benders Decomposition based solution algorithms and give some computational results. Finally, some concluding remarks are presented in Section 4.

## 3.1. A Practical Extension and Application Area

In this section, we propose an extension of DTCTP that incorporates the notion of opportunity cost into the model. The model is suitable for Build-Operate-Transfer (BOT) projects in which consideration of opportunity costs is crucial. The BOT model describes the situation in which a public service or an infrastructure investment is made and operated for a specific period of time by a private enterprise and then transferred to a public institution. The operating period is sufficiently long, typically 10 to 20 years, so that the investment is paid off and the private enterprise realizes a tangible profit.

The BOT model has several advantages for the public sector as it functions as an alternative financing mechanism in undertaking large investment projects. Secondly, it promotes foreign investment inflow. In some cases, technology transfer accompanies foreign investment, implying that some benefits that cannot be expressed in monetary terms can be achieved. Finally, it can be assumed that the private enterprises as profit maximizers operate the system in the most productive way and use the required up-to-date technology. From the private sector's perspective, the BOT model tends to reduce demand and credit risks because the government is generally the sole customer. This means that the risks associated with insufficient demand and ability to pay is minimal. Due to the mutual benefits explained above, the BOT contracts are becoming popular and widely accepted in both developed and developing countries. One of the application areas of this model is private toll roads. First, the private enterprise constructs the roads, and operates them for a period of time and then transfers the right to operate these roads to the public. In this case, the enterprise can elongate the operating period via completing construction earlier and hence increases the profit. For a detailed analysis of BOT contracts and further application areas, the reader is referred to Walker and Smith (1995).

As the BOT projects favor early completions, incorporating opportunity costs into the model is essential. Scheduling with due dates is extensively studied in machine scheduling literature, whereas the research addressing project environments is limited. The existing scheduling studies incur a penalty cost both in cases when the project finishes earlier and later than the due date. Furthermore the existing time/cost trade-off problems are restrictive models in the sense that either they place a limit on the spending or enforce a deadline for the project completion. Though these models address the time/cost trade-off, they do not perform a cost/benefit analysis. To give an example, the DTCTP-B aims to exploit the given budget such that the project is completed as early as possible. However, there may be cases in which cost of expedition may significantly surpass the benefits gained from early completions. In contrast, in BOT projects, a small surpass in the given budget could save days which in turn drastically increases profitability.

The extended version we propose uses monetary units as a common basis to measure the performance of the project. It performs a cost benefit analysis. That is, if the burden of the unit increase in parameters is less than the yield, the increase is allowed. Indeed, in real life applications, the budget is an estimate of cash flows and whenever it is profitable to do so, it is elaborated in time via allocating extra funds created through various financial alternatives, such as loans, bonds, or stocks. Furthermore, the extended version considers due dates and penalty costs of tardiness instead of the deadlines, hence allowing tardiness.

We call the model "generalized discrete time/cost trade-off problem" (GDTCTP) since it encompasses both DTCTP-D and DTCTP-B. When the opportunity cost and interest paid for budget overruns are zero and the tardiness penalty is very large, the model reduces to the DTCTP-D. It also reduces to the DTCTP-B when the budget coefficient and tardiness penalty is zero and the interest paid for budget overruns are very large. In the next section, GDTCTP is formally defined and the model formulation is presented.

**A Generalized Model:**

A project with *n* activities is represented by an activity-on-node graph, i.e. *G* (*N, A*); where *N* is the set of nodes, and $A \subset N \times N$ is the set of immediate precedence constraints on the activities. Two dummy activities corresponding to the project start and end, activity *0* and activity *n+1*, are included in the network; i.e. *N* = {*0,1,2……,n+1*}. A set of time/cost pairs (modes) is given for each activity and an activity *j* is performed in one of the possible modes, i.e. $m \in M_j = \left\{ \underline{m}_j, ......, \overline{m}_j \right\}$.

Activity *j* performed in mode *m*, is characterized by a processing time $p_{jm}$ and cost $c_{jm}$ and without loss of generality, it can be assumed that, for each $j \in N$, $m < m'$ implies $p_{jm} > p_{jm'}$ and $c_{jm} < c_{jm'}$. Given a project due date, $\delta$, and budget, $B_0$, an execution mode is to be selected for each activity in such a way that the total cost function including opportunity costs and penalty costs are minimized. In the cost function, $\alpha$, $\phi$ and $\zeta$ denote the opportunity cost per unit time, the interest paid for budget overruns and the tardiness penalty per unit time, respectively. $\beta$ is a weighting factor for the budget.

A mixed integer-programming model of the problem could be stated as follows:

$$\text{Min } \alpha C_{n+1} + \beta B + \phi Max\left\{0, B - B_0\right\} + \zeta Max\left\{0, C_{n+1} - \delta\right\} \tag{3.1}$$

subject to

$$\sum_{m \in M_j} x_{jm} = 1, \ \forall j \in N \tag{3.2}$$

$$C_j - C_i - \sum_{m \in M_j} p_{jm} x_{jm} \geq 0 \qquad \forall (i,j) \in A \tag{3.3}$$

$$B - \sum_{j \in N} \sum_{m \in M_j} c_{jm} x_{jm} \geq 0 \tag{3.4}$$

$$C_j \geq 0 \qquad \forall j \in N \tag{3.5}$$

$$B \geq 0 \tag{3.6}$$

$$x_{jm} \in \{0,1\} \qquad \forall m \in M_j, \forall j \in N \tag{3.7}$$

The non-negative continuous decision variable $C_j$ denotes the completion time of activity *j*. The binary decision variable $x_{jm}$ assigns modes to the activities and takes the value 1 if mode *m* is chosen for activity *j*. Otherwise it is 0 (3.7). While minimizing the total cost including opportunity costs (3.1), a unique mode should be assigned to each activity (3.2); precedence constraints should not be violated (3.3). A continuous variable *B* is used to denote the total cost spent to complete the activities (3.4).

Note that when $\alpha = 0$, $\phi = 0$, $\zeta = M$ (where *M* is a "very big" number), the problem reduces to DTCTP-D, and when $\beta = 0$, $\phi = M$, $\zeta = 0$ the model reduces to DTCTP-B. Furthermore, we also address the special case where $\beta = 1$, and propose to use this special case to model Build-Operate-Transfer (BOT) projects. This case encourages early completions because the model incorporates the opportunity cost. Integrating additional variables *T* and *W* into the model, we convert the problem into a linear form and formulate the BOT Model as follows:

$$\text{Min } \alpha C_{n+1} + B + \phi \text{W} + \zeta T \tag{3.8}$$

subject to

$$\sum_{m \in M_j} x_{jm} = 1, \ \forall j \in N \tag{3.9}$$

$$C_j - C_i - \sum_{m \in M_j} p_{jm} x_{jm} \geq 0 \qquad \forall (i,j) \in A \tag{3.10}$$

$$T - C_{n+1} \geq -\delta \tag{3.11}$$

$$W - B \geq -B_0 \tag{3.12}$$

$$B - \sum_{j \in N} \sum_{m \in M_j} c_{jm} x_{jm} \geq 0 \tag{3.13}$$

$$W \geq 0,\ T \geq 0,\ B \geq 0;\ C_j \geq 0 \qquad \forall j \in N \tag{3.14}$$

$$x_{jm} \in \{0,1\} \qquad \forall m \in M_j, \forall j \in N \tag{3.15}$$

In the next section, the DTCTP-D is formally defined and solved exactly for large project instances.

## 3.2. The Deadline Problem

### 3.2.1. Problem Definition and Model Formulation

In DTCTP-D, given a project deadline of $\delta$ time units, a set of execution modes is to be selected for each activity so that the total operational cost is minimized. A mixed integer-programming model of the deadline problem, DTCTP-D, could be stated as follows:

$$\text{Min} \sum_{j \in N} \sum_{m \in M_j} c_{jm} x_{jm} \tag{3.16}$$

subject to

$$\sum_{m \in M_j} x_{jm} = 1,\ \forall j \in N \tag{3.17}$$

$$C_j - C_i - \sum_{m \in M_j} p_{jm} x_{jm} \geq 0, \qquad \forall (i,j) \in A \tag{3.18}$$

$$C_{n+1} \leq \delta \tag{3.19}$$

$$C_j \geq 0, \qquad \forall j \in N \tag{3.20}$$

$$x_{jm} \in \{0,1\} \qquad \forall m \in M_j, \forall j \in N \tag{3.21}$$

While minimizing the total cost (3.16), a unique mode should be assigned to each activity (3.17), precedence constraints should not be violated (3.18), and the deadline should be met (3.19). Given the NP-hard nature of the problem, it is difficult to solve large-size problems exactly. We propose an exact algorithm based on Benders Decomposition to solve the large scale real-life problems. In the following section, we present the formal description the solution algorithm.

### 3.2.2. Overview of Benders Decomposition

Benders (1962) introduces this decomposition algorithm to solve specially structured large-scale linear and mixed integer programs. The basic idea behind this method is to decompose the problem into two simpler problems: the first part, called the master problem (MP), solves a relaxed version of the problem and generates trial values for the integer variables and a lower bound for a minimization objective. The second problem, called the subproblem (SP), is the original problem with the values of the integer variables temporarily fixed by the MP. The dual of the SP inserts cuts into the master problem and obtains an upper bound for a minimization objective. We use the following general formulation to illustrate the decomposition approach:

$$\text{Min } c_1x_1 + c_2x_2 \tag{3.22}$$

subject to

$$A_1x_1 + A_2x_2 \geq b \tag{3.23}$$

$$x_2 \in X_2, x_1 \geq 0, \tag{3.24}$$

The variables are partitioned into two groups such that $x_2 \in X_2$ refers to the complicating variables. To give an example, in a mixed integer program, $x_1$ and $x_2$

could be defined as the continuous and integer variables, respectively. In that case, $X_2$ defines the feasible set of discrete variables. Note that (3.22-3.24) is equivalent to:

$$\operatorname*{Min}_{x_2} \{c_2 x_2 + \operatorname{Min}\{ c_1 x_1 : A_1 x_1 \geq b - A_2 x_2\} : x_2 \in X_2\} \tag{3.25}$$

Note that when $x_2$ is fixed, the inner formulation becomes a LP. Taking the dual of the LP with dual vector $w$, (3.25) becomes:

$$\operatorname*{Min}_{x_2} \{c_2 x_2 + \operatorname*{Max}_{w \geq 0} \{ \mathrm{w}(b - A_2 x_2) : wA_1 \leq c_1 \} : x_2 \in X_2\} \tag{3.26}$$

The inner formulation, called dual SP, could be either bounded or unbounded. If bounded, the solution is one of the extreme points $w^k$, $k = 1...K$. Otherwise extreme rays $v^r$, $r = 1,...,R$ are generated. As a result, (3.26) could be reformulated as:

$$\mathrm{Min}\ \ c_2 x_2 + z \tag{3.27}$$

subject to

$$\mathrm{z} \geq w^k(b - A_2 x_2) \qquad k = 1,...,K. \tag{3.28}$$

$$v^r(b - A_2 x_2) \leq 0 \qquad r = 1,...,R. \tag{3.29}$$

$$x_2 \in X_2,\ z \geq 0 \tag{3.30}$$

The above reformulation is called Benders MP, and (3.28) and (3.29) are called *feasibility* and *optimality cuts*, respectively. A relaxation approach is required in order not to enumerate all the extreme points and rays, and the cuts are generated when needed. The MP and the SP are solved iteratively until the lower bound converges to the upper bound.

Benders Decomposition has been used to solve many combinatorial optimization problems; specifically, successful applications of this methodology on network design are numerous (Costa, 2005). However, only a few applications for project scheduling problems are reported in the literature and these are discussed below:

Maniezzo and Mingozzi (1999) use a heuristic algorithm based on Benders decomposition to solve the Multi-Mode Resource Constrained Project Scheduling Problem (MRCPSP) approximately. In the MRCPSP, each activity is assigned to one of the possible modes and the starting time of each activity is determined so that project completion time is minimized while precedence and resource constraints are satisfied. The DTCTP-B is a special case of MRCPSP in which only a single nonrenewable resource (money) is in use, whereas the general MRCPSP allows the use of both renewable and nonrenewable resources. Maniezzo and Mingozzi (1999) propose a mathematical formulation to the MRCPSP, relax some of the constraints and solve the relaxed version with a heuristic procedure based on Benders Decomposition. In their model, the MP corresponds to the assignment of one mode to each activity, while the SP is a single mode RCPSP. They solve both the MP and SP approximately.

Erengüç et al. (1993) use Benders Decomposition to solve the time/cost trade-off problem with discounted cash flows, which is indeed a combination of the DTCTP and the payment-scheduling problem. Kuyumcu and Garcia-Diaz (1994) solve the project compression problem with concave or convex piecewise linear cost-duration functions. All project scheduling applications mentioned above are tested with small to medium size problem instances.

Our research in this section differs from these studies regarding both the problems addressed and the solution approach. We incorporate mechanisms into the Benders algorithm to accelerate the solution process thus we solve problems of realistic sizes. In the following subsections, we present the details of the Benders Decomposition algorithm for the DTCTP-D.

### 3.2.3. Benders Reformulation of the Deadline Problem

As activity durations are nonnegative and $C_0$ refers to a reference project start time, we can relax the non-negativity constraints on $C_j$ and reformulate the DTCTP-D as follows:

$$\text{Min } \sum_{j\in N}\sum_{m\in M_j} c_{jm}x_{jm} \tag{3.31}$$

subject to

$$\sum_{m\in M_j} x_{jm} = 1,\ \forall j\in N \tag{3.32}$$

$$C_j - C_i - \sum_{m\in M_j} p_{jm}x_{jm} \ge 0,\ \ \forall (i,j)\in A \tag{3.33}$$

$$C_{n+1} - C_0 \le \delta \tag{3.34}$$

$$C_j \text{ u.r.s} \qquad \forall j\in N \tag{3.35}$$

$$x_{jm}\in\{0,1\},\forall m\in M_j,\ \forall j\in N \tag{3.36}$$

Given a vector $x\in X^0 = \{\, x_{jm}\in\{0,1\}\ \forall m\in M_j, \forall j\in N\ :\ \sum_{m\in M_j} x_{jm} = 1\ \forall j\in N\,\}$, consider the following polyhedron:

$$P(x) = \{\mathrm{C}\in R^{n+2} : C_j - C_i - \sum_{m\in M_j} p_{jm}x_{jm} \ge 0\ \forall (i,j)\in A, C_{n+1} - C_0 \le \delta\,\}.$$

Note that any vector $x$ is feasible for the DTCTP-D, if and only if $x\in X^0$ and $P(x)\ne\varnothing$. Therefore, the set of feasible solutions to the DTCTP-D could be formulated as:

$$X^D = \{\, x\in X^0 : P(x)\ne\varnothing\,\} \tag{3.37}$$

The following lemma provides an alternative path formulation of $X^D$:

**Lemma 3.1:** The set of feasible solutions of the DTCTP-D could be reformulated as: $X^D = \{x \in X^0 \colon \sum_{(i,j)\in A} \sum_{m\in M_j} p_{jm} w^s_{ij} x_{jm} \le \delta, s = 1,\dots,S\}$, where $S$ refers to the total number of paths between node *0* to *n+1* in *G* (*N, A*) and $w^s_{ij}$ is the incidence vector of the path $s, \forall (i,j) \in A,\ s = 1,\dots,S$.

**Proof:** Given any vector, $\bar{x} \in X^0$, define the following linear feasibility seeking problem, namely, $Q(\bar{x})$:

$$\text{Min} \sum_{(i,j)\in A} \gamma_{ij} \tag{3.38}$$

subject to

$$\gamma_{ij} + C_j - C_i \ge \sum_{m\in M_j} p_{jm} \overline{x_{jm}} \qquad \forall (i,j) \in A \tag{3.39}$$

$$-C_{n+1} + C_0 \ge -\delta \tag{3.40}$$

$$\gamma_{ij} \ge 0 \tag{3.41}$$

In this formulation, $\gamma_{ij}$ are the auxiliary variables that are used for feasibility checking and $\gamma^*_{ij}$ are the optimal values. Note that $P(\bar{x}) \ne \varnothing$, if and only if $\gamma^*_{ij} = 0$ $\forall (i,j) \in A$. Let $w_{ij}$ and $v$ be the dual variables associated with the constraint sets (3.39) and (3.40), respectively. Deriving the dual of the $Q(\bar{x})$ using these variables, we obtain the dual sub-problem $SP(\bar{x})$, given below. In this formulation, *Su*(*i*) and *Pr*(*i*) define the set of immediate successors and predecessors of activity *i*, respectively.

$$\text{Max} \sum_{(i,j)\in A} \sum_{m\in M_j} p_{jm} \overline{x_{jm}}\, w_{ij} - \delta \text{v} \qquad (3.42)$$

subject to

$$\sum_{k\in Su(j)} w_{jk} - \sum_{k\in Pr(j)} w_{kj} = 0 \qquad j = 1,\dots,n \qquad (3.43)$$

$$\sum_{k\in Su(0)} w_{0k} - v = 0 \qquad (3.44)$$

$$\sum_{k\in Pr(n+1)} w_{kn+1} - v = 0 \qquad (3.45)$$

$$0 \le w_{ij} \le 1 \qquad \forall (i,j)\in A \qquad (3.46)$$

$$v \ge 0 \qquad (3.47)$$

We know that if $P(\overline{x}) \neq \varnothing$, then the optimal solution to the primal problem, $Q(\overline{x})$, has zero objective function at optimality. Hence by strong duality, the dual problem, $SP(\overline{x})$ has also zero objective function at optimality.

If any vector $\{(w^*, v^*) : v^* \neq 0\}$ is optimal to $SP(\overline{x})$, $\frac{1}{v^*}(w^*, v^*)$ is feasible and has zero objective function as well. Hence, we can set $v = 1$, and $SP(\overline{x})$ becomes equivalent to the following network flow problem, of which the feasible region will be denoted with $W$.

$$\text{Max} \sum_{(i,j)\in A} \sum_{m\in M_j} p_{jm} \overline{x_{jm}}\, w_{ij} \qquad (3.48)$$

subject to

$$\sum_{k\in Su(j)} w_{jk} - \sum_{k\in Pr(j)} w_{kj} = 0 \qquad j = 1,\dots,n \qquad (3.49)$$

$$\sum_{k\in Su(0)} w_{0k} = 1 \qquad (3.50)$$

$$\sum_{k\in Pr(n+1)} w_{kn+1} = 1 \qquad (3.51)$$

$$0 \le w_{ij} \le 1 \qquad \forall (i,j)\in A \qquad (3.52)$$

To denote shortly,

$$SP(\bar{x}) \equiv \text{Max } \{ \sum_{(i,j)\in A} \sum_{m\in M_j} p_{jm} \overline{x_{jm}} \, w_{ij} : w_{ij} \in W \ \forall (i,j) \in A \} \tag{3.53}$$

Due to the total unimodularity of the constraint matrix, if there is an optimal solution to *SP*($\bar{x}$)*,* then there is an optimal solution with *{0,1}* dual variables, and this solution has the form :

$$w^s_{ij} = \begin{cases} 1 & \text{if (i,j) belongs to path s} \\ 0 & \text{o/w} \end{cases} \tag{3.54}$$

As *SP(*$\bar{x}$*)* has also zero objective function at optimality if $P(\bar{x}) \neq \emptyset$ and using (3.54),

$$\text{Max } \left\{ \sum_{(i,j)\in A} \sum_{m\in M_j} p_{jm} \overline{x_{jm}} \, w^s_{ij} : \text{s} = 1,\ldots,\text{S} \right\} - \delta = 0 \tag{3.55}$$

Combining (3.37) and (3.55), we get

$$X^D = \{ x \in X^0 : \sum_{(i,j)\in A} \sum_{m\in M_j} p_{jm} w^s_{ij} x_{jm} \leq \delta, \ s = 1,\ldots,S \}$$

Q.E.D.

As an immediate result of Lemma 3.1, the following corollary could be stated:

**Corollary 3.1:** DTCTP-D could be formulated as follows:

$$\text{Min } \sum_{j\in N} \sum_{m\in M_j} c_{jm} x_{jm}$$

subject to

$$\sum_{(i,j)\in E}\sum_{m\in M_j} p_{jm} w_{ij}^{s} x_{jm} \le \delta \qquad s = 1,\ldots,S$$

$$\sum_{m\in M_j} x_{jm} = 1 \qquad \forall\, j \in N \tag{3.56}$$

$$x_{jm} \in \{0,1\} \qquad \forall\, m \in M_j, \forall j \in N$$

Enumerating all the paths is burdensome, so we use a relaxation approach and generate the constraints as needed. In the subsequent sections, we discuss the details of the proposed Benders Decomposition algorithm to solve the problem exactly in an efficient manner. Next, we provide the solution algorithm.

**The Algorithm:**

Introduce an additional index *t* to the notation to denote the values at iteration *t*.

```
1. Start with an initial solution, $\overline{x^1} \in X^0$; set $t = 1$.
2. Solve the $SP\,(\overline{x^t})$ : Max { $\sum_{(i,j)\in A}\sum_{m\in M_j} p_{jm}\,\overline{x_{ij}^t}\, w_{ij}$ : $w \in W$}
   If $SP\,(\overline{x^t})$ is unbounded (primal infeasible) then
        Get an extreme ray ($\overline{w^t}$)
        $X^t = X^{t-1} \cap \{ x \in X^0 : \sum_{(i,j)\in A}\sum_{m\in M_j} p_{jm}\,\overline{w_{ij}^t}\, x_{jm} \le \delta \}$
   Else
        If ($t > 1$)
         Stop and report $\overline{x^t}$ as the optimal solution.
    End if
3. Solve the relaxed master problem, $MP^t$: $z^t$ = Min { $\sum_{j\in N}\sum_{m\in M_j} c_{jm} x_{jm}$ : $x \in X^t$}.
4. Set $LB = z^t$, $t = t + 1$, $\overline{x^t} = x^{t-1}$.
5. Return to Step 2
```

Note that solving $SP(\bar{x})$ is equivalent to determining the length of the critical path ($C_{n+1}$) with respect to the given mode assignments and comparing it with the deadline. For example if $C_{n+1} \leq \delta$, the problem is feasible, otherwise it is infeasible. In order to demonstrate the steps of the algorithm, we solve an illustrative example and report in Appendix A. Benders Decomposition is known to exhibit slow convergence. Therefore, we include several features to accelerate the convergence and solve large instances to optimality.

### 3.2.4. Algorithmic Enhancements

The computational efficiency of Benders Algorithm depends on three issues: (i) the number of iterations; (ii) the time needed to solve the SP at each iteration; (iii) the time needed to solve the MP at each iteration.

In order to decrease the number of iterations, we add multiple cuts at each iteration. Instead of finding the critical path and adding a single feasibility cut at each iteration, we determine the longest *K* paths and insert *K* cuts. To find these paths, we use a modification of the well-known Yen's (1971) K-shortest loopless paths (KSP) algorithm. KSP lists *K* shortest paths between a given source-destination pair in the directed graph without revisiting the same node (loopless paths). Note that since PERT networks are acyclic, critical path problems may be solved as shortest path problems with cost parameters equal to the negative of the activity durations. *K* is in fact a parameter, which affects the computational efficiency. It is observed that as *K* increases, the number of iterations required attaining global convergence decreases, whereas the time and the computer effort demanded for solving each MP at each iteration increases. In order to decide on the level of parameter K, we performed 30 pretests involving problems with different sizes. As a result, we decided on the following strategy: $K$ is fixed to 2 in the first 10 iterations, later $K$ is increased at each iteration until $K = 15$. This strategy prevents generating large number of constraints which are nonbinding at optimality.

In order to solve $SP_1$, we use the modified label-correcting algorithm of Ahuja et al. (1993). This algorithm is shown to be efficient and to run in $O(|A|)$ time, when finding the shortest path from the source to destination for acyclic networks (Ahuja et al. (1993)). The major difficulty in this decomposition lies in the solution of the MP, which may become a very large 0–1 programming problem. To solve the MP efficiently, we propose the following enhancements:

**1)** We use preprocessing to eliminate some of the mode combinations so that the number of decision variables in the MP is decreased. We apply the preprocessing techniques of Akkan et al. (2005) to eliminate long and short modes.

**2)** As McDaniel and Devine (1977) suggested, we solve the LP relaxations first and generate cuts from the fractional solutions. Then, integrality constraints are added and the algorithm is restarted.

**3)** Not all of the master problems are solved to optimality, i.e. feasibility cuts are generated from heuristic solutions. However, the last MP should be solved to optimality.

**4)** In order to solve the MP iterations efficiently, we develop a customized branch-and-cut algorithm that groups the variables as special ordered sets (SOS) and that incorporates an effective neighborhood search strategy.

In the following proposition, we demonstrate that there is no need to solve all the MP iterations exactly.

**Proposition 3.1:** It is not necessary to solve all the MP iterations of the DTCTP-D to optimality except in the last iteration.

**Proof:** At each MP iteration, new feasibility cuts are inserted. As the feasibility cuts correspond to a subset of the path set, each MP solves a relaxation of the DTCTP-D. This statement holds even though the MP iterations are solved approximately as every feasible solution generates a path in the network. For any IP, if an optimal solution of the relaxed problem is feasible for the original IP, then it is also optimal for the original IP (Wolsey, 1998, Proposition 2.3). Therefore, if the last MP iteration is solved exactly and leads to a feasible solution for the DTCTP-D, the solution is optimal for the DTCTP-D as well.

Q.E.D.

Using the previous proposition, the algorithm is modified as follows:

We solve the MP iterations approximately at a relative optimality tolerance level of *ε = 2 %* until feasibility is satisfied. In the test runs, we tried *ε = 1, 2, 3, 4, 5 %*. The algorithm run fastest with *ε = 2 %*. This means that at each iteration branch-and-bound algorithm is truncated and the up-to-now best feasible solution, the incumbent solution, is used. The incumbent solution of each MP is guaranteed to be within *2 %* of the optimal value. In the following sections, we report the quality and the computational effort requirements of this solution as the heuristic solution. We demonstrate that this solution could be used as a reliable approximate solution and it is much faster to reach this solution. In this case, however, the cost of the relaxed MP does not necessarily provide a lower bound on the cost of the optimal solution. Using the resulting feasible solution as the initial solution, we restart the algorithm. From this point on, we solve the MP iterations to optimality so that they provide a lower bound on the cost of the optimal solution. In the following section, we describe the customized branch-and-cut algorithm to solve the MP exactly.

### 3.2.5. A Branch-and-Cut Procedure

The branch-and-cut method combines the branch-and-bound method and the cutting plane algorithm. A cutting plane is a valid inequality (an inequality that is satisfied by all feasible solutions) that violates some feasible points of the LP relaxation. A cutting plane, which cuts off a given LP solution, can strengthen the LP relaxation. The relaxed feasible region becomes smaller, but the IP feasible region does not change. In a cutting plane algorithm, the LP relaxation of the new formulation is solved with the inserted cuts and the cuts are generated when required. In a branch-and-cut algorithm, violated cuts are added at the nodes of the branch-and-bound tree. If no violated cuts are found or the effectiveness of the cutting planes in improving the LP bound decreases, the node is branched further.

In a classical branch-and-bound algorithm, for each node the LP relaxation is solved, and a fractional variable (if there is one) on which to branch is chosen. In this dissertation, we branch on sets of variables instead of branching on individual variables. A set of variables, in which at most one variable in the set is allowed to be nonzero, forms a "Special Ordered Set of Type 1" ($SOS_1$). In the DTCTP, the constraint set (3.17) represents an $SOS_1$. Branching strategies have large impacts on the size of the branch-and-bound tree and the computation time. Branching on $SOS_1$ instead of a single variable has some advantages. For example the tree is more balanced, and if the variables in the $SOS_1$ are ordered, fractional LP solutions may be used to determine the variables to be set to one in the optimal solution. Linderoth and Savelsbergh (1999) explain these advantages and give an illustrative example.

We assign an order to the variables among $SOS_1$ by using the activity durations as weights. The variables are divided into two sets: the first subset includes the variables that have weights greater than the average weight, as calculated by using the solution of the relaxed problem. The second subset consists of variables that have weights less than the average value. Two branches are created by setting the variables in each subset to zero.

In our algorithm, the MP includes knapsack constraints, or the sums of binary variables with nonnegative coefficients less than or equal to a nonnegative right-hand side. Previous computational studies show that cover cuts are effective for problems with knapsack constraints (Atamturk and Savelsbergh, 1999). Cutting planes derived from knapsack constraints may be strengthened by using $SOS_1$. These strengthened constraints are called Generalized Upper Bound Constraints (GUBs). Gu et al. (1998) examines this type of constraints in depth. Combining the knapsack and GUB constraints usually result in stronger cutting planes (GUBs).

We develop the branching tree using CPLEX. It allows detailed control over the solution process and customizes the branch-and-cut tree (Atamturk and Savelsbergh (1999)). Branching nodes are selected using a best-estimate search strategy, a strategy in which one chooses the node estimated to have the best feasible integer solution obtainable. In our branch-and-cut tree, we apply $SOS_1$ branching and insert the GUB cuts up to five times the total number of constraints.

Adding an excessive number of cuts works to improve the global lower bound. In order to improve the global upper bound, we integrate a heuristic that works to find good feasible solutions early in the search process. For this purpose, we use the Relaxation Induced Neighborhoods (RINS) heuristic. This heuristic explores the neighborhood using information contained in the relaxation of the MIP model. In other words, it fixes variables which have the identical value in the incumbent and the LP solution, and solves the remaining MIP with limited branching. Danna et al. (2005) show RINS to be efficient in finding good feasible solutions early in the process.

### 3.2.6. Experimentation and Computational Results

Mainly three factors affect the difficulty of solving a particular problem instance: the network structure, the number of modes per activity, the tightness of the deadline. To test the presented algorithm, we use the test-bed 1 of Akkan et al. (2005), which contains large sized problems. The network structure is commonly defined with two parameters: the complexity index, CI, and the coefficient of network complexity, CNC. CI is a measure developed by Bein et al. (1986) to assess how far the given network is from being series–parallel. It is defined to be the minimum number of node reductions required to reduce a given two-terminal directed acyclic graph into a single-arc graph, when used together with series and parallel reductions. Assessing the distance of a given network from being series–parallel is important because DTCTP with series–parallel graphs could be solved quickly (Demeulemeester et al.,1996). The second complexity measure, CNC, is developed by Pascoe (1966) and defined to be the ratio of the number of arcs to the number of nodes.

The number of modes per activity is randomly generated with discrete uniform distribution using two intervals: U[2, 10] and U[11, 20]. To compute the deadline for each instance, first the minimum possible project duration, *Tmin* (length of the critical path with modes that have the shortest duration) and the maximum possible project duration, *Tmax* (length of the critical path with that have the longest duration), are calculated. Then, the deadline is determined as follows:

$$\delta = Tmin + \theta\,(Tmax - Tmin), \text{ where } \theta \in \{0.15,\ 0.30,\ 0.45\} \qquad (3.57)$$

In this function the parameter $\theta$ defines the tightness of the deadline. Each mode of an activity defines a time/ cost pair. A cost function characterizes the time/cost relationship of the modes. In this test bed, concave (ccv), convex (cvx), and neither concave nor convex cost functions (hyb) are used to generate the cost figures. Table 2 summarizes the parameters and the related levels of the instances in the test bed.

**Table 2. Experimental Setting for Solving Deterministic Problems**

| **Parameters** | **Level(s)** |
|---|---|
| **Number of Nodes** | 17 |
| **CI** | 13, 14 |
| **CNC** | 5 , 6, 7, 8 |
| **Number of Modes** | U[2,10], U[11,20] |
| **Deadline Parameter (θ)** | 0.15, 0.30, 0.45 |
| **Cost Function (CF)** | ccv, cvx, hyb |

CNC and CI are the determinants of the number of activities in a project. This experimental setting involves projects that contain from 85 to136 activities. We implement all the algorithms in C programming language on a Sun UltraSPARC 12x400 MHz workstation with 3 GB RAM. Optimization software CPLEX 9.1 is used to solve the linear and integer programs at each step of the master problem. We use 432 problem instances, 144 project settings with 3 replicates, from the data set provided by Akkan et al. (2005). Table 3 and Table 4 summarize the experimental results when the number of modes lies in the interval U[2,10] and U[11,20], respectively. The results of the exact procedure are presented under the column labeled with "Optimum". The average number of linear and integer master problems solved and the average CPU time to solve the instances are reported under the columns "LP Iter.", "IP Iter." and "CPU(s)", respectively. On the other hand, we depict the results of the approximate method under the column labeled "Truncated Solution", within which the percentage of problem instances that the optimal solution is found, the average and maximum percentage of deviation from the optimal solution and the average percentage reduction in CPU time are reported under the columns "Ins Opt (%)", "Avg. Dev (%)", "Max Dev (%)", "Dec. CPU (%)", respectively.

**Table 3. Summary of Computational Results (Number of Modes $\in$ U[2, 10])**

| | | **Optimum** | | | **Truncated Solution** | | | |
|---|---|---|---|---|---|---|---|---|
| | | **LP Iter.** | **IP Iter.** | **CPU (s)** | **Ins Opt (%)** | **Avg. Dev (%)** | **Max Dev (%)** | **Dec. CPU (%)** |
| **CI** | **13** | 14.30 | 13.25 | 2187.81 | 57.14 | 0.13 | 0.87 | 34.00 |
| | **14** | 11.55 | 15.63 | 2120.74 | 49.51 | 0.22 | 1.27 | 43.86 |
| **CNC** | **5** | 9.51 | 9.24 | 147.05 | 39.22 | 0.22 | 0.97 | 47.25 |
| | **6** | 13.06 | 12.08 | 627.09 | 51.92 | 0.20 | 0.94 | 38.01 |
| | **7** | 13.85 | 16.17 | 2857.73 | 54.35 | 0.15 | 1.11 | 40.09 |
| | **8** | 15.53 | 23.16 | 6069.70 | 71.05 | 0.13 | 1.27 | 30.08 |
| **θ** | **0.15** | 18.27 | 21.76 | 4730.22 | 76.19 | 0.06 | 0.82 | 29.45 |
| | **0.30** | 12.43 | 13.18 | 1645.35 | 45.90 | 0.18 | 0.97 | 45.41 |
| | **0.45** | 7.65 | 8.70 | 60.98 | 36.51 | 0.30 | 1.27 | 43.62 |
| **CF** | **ccv** | 12.76 | 13.63 | 3264.09 | 69.35 | 0.08 | 0.72 | 27.01 |
| | **cvx** | 12.05 | 15.66 | 1447.30 | 28.13 | 0.31 | 1.27 | 56.67 |
| | **hyb** | 13.59 | 14.36 | 1757.56 | 62.30 | 0.15 | 0.94 | 33.96 |

**Table 4. Summary of Computational Results (Number of Modes $\in$ U[11, 20])**

| | | Optimum | | | Truncated Solution | | | |
|---|---|---|---|---|---|---|---|---|
| | | LP Iter. | IP Iter. | CPU (s) | Ins Opt (%) | Avg. Dev (%) | Max Dev (%) | Dec. CPU (%) |
| **CI** | **13** | 17.34 | 14.34 | 14745.24 | 18.18 | 0.38 | 1.39 | 77.32 |
| | **14** | 14.62 | 17.24 | 16249.18 | 18.00 | 0.51 | 1.57 | 76.33 |
| **CNC** | **5** | 14.38 | 12.88 | 8117.71 | 18.00 | 0.50 | 1.57 | 75.22 |
| | **6** | 17.20 | 17.68 | 14875.06 | 19.51 | 0.36 | 1.20 | 77.30 |
| **θ** | **0.15** | 25.65 | 24.43 | 32288.42 | 26.09 | 0.25 | 0.95 | 72.67 |
| | **0.30** | 15.09 | 15.52 | 7419.13 | 15.15 | 0.40 | 1.04 | 82.97 |
| | **0.45** | 9.60 | 8.43 | 808.51 | 17.14 | 0.60 | 1.57 | 72.03 |
| **CF** | **ccv** | 17.10 | 12.06 | 8138.58 | 45.16 | 0.17 | 1.44 | 56.89 |
| | **cvx** | 13.15 | 15.50 | 7945.58 | 3.85 | 0.71 | 1.56 | 93.11 |
| | **hyb** | 16.24 | 17.41 | 16378.87 | 5.88 | 0.49 | 1.57 | 80.76 |

We apply a Fixed Effects ANOVA test to the results in order to find out the variance effect of the experimental design factors to CPU time when the number of modes lies in the interval U[2, 10]. Test results including the coefficient of determination ($R^2$) values of the tests are reported in Table 5.

**Table 5. The Effect of the Factors on the CPU Time: ANOVA Test**

| Source | DF | Sum of Squares | Mean Square | F | p |
|---|---|---|---|---|---|
| CI | 1 | 10.981 | 10.981 | 3.240 | 0.073 |
| CNC | 3 | 454.539 | 151.513 | 44.72 | 0.000 |
| CF | 2 | 8.124 | 4.062 | 1.20 | 0.304 |
| θ | 2 | 1040.568 | 520.284 | 153.56 | 0.000 |
| CI*CNC | 3 | 90.889 | 30.296 | 8.94 | 0.000 |
| CI*CF | 2 | 6.492 | 3.246 | 0.96 | 0.386 |
| CI* θ | 2 | 5.982 | 2.991 | 0.88 | 0.415 |
| CNC*CF | 6 | 40.314 | 6.719 | 1.98 | 0.070 |
| CNC* θ | 6 | 34.746 | 5.791 | 1.71 | 0.121 |
| CF* θ | 4 | 5.884 | 1.471 | 0.43 | 0.784 |
| Error | 184 | 623.406 | 3.388 | | |
| Total | 215 | 2321.927 | **$R^2 = 0.731$** | | |

Before the test is performed, the necessary assumptions for ANOVA are checked and Box-Cox transformation is utilized to restore constant error variances and normality assumptions. From the experimental results given above, the following conclusions could be derived:

- When Tables 3 and 4 are compared, it is evident that the average number of modes is highly influential on computational effort. The number of modes identifies the number of binary decision variables; therefore it is significantly effective on problem complexity.
- ANOVA test reveals that the tightness of the deadline ($\theta$) and the network complexity, measured with CNC and CI, significantly influence the computational effort. There is an inverse relationship between the project deadline and the computational effort. As the project deadline becomes looser, the number of MIP iterations and the time needed to solve the MP at each iteration both drastically decrease. As a result, the problem instances with loose deadlines could be solved more quickly.
- The instances with complex network structure, i.e. networks with larger CNC and CI, demand more computational effort. Tables 3 and 4 indicate that CNC is more influential on problem complexity when compared to CI. The reason is that CNC directly sets the number activities in the network and as the number of activities increase the number of binary variables increases as well. Additionally there exists an interaction between network complexity parameters.
- Finally, the approximate truncation-based procedure generates solutions that are close to the optimal solution quickly. Moreover, in majority of the problem instances of the test bed, it obtains the optimal solution. One

major difference of the approximate results summarized in Tables 3 and 4 is that the reductions in CPU times become significant as the number of modes increases. Therefore, especially for complex problem instances truncation-based heuristic provides a good solution alternative.

In the next section, we address another special case of the DTCTP, namely the budget problem, and solve it exactly for large project instances.

## 3.3. The Budget Problem

### 3.3.1. Model Formulation

A mixed integer-programming formulation of the budget problem DTCTP-B can be given as follows:

$$\text{Min } C_{n+1} \tag{3.58}$$

subject to

$$\sum_{m \in M_j} x_{jm} = 1, \; \forall j \in N \tag{3.59}$$

$$C_j - C_i - \sum_{m \in M_j} p_{jm} \, x_{jm} \geq 0 \qquad \forall (i,j) \in A \tag{3.60}$$

$$\sum_{j \in N} \sum_{m \in M_j} c_{jm} x_{jm} \leq B_0 \tag{3.61}$$

$$C_j \geq 0 \qquad \forall j \in N \tag{3.62}$$

$$x_{jm} \in \{0,1\} \qquad \forall m \in M_j \; \forall j \in N \tag{3.63}$$

In this formulation, the project completion time, *$C_{n+1}$,* is minimized (3.58), while the given budget, *$B_0$*, is met (3.61).

## 3.3.2. Benders Reformulation of the Budget Problem

**Corollary 3.2:** Using the path formulation defined by Lemma 3.1, DTCTP-B could be formulated as follows:

Min $z$

subject to

$$z - \sum_{(i,j)\in E} \sum_{m\in M_j} p_{jm} \overline{w_{ij}^{s}} x_{jm} \geq 0 \qquad s = 1,\dots,S$$

$$\sum_{j\in N} \sum_{m\in M_j} c_{jm} x_{jm} \leq B_0$$

$$\sum_{m\in M_j} x_{jm} = 1 \qquad \forall j \in N \tag{3.64}$$

$$x_{jm} \in \{0,1\} \qquad \forall m \in M_j, \forall j \in N$$

$$z \geq 0$$

**Proof:** As demonstrated in the proof of Lemma 3.1, given a vector $x \in X^0$, the length of the longest path, which will be defined with a continuous variable $z$ from now on, could be formulated as:

$$\mathrm{z} = \mathrm{Max}\left\{ \sum_{(i,j)\in A} \sum_{m\in M_j} p_{jm}\, x_{jm} w_{ij}^{s} - \delta : s = 1,\dots,S \right\} \tag{3.65}$$

Combining 3.65 with the assignment and budget constraints, (3.64) formulates the DTCTP-B.

Q.E.D.

For a given solution vector, the dual sub-problem turns out to be a critical path problem, which is a characteristic of the reformulation of the deadline problem as well. The difference is that at each iteration instead of feasibility cuts, optimality cuts are inserted. Since there is no constraint imposed on project completion time, feasibility is maintained in each iteration. In order not to enumerate all the paths, we use a relaxation approach and generate the constraints when needed. We propose the following Benders Decomposition Algorithm to solve the DTCTP-B.

1. Start with an initial solution, $\overline{x^1} \in X^0$; set $LB = -\infty$ , $UB = \infty$, $t = 1$.
2. Solve the $SP^t$ ( $\overline{x^t}$ ): Max $\{\sum_{i\in N}\sum_{j\in S(i)}\sum_{m\in M_j} p_{jm}\overline{x^t_{jm}}w_{ij} : w \in W\}$

   Get an extreme point $\overline{w^t}$ , set $UB = Min$ $\{UB, \sum_{i\in N}\sum_{j\in S(i)}\sum_{m\in M_j} p_{jm}\overline{x^t_{jm}}\ \overline{w_{ij}^{\ t}}\ \}$

   *If (UB = LB)*

   Stop and report $\overline{x^t}$ as the optimal solution.

   Else

   $$X^t = X^{t-1} \cap \left\{ x \in X^0 : z \geq \sum_{i\in N}\sum_{j\in S(i)}\sum_{m\in M_j} p_{jm}\overline{w^t_{ij}}x_{jm} \right\}$$
3. Solve the relaxed master problem, $MP^t$: $z^t$ = Min $\{z: x \in X^t\}$. Let $x^t$ be the optimal solution.
4. Set $LB = z^t$, $t = t + 1$, $\overline{x^t} = x^{t-1}$.
5. Invoke Preprocessing
6. Return to Step 2

As Benders Decomposition is known to converge slowly, we propose some enhancements to accelerate convergence and to solve large-scale instances exactly.

### 3.3.3. Algorithmic Enhancements

The first enhancement is integration of a preprocessing technique that serves to eliminate some of the modes, hence reducing the number of binary variables.

At each iteration of Benders Algorithm, MP solves a relaxed version of the problem and generates a LB for a minimization objective. LB is non-decreasing with respect to the number of iterations. We call the preprocessing after each MP iteration. In the following theorem, we show that some of the long modes, the modes with a long processing time, could be redundant and hence could be eliminated. The following theorem expresses the mode elimination procedure.

**Theorem 3.1:** Given any *LB* for the DTCTP-B, for any activity *i* if there exists a mode *m* such that the length of the longest possible path that passes through activity *i* is less than or equal to the *LB*, then there exists an optimal solution such that activity *i* is performed either at mode *m* or at a mode with longer duration.

**Proof:** Let $X_B$ denote the set of feasible mode assignments for DTCTP-B and $l_{uv}(x)$ be the length of the longest path between node *u* and *v,* and $A(x,i)$ be the length of the longest path that does not pass through activity *i* given the mode assignment, $x \in X_B$.

The longest possible path that passes through activity *i* is the mode assignment such that mode *m* is assigned to activity *i* and the modes with longest durations are assigned to the remaining activities. We will denote this assignment as $x^L$ hereafter. Note that $x^L \in X_B$, otherwise as the modes with longest durations have the least cost, mode *m* would never become feasible and could be eliminated.

Let us assume that there exists an optimal solution $x'$ such that $x_{im'} = 1$, where $m < m'$; without loss of generality, for any $i \in N$, $m < m'$ implies $p_{im} > p_{im'}$ and $c_{im} < c_{im'}$.

Given that, $p_{im} + l_{0i}(x^L) + l_{in+1}(x^L) \leq LB,$ (3.66)

we have,

$$p_{im'} + l_{0i}(x') + l_{in+1}(x') < p_{im} + l_{0i}(x^L) + l_{in+1}(x^L) \leq LB \leq C_{n+1}(x') \qquad (3.67)$$

Keep all the mode assignments in $x'$ other than activity $i$ and assign mode $m$ to activity $i$. Call this new assignment $y$, *i.e.* $y_{im} = 1$ and $y_{jm} = x_{jm}$ $\forall j \in N \setminus i$.

Since $c_{im} < c_{im'}$, the mode assignment remains feasible, i.e. $y \in X_B$. For this new assignment,

$A(y,i) = A(x',i) \leq C_{n+1}(x')$ (3.68),

$C_{n+1}(y) = \text{Max}\ \{ p_{im} + l_{0i}(y) + l_{in+1}(y), A(y,i) \}$ (3.69),

and $l_{0i}(y) + l_{in+1}(y) = l_{0i}(x') + l_{in+1}(x')$ (3.70)

Using (3.68) and (3.70), we have

$p_{im} + l_{0i}(y) + l_{in+1}(y) = p_{im} + l_{0i}(x') + l_{in+1}(x') \leq p_{im} + l_{0i}(x^L) + l_{in+1}(x^L) \leq LB \leq C_{n+1}(x')$, hence due to (3.68), (3.69) , $C_{n+1}(y) \leq C_{n+1}(x')$

Therefore, when (3.66) holds for any optimal solution such that $x_{im'} = 1$: $m < m'$, there exists an alternative optimal solution such that $x_{im''} = 1$: $m'' = m$ or $m'' < m$. Hence, all the modes $m' > m$ could be eliminated.

Q.E.D.

In order to solve the DTCTP-B efficiently, we apply a branch-and-cut algorithm similar to the one described in Section 3.2.2. However, there exist some differences. The basic difference is that an initial phase that aims to provide a good UB quickly is integrated. This UB serves to cut off some of the branches in the subsequent MIP iterations; the nodes of which the optimal solution to the LP relaxation at that node is worse than the UB are not further branched.

All the values of the parameters of the experimentation are set through a number of 30 pretests involving problems with various sizes. To obtain a good UB quickly, we insert many feasibility cuts in the first phase, i.e. the longest $K_1 = 100$

paths are determined and, 100 cuts (only violated ones) are inserted at each stage. Furthermore in this phase, we solve the MP iterations approximately at a relative optimality tolerance level of $\varepsilon_1 = 4\ \%$. Note that $\varepsilon_1$ should be larger than the tolerance level used for solving the deadline problem in order to get an UB quicker. In the test runs, we try $\varepsilon = 3, 4, 5\ \%$. The fastest solutions are achieved with $\varepsilon = 4\ \%$.

Every MIP iteration in the budget problem generates a feasible solution, and the feasible solution with the smallest objective value found so far, called the incumbent solution, provides an UB. However in the deadline problem, the feasibility is not guaranteed at each iteration. In the first stage branch-and-bound algorithm is truncated and the objective value of the best solution found in that iteration is recorded as $F_R$. The incumbent solution at the end of phase one is reported as the *truncated heuristic solution*. This solution is guaranteed to be within $\varepsilon_1 = 4$ % of the optimal value, and we test the quality and efficiency of this solution with computational experiments.

We move to the second phase when the truncated solution of the relaxed problem exceeds the upper bound, i.e. $F_R \geq UB$. This phase starts with only cuts generated from the LP relaxations and the algorithm is restarted with the last solution obtained at the end of phase 1. From now on, we set $K_2 = 25$, $\varepsilon_2 = 2$ % i.e. the longest 25 paths are determined and, 25 cuts (only violated ones) are inserted at each stage. Note that the cuts could be generated from any feasible solution; hence until $F_R \geq UB$, the MP iterations are solved approximately to generate the cuts. Afterwards, the MP is solved exactly until optimality is proven, i.e. $UB = LB$.

## 3.3.4. Computational Results

We generate 120 problem instances, corresponding to 24 project settings with 5 replicates, from the data set provided by Akkan et al. (2005). To compute the budget parameter for each instance, first the minimum possible project cost, *Cmin* (total cost with cheapest modes) and the maximum possible project cost, *Cmax* (total cost with most expensive modes), are calculated. Then, the budget is set as follows:

$$B_0 = Cmin + \theta\,(Cmax - Cmin), \text{ where } \theta = 0.15 \qquad (3.71)$$

For comparison purposes, we solve the hardest instances in the data set, hence we concentrate on the instances which the number of modes lie in the interval U[11, 20]. According to the experimentation results discussed in Section 3.2.7., these instances have been shown to demand the highest computational effort. In order to assess the efficiency of the proposed algorithm, we solve the MIP formulation given in 3.58 - 3.63 with optimization software CPLEX 9.1 and compare the results. A maximal time limit of three hours is set for each exact procedure. We classify the problems with respect to the coefficient of network complexity (CNC), which is the ratio of the number of arcs to the number of nodes. Table 6 and Table 7 summarize the experimental results for the instances with $CNC = \{5, 6\}$ and $CNC = \{7, 8\}$ respectively. A great majority of the instances with $CNC = \{5, 6\}$ could be solved exactly within the time limits (Table 6), however both methods could solve only *23.33 %* of the instances when $CNC = 7$, and 10 % of the instances when $CNC = 8$.

We present the results of the Benders Decomposition-based exact procedure and CPLEX implementation under the columns labeled with “BENDERS DECOMPOSITION” and “CPLEX”, respectively; the percentage of problem instances that the optimal solution is found within time limit and the average CPU time of the problem instances for which an optimal solution is found with Benders algorithm are reported under the columns “Ins Opt (%)” and “CPU(s)”, respectively. CPLEX CPU time is assumed to be three hours for instances that CPLEX could not solve within time limit, hence average CPLEX CPU times are underestimated. Additionally, we report the percentage of modes eliminated, average and maximum optimality gaps (only the problem instances for which an optimal solution is found within the time limit) of the Benders Decomposition based exact procedure under the columns “Mode Elim (%)”, “Avg Gap (%) ”, and “Max Gap (%)”.

On the other hand, the results of the approximate method are depicted under the column called “Truncated Solution”, the average of the percentage deviations from the optimal solution,  (only the problem instances for which an optimal solution is found) and the average CPU time in seconds are reported under the columns “Dev (%)”, “CPU(s)”, respectively.

**Table 6. Summary of Computational Results (*CNC = 5, 6*)**

| Cost Function | CI | CNC | CPLEX | | BENDERS DECOMPOSITION | | | | | TRUNCATED SOLUTION | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | Ins Opt (%) | CPU(s) | Ins Opt (%) | Mode Elim (%) | CPU(s) | Avg Gap (%) | Max Gap (%) | Dev (%) | CPU(s) |
| CCV | 13 | 5 | 100.00 | 984.61 | 100.00 | 15.63 | 506.45 | - | - | 0.94 | 30.30 |
| | | 6 | 60.00 | 1143.62 | 60.00 | 12.12 | 735.93 | 0.84 | 1.04 | 0.97 | 248.96 |
| | 14 | 5 | 100.00 | 96.86 | 100.00 | 12.29 | 255.87 | - | - | 1.01 | 24.542 |
| | | 6 | 100.00 | 1334.25 | 100.00 | 9.89 | 706.17 | - | - | 0.83 | 77.13 |
| CVX | 13 | 5 | 20.00 | 9281.64 | 100.00 | 9.76 | 4368.34 | - | - | 1.81 | 9.83 |
| | | 6 | 0.00 | 10800.00 | 20.00 | 7.31 | 7077.55 | 0.38 | 0.49 | 1.04 | 180.35 |
| | 14 | 5 | 80.00 | 6288.21 | 100.00 | 7.98 | 1983.08 | - | - | 1.59 | 20.63 |
| | | 6 | 0.00 | 10800.00 | 40.00 | 5.30 | 5719.84 | 0.38 | 0.47 | 0.62 | 377.49 |
| HYB | 13 | 5 | 80.00 | 3680.55 | 100.00 | 15.10 | 186.29 | - | - | 1.72 | 19.27 |
| | | 6 | 60.00 | 5691.62 | 100.00 | 11.26 | 826.30 | - | - | 1.76 | 93.90 |
| | 14 | 5 | 100.00 | 1409.49 | 100.00 | 10.56 | 459.28 | - | - | 1.56 | 18.13 |
| | | 6 | 40.00 | 6663.28 | 100.00 | 8.26 | 2676.92 | - | - | 1.32 | 435.93 |

**Table 7. Summary of Computational Results (*CNC* = *7, 8*)**

| Cost Function | CI | CNC | BENDERS DECOMPOSITION | | |
|---|---|---|---|---|---|
| | | | Mode Elim (%) | Avg Gap (%) | Max Gap (%) |
| CCV | 13 | 7 | 10.10 | 2.32 | 3.39 |
| | | 8 | 8.18 | 3.11 | 3.2 |
| | 14 | 7 | 8.14 | 2.56 | 3.29 |
| | | 8 | 8.28 | 2.09 | 3.44 |
| CVX | 13 | 7 | 6.54 | 1.64 | 2.61 |
| | | 8 | 4.70 | 1.99 | 3.20 |
| | 14 | 7 | 4.32 | 2.05 | 2.44 |
| | | 8 | 4.35 | 2.47 | 3.22 |
| HYB | 13 | 7 | 9.10 | 0.95 | 2.19 |
| | | 8 | 6.94 | 0.78 | 1.58 |
| | 14 | 7 | 6.17 | 1.05 | 1.56 |
| | | 8 | 6.77 | 1.38 | 2.27 |

Fixed Effects ANOVA test is applied to the results in order to find out the variance effect of the experimental design factors to CPU time (only for the problem instances for which an optimal solution has been found) and the number of modes eliminated (for all instances). The treatment levels are fixed for these following factors: CNC, CI, and type of cost function (CF). Furthermore, the interactions between these factors are investigated. Before the test is performed, the necessary assumptions for ANOVA are checked and Box-Cox transformation is utilized to restore constant error variances and normality assumption. Test results including the coefficient of determination ($R^2$) values of the tests are reported in Table 8 and Table 9.

**Table 8. The Effect of the Factors on the CPU Time: ANOVA Test**

| Source | DF | Sum of Squares | Mean Square | F | p |
|---|---|---|---|---|---|
| CI | 1 | 3.067 | 3.067 | 1.400 | 0.244 |
| CNC | 1 | 15.838 | 15.838 | 7.240 | 0.011 |
| CF | 2 | 75.945 | 37.973 | 17.350 | 0.000 |
| CI*CNC | 1 | 17.667 | 17.667 | 8.070 | 0.007 |
| CI*CF | 2 | 5.696 | 2.848 | 1.300 | 0.284 |
| CNC*CF | 2 | 4.899 | 2.450 | 1.120 | 0.337 |
| Error | 38 | 83.181 | 2.189 | | |
| Total | 47 | 235.037 | $R^2 = 0.647$ | | |

**Table 9. The Effect of Factors on Number of Modes Eliminated: ANOVA Test**

| Source | DF | Sum of Squares | Mean Square | F | p |
|---|---|---|---|---|---|
| CI | 1 | 5.480 | 5.480 | 36.890 | 0.000 |
| CNC | 3 | 22.463 | 7.488 | 50.410 | 0.000 |
| CF | 2 | 19.927 | 9.963 | 67.070 | 0.000 |
| CI*CNC | 3 | 1.554 | 0.518 | 3.490 | 0.019 |
| CI*CF | 2 | 0.197 | 0.099 | 0.660 | 0.517 |
| CNC*CF | 6 | 0.211 | 0.035 | 0.240 | 0.963 |
| Error | 102 | 15.152 | 0.148 | | |
| Total | 119 | 64.985 | $R^2 = 0.767$ | | |

We also illustrate the relationship between design factors to CPU time and to the number of modes eliminated in Figure 2 and Figure 3, respectively. It is noticeable that both computational effort and effectiveness of the preprocessing algorithms depends on CNC and cost type significantly.

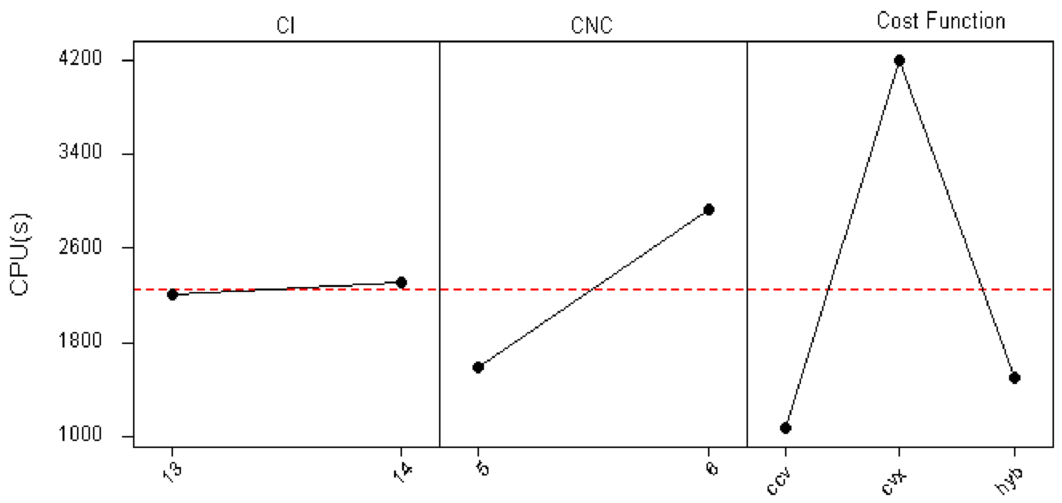

**Figure 2. The Effect of Design Factors on the CPU Time: Main Effects Plot**

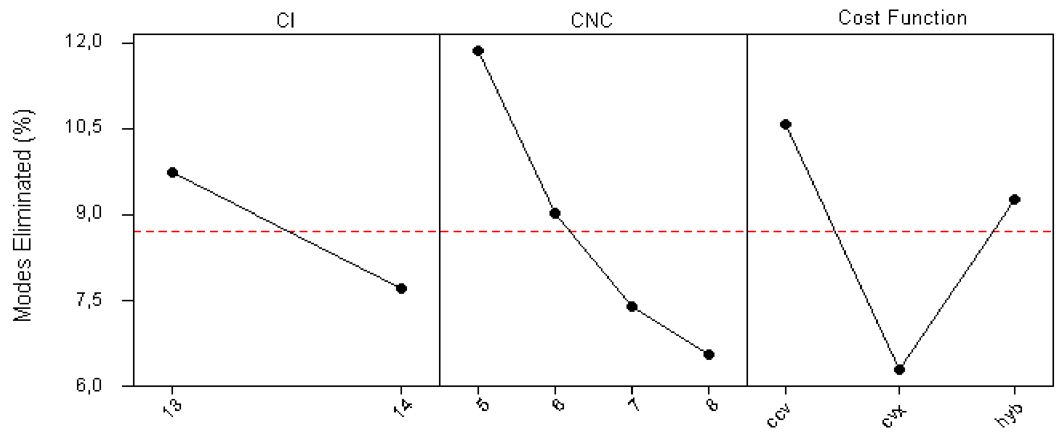

**Figure 3. The Effect of Design Factors on Mode Elimination: Main Effects Plot**

From the experimental results given above, the following conclusions could be derived:

- There are two measures of network complexity: CI and CNC. No significant effect of CI on computational effort is observed, that is a result in line with Akkan et al. (2005)'s finding for approximate solutions of DTCTP-D. However, it is easier to eliminate the modes of networks that have structures close to being series-parallel; hence CI is significantly effective on the performance of the preprocessing method. On the other hand, the instances with larger CNC involve more activities, hence the larger the CNC, the larger the problem complexity.

- The type of cost function is influential on both the computational effort and also on the effectiveness of the preprocessing method. The convex cost functions demand more computational effort and this is mainly due to the fact that less of the modes could be eliminated with preprocessing for these types of problem instances. This is illustrated in Figure 3.
- Finally, the truncation-based heuristic generates solutions that are very close to the optimal solution fast. Besides, in most of the instances for which an optimal solution is found, it finds the optimal solution. Therefore, especially for harder instances, it represents a good solution alternative.

## 3.4. Conclusions

The DTCTP is a well-known project scheduling problem with practical implications. In this chapter, we mainly investigated two versions of the deterministic problem: deadline and budget versions. We have proposed an exact algorithm to solve these versions of realistic sizes. The major contribution of the work in this section lies in the decomposition approach and the developed branch-and-cut procedure. We have included several features to accelerate the convergence into the solution algorithm and in this way; we manage to solve large instances, project networks with up to 136 activities, to optimality.

Computational experiments are performed to measure the efficiency of the algorithm under various problem settings. Mainly three factors are affecting the difficulty in solving a particular problem instance: the network structure, the number of modes per activity and the tightness of the deadline. The major advantage of the proposed algorithm is that the optimal solutions can be obtained even in projects with complex network structures, large number of modes and tight deadlines.

In the next chapter, in order to address project environments more realistically, we relax the complete information and deterministic environment assumptions. We incorporate uncertainty into the analysis and propose robust optimization models for the DTCTP.

# CHAPTER 4

# ROBUST OPTIMIZATION MODELS FOR THE DISCRETE TIME/COST TRADE-OFF PROBLEM

Existing studies on the DTCTP assume complete information and a deterministic environment; however projects are subject to different sources of uncertainty. Uncertainty may arise from activity durations, activity costs, resource availabilities and project network structure. This research addresses the uncertainty in activity costs. In practice, fluctuations in the exchange rates, in the factor prices or in the resource usages result in variability in costs. These fluctuations threaten achievement of project cost objectives. Therefore, it is essential to develop systematic methods to generate robust project schedules, which are less sensitive to uncertainty. For this purpose, we propose three robust optimization models for the DTCTP-D. In these models, we model the uncertainty via interval costs. The first model uses the restricted uncertainty approach of Bertsimas and Sim (2004), whereas the alternative novel models, Model 2 and Model 3, account for activity slacks, and criticality of the activities. The models will be explained further in the subsequent sections.

We develop an exact algorithm based on Benders Decomposition to solve the first model. Computational experiments are carried out to show the efficiency of this approach and evaluate the performance of the algorithm under various problem settings. Due to the complexity of the other two models, we develop a tabu search-based heuristic algorithm for solving the problem instances. Furthermore, we propose some robustness measures and evaluate the robustness of the schedules generated by the proposed models using these measures. The main contribution of the research in this section is the incorporation of uncertainty into a practically relevant project scheduling problem, the modeling approach and the development of customized solution algorithms. Furthermore, to the best of our knowledge, this research is the first application of robust optimization to the DTCTP.

In Section 4.1, we review the applications of robust optimization in project scheduling. In Section 4.2, we propose a robust scheduling model and a Benders Decomposition-based solution algorithm for the robust DTCTP-D and present some computational results. In section 4.3 two alternative models are proposed. These models are compared with the model proposed in section 4.2 by using some robustness measures.

## 4.1. Robust Optimization and Project Scheduling

Robust optimization has attracted many researchers and has been applied to some well-known combinatorial optimization problems such as the shortest path and the knapsack problems during the last decade (see Section 1.3.2 for a detailed discussion of the methodology and for the related applications). However, it has been implemented in only a few project scheduling problems. Valls et al. (1998) examine a special resource constrained project scheduling problem (RCPSP) in which the activities might be interrupted for an uncertain period. Yamashita et al. (2008) address the resource availability cost problem (RACP), which minimizes a non-decreasing discrete cost function of resources under the constraint that the project has to be finished by a given deadline. They propose two alternative models: the first model minimizes the sum of the mean and variance of the costs, whereas the second one minimizes the maximum regret function.

Both Valls et al. (1998) and Yamashita et al. (2008) follow a scenario-based approach, where a scenario represents a realization of the duration of the activities. On the other hand, Cohen et al. (2008) use interval uncertainty in their recent robust scheduling study. This study addresses the effects of uncertainty on the continuous time-cost trade-off problem and model the robust problem using the ARC methodology of Ben-Tal and Nemirovski (2004). Our research differs from the previous studies in the literature regarding both the problem addressed and uncertainty modeling approach followed.

In the next section, a robust optimization model and a solution algorithm for the robust DTCTP-D are proposed.

## 4.2. Robust Discrete Time/Cost Trade-Off Problem with Interval Data

In many real-life projects a tardiness penalty or an opportunity cost may be incurred for each additional time unit the project is late. The cost may include explicit monetary charges, foregone revenue, lost profits, or goodwill losses. Due to these potential costs and early completion benefits, organizations seek on-time completion and aggressively monitor actual progress of these activities. The model proposed in this section addresses project environments in which timely completion of project activities is crucial. Build-Operate-Transfer (BOT) projects are good examples that favor early completions. Since early completion is advantageous in these types of projects, timeliness of the activities is essential.

When deviations from the baseline plan are observed and they are judged to threaten the completion of these activities on time, typically extra resources such as additional workers or extra machinery are assigned to these activities. As the activity durations generally depend on the resource allocation, it is usually possible to achieve time plan goals by allocating additional resources. These additional allocations create fluctuations in the amount of resources allocated to each activity and result in cost uncertainty. They also seriously affect the profitability of the projects. From this point of view, protection against deviations in total cost becomes the key concern of project managers. Hence, in this chapter, we fix the activity durations and assume that activity costs can take values in the intervals, i.e. $c_{jm} \in \left[\underline{c_{jm}}, \overline{c_{jm}}\right], \forall m \in M_j, \forall j \in N$. However, in order to address cost uncertainty robust optimization will be used and worst-case conditions will be examined.

The traditional minmax (absolute robustness) criterion focuses on the worst-case alternative, which corresponds to the scenario where each cost $c_{jm}$ is given by $\overline{c_{jm}}$, the upper bound of the corresponding interval. Optimization with respect to absolute robustness criterion is equivalent to the classic DTCTP-D with $c_{jm} = \overline{c_{jm}}$. However, this approach of robustness is extremely pessimistic and rather unrealistic. A more realistic idea would be modeling the uncertainty only over a subset of the scenario space. One recent application of this restriction idea is the robust discrete optimization approach proposed by Bertsimas and Sim (2004). They assume that only a subset of the uncertain parameters is allowed to deviate from their estimates;

in other words, only $\Gamma$ of activity cost parameters (out of a total of $n$) involve random behavior. If $\Gamma = 0$, the influence of the cost deviations is ignored and the deterministic problem with nominal cost values is obtained. In contrast, if $\Gamma = n$, maximal cost deviations are considered and the problem becomes a minmax optimization problem. Therefore, $\Gamma$ can be regarded as a parameter that reflects the pessimism level of the decision maker. High values of this parameter indicate a risk-averse decision making behavior.

### 4.2.1. Model 1

In this section, a MIP model for DTCTP-D using Bertsimas and Sim's approach is presented. We assume that at most $0 \le \Gamma \le n$ activities have cost values at their upper bounds and the remaining $n\text{-}\Gamma$ coefficients are forced to be deterministic, i.e. they are set to nominal values: $c_{jm} = \dfrac{\overline{c}_{jm} + \underline{c}_{jm}}{2}$ and $d_{jm} = \overline{c}_{jm} - c_{jm}$. The restricted uncertainty model, which will be called Model 1, could be expressed by the following nonlinear formulation:

$$f(\Gamma) = \text{Min}\left\{ \sum_{j \in N} \sum_{m \in M_j} c_{jm} x_{jm} + \text{Max}\left\{ \sum_{j \in N} \sum_{m \in M_j} d_{jm} x_{jm} u_j : \sum_{j \in N} u_j \le \Gamma, u \in B^n \right\} : x \in X^D \right\} \quad (4.1)$$

In this formulation, the set of coefficients, which are subject to uncertainty, are determined by the binary vector $u$, i.e. $B^n$ refers to $n$ dimensional binary vector and $u$ chooses $\Gamma$ of the activities which have the largest cost deviations among $n$ activities. This is a restrictive uncertainty approach as cost deviations for only a subset of the activities are considered. For demonstration purposes, we use the simple network in Figure 4, which is adapted from the example of De et al. (1995). The project has a deadline of $\delta = 6$. Each activity has two mode alternatives and for each mode alternative, the triplet, ($p_{jm}$, $c_{jm}$, $\overline{c_{jm}}$) values are shown above the nodes.

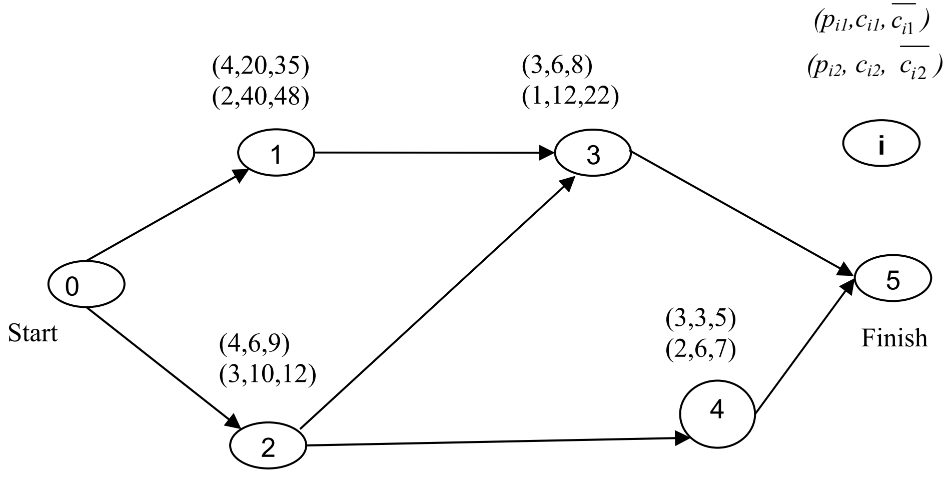

**Figure 4. The Example Network (Robust Problem)**

For this small problem, the objective function values of the deterministic and robust problems are displayed in Table 10. Optimal solutions are marked with a "*". The rows of the table illustrate the feasible mode combinations for the activities.

**Table 10. Comparison of Robust and Deterministic Solutions**

| | **Activity** | | | | $C_5$ | $f(0)$ | $d_{jm} = \overline{c_{jm}} - c_{jm}$ | | | | **Robust Objective:** $f(\Gamma)$ | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | *1* | *2* | *3* | *4* | | | $j=1$ | $j=2$ | $j=3$ | $j=4$ | $\Gamma=1$ | $\Gamma=2$ | $\Gamma=3$ |
| **Feasible Mode Combinations (m)** | 2 | 2 | 2 | 2 | 5 | 68 | 8 | 2 | 10 | 1 | 78 | 86 | 88 |
| | 2 | 2 | 2 | 1 | 6 | 65 | 8 | 2 | 10 | 2 | 75 | 83 | 85 |
| | 2 | 2 | 1 | 2 | 6 | 62 | 8 | 2 | 2 | 1 | 70 | 72 | 74 |
| | 2 | 2 | 1 | 1 | 6 | 59 | 8 | 2 | 2 | 2 | 67 | 69* | 71* |
| | 1 | 2 | 2 | 2 | 5 | 48 | 15 | 2 | 10 | 1 | 63 | 73 | 75 |
| | 1 | 2 | 2 | 1 | 6 | 45 | 15 | 2 | 10 | 2 | 60 | 70 | 72 |
| | 1 | 1 | 2 | 2 | 6 | 44* | 15 | 3 | 10 | 1 | 59* | 69* | 72 |
| | 2 | 1 | 2 | 2 | 6 | 64 | 8 | 3 | 10 | 1 | 74 | 82 | 85 |

It could be observed from Table 10 that the optimal mode assignments to the deterministic problem and the robust problem with $\Gamma = 3$ are different. Therefore it can be concluded that, $\Gamma$, the parameter that controls the level of pessimism, is effective on the choice of activity modes.

#### 4.2.1.1. Benders Reformulation

In this section, we show how to solve Model 1, which is given in Equation 4.1, using a Benders Decomposition algorithm.

**Proposition 4.1:** Model 1 could be formulated as follows:

$$\text{Min} \sum_{j\in N}\sum_{m\in M_j} c_{jm}x_{jm} + z$$

subject to

$$\begin{aligned}
& z - \sum_{j\in N}\sum_{m\in M_j} d_{jm}u_j^{\,k}x_{jm} \ge 0 && k = 1,\dots,K \\
& \sum_{(i,j)\in A}\sum_{m\in M_j} p_{jm}w_{ij}^{s}x_{jm} \le \delta && s = 1,\dots,S \\
& \sum_{m\in M_j} x_{jm} = 1 && \forall j \in N \qquad (4.2) \\
& x_{jm} \in \{0,1\} && \forall m \in M_j, \forall j \in N, \\
& z \ge 0,
\end{aligned}$$

where, $u^k = (u_1^{\,k},\dots,u_n^{\,k})$, for $k = 1,\dots,K$ are the extreme points of the polytope

$U = \{u \in R^N : \sum_{j\in N} u_j \le \Gamma,\ 0 \le u \le 1\}.$

**Proof :**

We could reformulate (4.1) as:

$$f(\Gamma) = \text{Min}\left[\sum_{j\in N}\sum_{m\in M_j} c_{jm}\,x_{jm} + g(x) : x \in X^D\right], \text{ where} \qquad (4.3)$$

$$g(x) = \text{Max}\left\{\sum_{j\in N}\sum_{m\in M_j} d_{jm}x_{jm}u_j : \sum_{j\in N} u_j \le \Gamma, u \in B^n\right\} \tag{4.4}$$

Given a solution vector, $x \in X^0$, $g(x)$ is a knapsack problem of which LP relaxation has binary optimal solutions (see Theorem 1 of Bertsimas and Sim 2003), hence $g(x)$ could be rewritten as:

$$g(x) = \text{Max}\left\{\sum_{j\in N}\sum_{m\in M_j} d_{jm}x_{jm}u_j : \sum_{j\in N} u_j \le \Gamma,\ 0 \le u_j \le 1,\ \forall \text{j} \in \text{N}\right\} \tag{4.5}$$

*U* is a polytope and is therefore bounded and nonempty polyhedral set, thus we have,

$$g(x) = \underset{1\le \text{k} \le \text{K}}{\text{Max}}\left\{\sum_{j\in N}\sum_{m\in M_j} d_{jm}x_{jm}u_j^{\,k}\right\} \tag{4.6}$$

Hence using (4.6), (4.3) could be reformulated as:

$$\underset{x\in X^D, z}{\text{Min}}\left\{\sum_{j\in N}\sum_{m\in M_j} c_{jm}x_{jm} + z : z \ge \sum_{j\in N}\sum_{m\in M_j} d_{jm}u_j^{\,k}x_{jm},\ k = 1,..,K\right\} \tag{4.7}$$

Combining Lemma 3.1 and (4.7), (4.2) formulates Model 1. Note that we add the non-negativity constraint on *z* for the sake of algorithmic convenience.

Q.E.D.

Enumerating all the extreme points and paths is burdensome, so we use a relaxation approach and generate the constraints as needed. In the subsequent sections, we discuss the details of the proposed Benders Decomposition algorithm to solve the problem exactly in an efficient manner:

***Solution Algorithm:***

Introduce an additional index *t* to the notation to denote the values at iteration *t*.

```
1. Start with an initial solution, $\overline{x^1} \in X^0$, set $z^0 = -\infty$ , $t = 1$.

2. Solve $SP_1{}^t(\overline{x^t})$ : $\eta^t$ = Max $\{\sum_{i\in N}\sum_{j\in S(i)}\sum_{m\in M_j} p_{jm}\overline{x^t_{jm}}w_{ij} - \delta v : (w, v) \in W\}$ and

   Solve $SP_2{}^t(\overline{x^t})$ : $\psi^t$ = Max$\{\sum_{j\in N}\sum_{m\in M_j} d_{jm}\overline{x^t_{jm}}u_j : \sum_{j\in N} u_j \le \Gamma, 0 \le u_j \le 1 , \forall j \in N\}$

   let $u^t$ be the optimal solution.

3. If $SP_1{}^t(\overline{x^t})$ is unbounded (primal infeasible) then

       Get an extreme ray ($\overline{w^t}$ , $\overline{v^t}$) and set $\overline{u^t} = u^t$.
       $X^t = X^{t-1} \cap \{x \in X^0 : \sum_{i\in N}\sum_{j\in S(i)}\sum_{m\in M_j} p_{jm}\overline{w^t_{ij}}x_{jm} - \delta\overline{v^t} \le 0\}$
       If ( $\psi^t > z^{t-1}$ )
       $X^t = X^t \cap \{x \in X^0 : z \ge \sum_{j\in N}\sum_{m\in M_j} d_{jm}\overline{u^t_j}x_{jm}\}$
   Else
       If ( $\psi^t > z^{t-1}$ )
       $X^t = X^{t-1} \cap \{x \in X^0 : z \ge \sum_{j\in N}\sum_{m\in M_j} d_{jm}\overline{u^t_j}x_{jm}\}$
       Else
       Stop and report $\overline{x^t}$ as the optimal solution.
   End if

4. Solve the relaxed master problem, $MP^t$:

$\varphi^t$ = Min $\{\sum_{j\in N}\sum_{m\in M_j} c_{jm}x_{jm} + z : x \in X^t\}$.

Let $x^t$ be the optimal solution and $z^t$ be the optimal objective function.

5. $LB = \varphi^t$, $t = t + 1$, $\overline{x^t} = x^{t-1}$.

6. Return to Step 2.
```

In order to accelerate convergence and to solve large instances, we follow the branch-and-cut procedure described in 3.2.5. The basic difference is that at each iteration, in addition to the feasibility cuts, optimality cuts are inserted through solving an additional LP, $SP_2(\bar{x})$. It could be solved to optimality through a greedy algorithm. This algorithm orders $\sum_{m \in M_j} d_{jm} \overline{x_{jm}}$ values and this order identifies $\Gamma$ of the activities which maximally affect the objective function.

### 4.2.1.2. Experimentation and Computational Results

We employ the test bed used for deterministic problems to test the robust model. In addition to the parameters used to define the deterministic problems, two additional parameters, namely robustness level ($\Gamma$) and uncertainty factor ($\psi$), are required to solve the robust problem. The uncertainty factor represents the rate by which the variables $d_{jm}$ are allowed to change around $c_{jm}$, i.e. $d_{jm} = \psi c_{jm}$, where $\psi$ is uniformly distributed in [0.1, 1]. Table 11 summarizes the parameters of the test bed.

**Table 11. Experimental Setting for Solving the Robust Problem**

| **Parameter** | **Level(s)** |
|---|---|
| CI | 13 |
| CNC | 5, 6, 7, 8 |
| Number of modes | U[2,10] |
| Deadline parameter ( $\theta$) | 0.15 |
| Nominal cost function | ccv, cvx, hyb |
| Uncertainty factor ($\psi$) | U[0.1, 1.0] |
| Pessimism Level ($\Gamma$) | 0, $\lfloor 0.25n \rfloor, \lfloor 0.5n \rfloor, \lfloor 0.75n \rfloor$ |

Three instances are solved for each parameter setting. All the algorithms are implemented in C programming language on a Sun UltraSPARC 12x400 MHz workstation with 3 GB RAM. Optimization software CPLEX 9.1 is used to solve the linear and integer programs.

Computational experiments demonstrate that the parameter *Γ* reflects the risk attitude of the decision maker and highly affects the schedule construction. As the decision makers become more risk-averse, larger *Γ* values should be employed in the model so that schedules with lower worst-case costs can be generated. However, these schedules usually perform worse when nominal costs are attained. In practice the expected activity cost is used as the nominal cost. Figure 5 sketches the impact of the parameter *Γ* on schedule generation and shows the multi-criteria behavior of the robust problem.

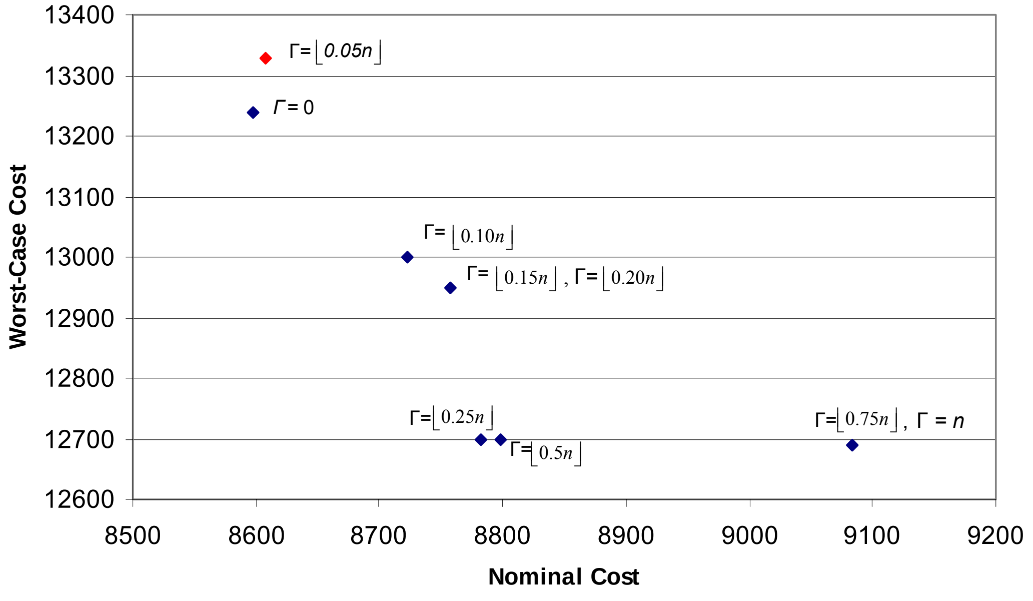

**Figure 5. The Impact of Pessimism Level on Schedule Generation**

Every point in the figure represents the optimal solution of Model 1 for a specific project network with $n = 85$ under a given *Γ*. Therefore every point is characterized by the risk attitude of the decision maker, and the corresponding schedule is generated by using Model 1. Note that the cost of each activity is represented with two parameters: the nominal cost and the worst-case cost. As it is illustrated in the figure, total nominal cost ( $\sum_{j \in N} \sum_{m \in M_j} c_{jm} \; x_{jm}$ ) and total worst-case cost ( $\sum_{j \in N} \sum_{m \in M_j} \overline{c_{jm}} x_{jm}$ ) of a schedule typically conflicts, i.e. a schedule with a lower nominal cost yields higher cost at the worst-case. However, there may be exceptional instances such as the solution generated with $\Gamma$ = $\lfloor 0.05n \rfloor$ in Figure 5, which is a dominated solution.

Besides the level of pessimism of the decision maker, we investigate the effects of various complexity parameters on the solution efficiency and summarize the results in Table 12. The results of the exact procedure are presented under the column labeled with "Optimum"; the average number of linear and integer master problems solved and the average CPU time in seconds to solve the instances are reported under the columns "LP Iter.", "IP Iter.", "CPU(s)", respectively.

**Table 12. Summary of Computational Results**

| | | | **Optimum** | | | **Truncated Solution** | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | | **LP Iter.** | **IP Iter.** | **CPU(s)** | **Ins Opt (%)** | **Avg Dev (%)** | **Max Dev (%)** | **Dec CPU (%)** |
| **Robust** | **CNC** | **5** | 16.32 | 19.23 | 1400.96 | 81.82 | 0.03 | 0.42 | 25.03 |
| | | **6** | 22.42 | 27.33 | 10427.91 | 100 | 0.00 | 0.00 | 15.04 |
| | | **7** | 23.00 | 31.14 | 18044.09 | 90.91 | 0.01 | 0.22 | 16.38 |
| | | **8** | 20.28 | 31.88 | 19139.61 | 84.00 | 0.03 | 0.37 | 16.20 |
| | **Γ** | $\lfloor 0.25n \rfloor$ | 19.42 | 26.84 | 8772.72 | 87.10 | 0.02 | 0.31 | 21.64 |
| | | $\lfloor 0.5n \rfloor$ | 20.40 | 27.73 | 10789.31 | 87.10 | 0.02 | 0.37 | 18.07 |
| | | $\lfloor 0.75n \rfloor$ | 21.77 | 27.58 | 11690.94 | 93.55 | 0.01 | 0.42 | 14.81 |
| **Deterministic** | **CNC** | **5** | 15.00 | 12.00 | 410.87 | 44.44 | 0.21 | 0.82 | 27.77 |
| | | **6** | 22.50 | 15.25 | 1935.93 | 100 | 0.00 | 0.00 | 21.42 |
| | | **7** | 22.00 | 17.17 | 7277.80 | 71.00 | 0.02 | 0.12 | 28.81 |
| | | **8** | 21.14 | 31.29 | 10974.72 | 75.00 | 0.01 | 0.04 | 30.44 |

The results of the approximate method are depicted under the column called "Truncated Solution", within which, the percentage of problem instances that the optimal solution has been found, the average percentage deviation from the optimal solution, the maximum percentage deviation from the optimal solution and the average CPU time reduction with truncation are reported under the columns "Ins Opt (%)", "Avg Dev (%) ", "Max Dev (%)", "Dec CPU (%)", respectively.

Table 12 reveals that the network complexity, which is measured with CNC, and the pessimism level are influential on the computational effort given in CPU seconds. The instances with complex network structure and higher pessimism level require more computational effort. Among these influential factors, CNC has already been shown to be influential in solving the deterministic problems in Chapter 3. When the CPU time for solving the deterministic problem and the robust problem are compared, we conclude that considerably higher computational effort is required when the notions of uncertainty and robustness are incorporated into the model. Finally, the truncation-based heuristic may be used as a solution alternative for large scale instances, as it is shown to be generating quick solutions that are very close to the optimal solution.

The main advantage of Model 1 is that it could be solved exactly by decomposing the problem into two subproblems. However, the drawback is that it assumes the activities with the same cost intervals are equally uncertain and all activities are likely to have cost values at the upper bounds. In real life, criticalities of the activities are crucial as well. Considering this shortcoming, we propose two novel models that integrate the criticality of multi-mode project networks and then compare the performances of all the three models in the next section.

## 4.2.2. Criticality-Based Uncertainty Models

### 4.2.2.1. Model 2

In Model 1, we assume that a subset of the activities has cost values at their upper bounds and the remaining coefficients are set to their nominal values. While determining this subset, only the activity costs are considered but the activity durations are ignored. However, cost and time are interdependent as they both depend on the amount of resource allocation. Activities having large slacks (i.e. non-critical activities) provide flexibility in resource allocation. It is possible to delay their starting times or to elongate the durations via lowering the amount of resource allocations. Due to these flexibilities, these activities involve less risk to achieve the cost targets when compared to the critical activities. On the other hand, in case of disruptions, managers usually allocate more resources to critical activities or in managerial terms crash these activities and this requires extra cost.

The conventional measure of an activity's criticality is the total slack, which is the amount of time by which the completion time of the activity can exceed its earliest completion time without delaying the project completion time. It is a measure of the insensitivity of schedule performance with respect to activity delays. The activities that have no slacks are defined to be critical activities. We will propose a criticality definition that is different from the conventional measure, as well.

Disregarding the activity slacks and assigning the worst-case costs to activities with ample slacks make Model 1 unrealistically pessimistic. To eliminate this over pessimism, the activities with cost values at the upper bounds are chosen from the critical ones in the criticality-based robust model. Given the mode assignments, only $\Gamma$ controls the pessimism level in model 1, however in the new model the critical activity set and *Γ* control the pessimism level. The following model represents the criticality-based approach.

$$f(\Gamma) = \mathrm{Max}\left\{ \sum_{j \in N} \sum_{m \in M_j} c_{jm} x_{jm} + \mathrm{Max}\left\{ \sum_{j \in N} \sum_{m \in M_j} d_{jm} x_{jm} u_j : \sum_{j \in CR} u_j \le \Gamma, u \in B^n \right\} : x \in X^D \right\} \quad (4.8)$$

*CR* refers to the set of critical activities in (4.8). In this new approach, criticality definition becomes crucial. For our problem, slacks are defined with respect to a specific mode assignment. As the mode assignments change, the slack distribution among the activities also changes.

In real-life projects, it would make much more sense to evaluate the activity slacks with respect to activity's duration since the higher the ratio of slack to activity time the higher its capability to compensate for a delay. The reason is that as the activity durations increase the probability of a larger number of disruptions to be observed while the activity is being performed, increases. Thus, we use the slack/duration ratio to assess criticality of activities and define the activities that have slack values less than *100ξ %* of the activity duration as *potentially critical activities*, i.e. $CR = \{ j \in N : TS_j / p_j \le \xi \}$ where $TS_i$ refers to the total slack of activity *i. 100ξ %* will be called slack duration threshold (SDT) from now on. In this study, we set the SDT to *25 %,* i.e. $\xi = 0.25$. The effect of this parameter on schedule construction will be discussed in the computational analysis section in Section 4.2.3.2. When compared with the classical definition, this new definition enlarges the criticality set.

In order to clarify the differences among the proposed models, the simple network in Figure 4 is used. Table 13 depicts the objective function values of the deterministic models ($\Gamma = 0$) and two robust models (with $\Gamma = 1, 2, 3$). The rows of the table show the feasible mode combinations for the activities. Optimal solutions are marked with a "*" and given a mode combination, the critical activities are underlined. Note that activity slacks depend on mode assignments, i.e. if a different feasible mode assignment is chosen, the slack amounts differs. It can be observed from the table that $\Gamma$ and $\Gamma_{CR,}$ which control the level of pessimism in the restricted uncertainty model and criticality-based model respectively, are effective on the choice of activity modes.

**Table 13. Comparison of Robust Models**

| | **Activity(*j*)** | | | | $C_5$ | $f(\Gamma=0)$ | $d_{jm} = \overline{c_{jm}} - c_{jm}$ | | | | **Robust Objective:** $f(\Gamma)$ | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | **1** | **2** | **3** | **4** | | | $j=1$ | $j=2$ | $j=3$ | $j=4$ | $\Gamma=1$ | $\Gamma_{CR}=1$ | $\Gamma=2$ | $\Gamma_{CR}=2$ |
| **Feasible Mode Combinations** | 2 | 2 | 2 | 2 | 5 | 68 | 8 | 2 | 10 | 1 | 78 | 68 | 86 | 68 |
| | 2 | 2 | 2 | 1 | 6 | 65 | 8 | $\underline{2}$ | 10 | $\underline{2}$ | 75 | 67 | 83 | 69 |
| | 2 | 2 | 1 | 2 | 6 | 62 | 8 | $\underline{2}$ | $\underline{2}$ | 1 | 70 | 64 | 72 | 66 |
| | 2 | 2 | 1 | 1 | 6 | 59 | 8 | $\underline{2}$ | $\underline{2}$ | $\underline{2}$ | 67 | 61 | 69* | 63 |
| | 1 | 2 | 2 | 2 | 5 | 48 | $\underline{15}$ | 2 | 10 | 1 | 63 | 63 | 73 | 63 |
| | 1 | 2 | 2 | 1 | 6 | 45 | $\underline{15}$ | $\underline{2}$ | 10 | $\underline{2}$ | 60 | 60 | 70 | 62* |
| | 1 | 1 | 2 | 2 | 6 | 44* | $\underline{15}$ | $\underline{3}$ | 10 | $\underline{1}$ | 59* | 59 * | 69* | 62 * |
| | 2 | 1 | 2 | 2 | 6 | 64 | 8 | $\underline{3}$ | 10 | $\underline{1}$ | 74 | 67 | 82 | 68 |

As Model 2 considers less risk premium in schedule costs, it is less pessimistic than Model 1. The following proposition proves this fact.

**Proposition 4.2:** Model 1 is more pessimistic than Model 2.

**Proof:**

Given a schedule $x \in X^D$, define the risk premiums considered in Model 1 and 2 as:

$$g_1(x) = \operatorname*{Max}_{u \in B^n} \left\{ \sum_{j \in N} \sum_{m \in M_j} d_{jm} x_{jm} u_j \; : \; \sum_{j \in N} u_j \le \Gamma \right\}, \tag{4.9}$$

$$g_2(x) = \operatorname*{Max}_{u \in B^n} \left\{ \sum_{j \in N} \sum_{m \in M_j} d_{jm} x_{jm} u_j \ : \ \sum_{j \in CR} u_j \le \Gamma \right\},$$

where $CR = \{j\text{: } TS_j/p_j \le \xi,\ j \in N\}$, (4.10)

The feasibility sets are: $U_1 = \left\{ u \in B^n : \sum_{j \in N} u_j \le \Gamma \right\}$, and $U_2 = \left\{ u \in B^n : \sum_{j \in CR} u_j \le \Gamma \right\}$.

From the definition of $CR$, any $j \in CR \subseteq N$; hence for any $u \in U_2$, then $u \in U_1$ as well, i.e. $U_2 \subseteq U_1$. Besides, both of the models defined in (4.9) and (4.10) have the same objective function, thus Model 1 is a relaxation of Model 2. As the problems are maximization problems,

$g_2(x) \le g_1(x) \qquad \forall x \in X^D$ (4.11)

Q.E.D.

**Corollary 4.1:** The optimal solution of the Model 2 provides a lower bound to Model 1.

**Proof:**

Using the notation introduced in Proposition 4.2, Models 1 and 2 could be reformulated as follows:

$z_1 = \text{Min } \{h_1(x) : x \in X^D\}$ (4.12)

$z_2 = \text{Min } \{h_2(x) : x \in X^D\}$ (4.13)

where $h_1(x) = \sum_{j \in N} \sum_{m \in M_j} c_{jm}\, x_{jm} + g_1(x)$ and $h_2(x) = \sum_{j \in N} \sum_{m \in M_j} c_{jm}\, x_{jm} + g_2(x)$.

Let $x^* \in X^D$ be the optimal solution of Model 1, then using Proposition 4.2,

$g_2(x^*) \le g_1(x^*)$ (4.14)

Combining (4.12), (4.13), and (4.14)

$$z_2 \leq h_2(x^*) = \sum_{j\in N}\sum_{m\in M_j} c_{jm}\, x^*_{jm} + g_2(x^*) \leq z_1 = h_1(x^*) = \sum_{j\in N}\sum_{m\in M_j} c_{jm}\, x^*_{jm} + g_1(x^*).$$

Q.E.D.

In order to be able to write the open form of the model in (4.8), an extra continuous decision variable set $E_j, \forall j \in N$ which denotes the earliest finishing time of activity $j$, and the following constraints are introduced to the deterministic model.

$$E_j - E_i - \sum_{m\in M_j} p_{jm}\, x_{jm} \geq 0 \qquad \forall (i,j) \in A \tag{4.15}$$

$$E_j \geq 0 \qquad \forall j \in N \tag{4.16}$$

The open form of Model 2 could now be written as follows:

$$\text{Min}\left\{ \sum_{j\in N}\sum_{m\in M_j} c_{jm} x_{jm} + M(E_j - C_j) + \omega(x) : x \in X^D, (4.15), (4.16) \right\} \tag{4.17}$$

$$\omega(x) = \text{Max}\left\{ \sum_{j\in N}\sum_{m\in M_j} d_{jm} x_{jm} u_j : \sum_{j\in N} u_j \leq \Gamma,\ (C_j - E_j)u_j \leq 0.25 \sum_{m\in M_j} p_{jm} x_{jm}, u \in B^n \right\} \tag{4.18}$$

Note that in this new formulation $C_j$ denotes the latest finishing time and $C_j - E_j$ represents the total slack of activity $j$, $\forall j \in N$. In (4.17), $M$ refers to a "big" number. The formulation includes nonlinear discrete components and has many binary integer variables and constraints. Given this complex structure, we use Tabu Search (TS) to solve the model and obtain good quality approximate solutions. Details of the algorithm and the parameter tuning will be discussed in Section 4.2.2.3. In the following subsection, we propose an alternative criticality-based approach, which lies in between the restricted uncertainty and criticality-based approaches regarding the level of conservatism. The same TS methodology is used to solve Models 2 and 3.

#### 4.2.2.2. Model 3

This model also accounts for the cost deviations in the non-critical activity set, but unlike the first criticality model, the critical activities have priority over non-critical ones. Given the mode combinations, $\Gamma$ activities which have cost values at their upper bounds are chosen from the critical activity set firstly. If the cardinality of the critical activity set is less than $\Gamma$, the remaining units are chosen from the non-critical activity set. While calculating the cost deviations both for the critical and non-critical activities, activities that have the cost coefficients influencing the objective most are given the priority. The following model illustrates the criticality-based alternative approach, namely Model 3:

$$f(\Gamma) = \text{Max}\left\{\sum_{j\in N}\sum_{m\in M_j} c_{jm}x_{jm} + \omega(x) : x \in X^D\right\} \tag{4.19}$$

$$\omega(x) = \text{Max}\left\{\sum_{j\in N}\sum_{m\in M_j} d_{jm}x_{jm}u_j : \sum_{j\in N} u_j \le \Gamma, \sum_{j\in CR} u_j \ge \text{Min}\left\{\Gamma, |CR|\right\}, u \in B^n\right\} \tag{4.20}$$

In the above formulation, $|CR|$ refers to the cardinality of the set of critical activities. This reformulation of Model 3 has a similar complex structure with the formulation of Model 2 (4.15 - 4.18) as it also includes nonlinear discrete components and many binary integer variables and constraints.

A feasible schedule of the network in Figure 4 and how criticality affects the possible courses of action are depicted in Table 14. Furthermore, the table summarizes the model characteristics to elucidate the differences among the models.

**Table 14. Summary of the Model Characteristics on an Example Schedule**

| **A Feasible Schedule from Figure 4.1** | | | | |
|---|---|---|---|---|
| | **Activity 1** | **Activity 2** | **Activity 3** | **Activity 4** |
| **Modes** | 2 | 2 | 2 | 1 |
| **Duration** | 2 | 3 | 1 | 3 |
| **Slack** | 3 | 0 | 2 | 0 |
| **Criticality** | Non-critical | Critical | Non-critical | Critical |

| **Possible Courses of Action in Cases of Disruptions** | | |
|---|---|---|
| **Activity Status** | Non-Critical | Critical |
| **Possible Courses of Action** | • No need for crashing.<br>• Ample slack creates resource allocation flexibility which increases the chance of meeting the expected cost. | • Need for crashing which requires extra cost.<br>• Deviation from the expected cost is highly likely in order to meet the schedule |

| **Differences Among Models** | |
|---|---|
| **Model 1** | Does not account for criticality. |
| **Model 2** | Accounts for slacks and only critical activities are assigned to upper bounds. |
| **Model 3** | Accounts for slacks, gives priority to critical activities but includes non-critical activities as well. |

The criticality requirement makes Model 2 and 3 more difficult to solve. Next, we provide the details of the TS heuristic to solve these criticality-based models.

### 4.2.2.3. Solution Methodology: Tabu Search and Parameter Settings

TS is a local-search improvement heuristic proven to be effective to solve many difficult combinatorial optimization problems (e.g., Hazir et al., 2008). It has a punishment mechanism to avoid getting trapped at local optima by forbidding or penalizing moves that cause cycling among solution points previously visited. These forbidden moves are called “tabu”. The short term memory keeps track of move attributes that have changed during the recent past and these attributes become tabu for a specific number of iterations. Under some conditions, tabu status of a move can be overridden. These conditions are called *aspiration criterion*. There are two commonly used strategies to obtain good solutions: diversification is used to direct the search into less visited regions of the search space, whereas intensification is used to fully explore a certain region.

Local search-based algorithms may not result in high quality solutions for the DTCTP-D, since it is not simple to identify a feasible solution in the neighborhood of the current solution; classical move operators do not guarantee feasibility. To overcome this shortcoming, we apply the features proposed by Kulturel-Konak et al. (2003), which are specially designed to solve constrained optimization problems. Their algorithm uses an adaptive penalty function which encourages the search to proceed through a portion of the infeasible region, namely the “*near feasibility threshold (NFT)*”. The generated solutions are penalized according to their distances to the feasibility region. Their method requires few parameters and is shown to be insensitive to parameter changes.

In our TS algorithm, each solution is represented with a mode assignment vector. Infeasible mode assignments are allowed with some penalties. The algorithm starts the exploration in the infeasible region with the least cost solution, the solution in which the activities are assigned to the longest modes. By using a cost-based fitness function that is composed of total project cost and an adaptive penalty cost, it keeps searching for feasible and efficient directions until the stopping criterion is satisfied. As a penalty function, we use the adaptive one proposed in Kulturel-Konak et al. (2003).

In order to fully explore the neighborhood of the generated solutions, we examine the entire neighborhood with both mode decreasing and increasing moves. The neighborhood of a solution is comprised of solutions with the all possible mode assignments that could be obtained by a single mode decreasing or increasing move. Hence the mode of a single activity alters.

To find the best parameters, some test problems of varying problem sizes are solved for a wide range of system parameters. In the test runs, we test a tabu list of size 5, 7, and 10. The best solutions are achieved with a tabu list of size 7. This short term memory structure prevents cycling in between some solutions. However, we define the aspiration criterion so that tabu status of a move can be overridden if it leads to a solution better than the incumbent solution.

The stopping criterion is set to 10,000 iterations after observing that it is sufficient to obtain convergence for the test instances. In order to direct the search into less-visited regions of the search space and escape from local optima, we use a simple diversification strategy. If the incumbent solution is not updated for 1,000 iterations, the algorithm restarts with a randomly generated neighbor of the initial solution; the tabu list is initialized and the move values are recalculated according to the new solution.

The only difference in TS algorithms of Model 2 and Model 3 is the cost component of the fitness function. Solution representation, memory structure, penalty function and parameter setting are the same. In the following subsection, we introduce some robustness measures to assess the robustness of the generated schedules and compare the effectiveness of the proposed robust project scheduling models by using these measures.

### 4.2.3. Comparison of the Proposed Models

In this section, we first give a brief review of the metrics to evaluate schedule robustness. Afterwards, we perform computational experiments and evaluate the generated schedules by using several robustness measures.

#### 4.2.3.1. Robustness Measures

Existing robust scheduling studies generally address machine environments and often follow scenario-based approaches, where scenarios for job attributes are required to be defined. They basically employ two types of robustness measures: direct measures, which are derived from realized performances, and heuristic approaches, which utilize simple surrogate measures. Computational burden of optimizing direct measures is generally higher when compared to surrogate measures. We refer the readers to Sabuncuoğlu and Gören (2005) for a detailed discussion of the measures.

Since achieving project completion time and project cost targets are crucial for project managers, we evaluate the robustness of project schedules both in terms of cost and time. The evaluation is based on the following measures in this dissertation.

## I. Cost-Based Measures

### a) Expected Realized Cost

The cost of performing each activity is represented with two parameters: the nominal cost and the worst-case cost. We assume that the nominal cost of a mode is equal to the expectation over all the scenarios corresponding to possible alternatives in practice. Therefore, for a given schedule the summation of the nominal costs over all activities defines the expected realized cost of the project schedule. The schedule which has the minimal expected realized cost is chosen as the most robust schedule.

### b) Worst-Case Cost

The upper bounds of the cost intervals define the worst-case costs, the maximal cost among all possible scenarios. Hence the summation of the upper bounds of the cost intervals over all activities characterizes the worst-case realized cost of a schedule. The schedule that has the smallest worst-case cost (among all schedules) is chosen. This is a risk-averse approach and corresponds to the minimax objective in decision analysis.

### c) Cost of the Reference Scenario

This measure concentrates on a specific scenario in which critical activities are realized at the worst-case costs with the remaining activities being realized at the nominal costs. The schedule that has the smallest cost with respect to this scenario is selected.

## II. Time-Based Measures

We introduce some time-based robustness measures and discuss them briefly in this chapter. A more detailed discussion and a comprehensive literature review of these measures will be given in Chapter 5.

### a) Average Total Slack

Average of the slacks, or equivalently the sum of the slacks over all the activities could be used to assess schedule robustness. The schedules with larger average total slacks are preferred in terms of robustness.

### b) Dispersion of Slack/Duration Ratios

Uniform distribution over the schedule is expected to be beneficial as it distributes the delay risk among activities evenly. As a dispersion measure, we use the coefficient of variation (CV), which is the standard deviation divided by its mean, of activity slacks. Since larger means and smaller variances of slack/duration ratios are preferred in terms of robustness, the schedules with larger CV's will be more robust.

### c) Percentage of Critical Activities

By definition, delays in critical activities may result in delays in the project completion time. Therefore, schedules with a smaller number of critical activities are preferred in terms of robustness. We employ the ratio of the number of critical activities to the total number of activities in the project as a robustness measure.

**d) Project Buffer Size**

Buffers are protection mechanisms against uncertainty in the duration of activities. Project buffers are inserted to the end of projects to avoid possible delays in project completion. We use the ratio of project buffer size to project deadline as a robustness measure. The larger the project buffer, the more protected the project against disruptions.

### 4.2.3.2. Computational Analysis

Computational experiments are carried out to evaluate the performance of the models under different problem settings and the models are compared using the above-mentioned robustness measures. We refer the readers to 4.2.3 for details of the experimental setting. For comparison purposes we set pessimism level to $\Gamma = \lfloor 0.25n \rfloor$. In the computational study, for each setting 3 different instances are solved, hence each model is tested with 36 problems. Table 15 compares the introduced models with 7 robustness measures. While comparing, Model 1 is taken as the reference model and percent differences between robustness measure of the reference model and the criticality-based models are reported. For each robustness measure the average percentage deviation from the reference model and the percentage of problem instances that the model dominates the reference model are reported in the rows % Dev, % Dom, respectively. We also report the paired t-test confidence intervals (CI) for percent differences between robustness measures of restricted uncertainty model and criticality-based models. 95 % confidence level is used in these intervals. Furthermore, statistical significance of the differences is reported. The last row expresses whether minimization or maximization of the measure is preferred in terms of robustness.

**Table 15. Model Comparison with Robustness Measures**

| | | Cost Based Measures | | | Time Based Measures | | | |
|---|---|---|---|---|---|---|---|---|
| | | Expected Cost | Worst Case Cost | Reference Scenario | Average Slack | Slack Dispersion | % Critical | Project Buffer |
| **Model 2** | **% Dev** | 3.60* | 14.56* | -17.33* | 10.37* | -10.34* | -72.58* | 3.17* |
| | **CI** | (2.51, 4.69) | ( 13.29, 15.84) | (-18.86, -15.81) | (8.45, 12.28) | (-14.27, -6.41) | (-77.98, -67.18) | (2.82, 3.52) |
| | **% Dom** | 13.89 | 0 | 100 | 100 | 86.11 | 100 | 100 |
| **Model 3** | **% Dev** | 0.67 | 9.21* | -9.24* | 2.97* | -55.94* | -17.67* | 0.57* |
| | **CI** | (-0.27, 1.61) | (7.96, 10.43) | ( -11.41, -7.07) | (1.46, 4.48) | (-58.28, -53.60) | (-22.22, -13.12) | (0.39, 0.76) |
| | **% Dom** | 38.89 | 2.78 | 94.44 | 80.56 | 100 | 88.89 | 91.67 |
| **Optimization Criterion** | | Min | Min | Min | Max | Min | Min | Max |

Note: Statistical significance of % Dev is tested.
* indicates that test statistic for is significant at 5% level

When worst-case robustness measure is considered, Model 1 dominates the criticality-based models. This finding is consistent with the argument that Model 1 is over-pessimistic. Model 1 is better in most of the problem instances when the expected realized cost of the models is compared. However, in this case differences among model performances are small. As could be expected, Model 1 performs poorly under the specific scenario in which critical activities are realized at the worst-case costs whereas the remaining activities at the nominal costs. Model 2 outperforms the other models under this scenario.

When time based measures are considered, Model 1 is dominated by Models 2 and 3. Model 2 minimizes the number of critical activities and it is more protected against disruptions since it has larger project buffers. Furthermore, it produces the largest average slack. Model 3 distributes the slacks more evenly among the activities, hence results in lower coefficient of variations. It could also be observed that the performance of Model 3 lies between the performances of the Model 1 and 2 for majority of the robustness measures.

The parameter $\Gamma$ reflects the risk attitude of the decision makers. As the decision makers become more risk-averse, larger $\Gamma$ values should be used so that schedules with lower worst-case costs could be generated. Note that Model 2 is not so sensitive to the changes in parameter $\Gamma$ when compared to other models. This is basically due to the criticality requirement. The second parameter that affects model characteristics is the slack/duration threshold (SDT). Lower SDT results in different reactions in criticality-based models. Since the risk premiums are incurred for only critical activities in Model 2, as the threshold decreases, the number of critical activities is reduced so that total risk premium decreases and the model converges to the deterministic DTCTP with nominal costs. However, in Model 3, a reduction in the threshold might increase the total risk premiums, as risks of non-critical activities are also incorporated.

Besides the model characteristics, we investigate the solution efficiency and report the results in Table 16. This table illustrates the average CPU time to solve the problem instances for each model. The problems are classified according to the coefficient of network complexity (CNC), which is the ratio of the number of arcs to the number of nodes. Each iteration of the criticality-based models requires higher computational efforts due to model complexity and we run TS sufficiently long (10,000 iterations) to achieve convergence to high quality solutions.

**Table 16. Models and Computational Requirements**

| | | **CPU (sec)** | | |
|---|---|---|---|---|
| | | **Model 1** | **Model 2** | **Model 3** |
| **CNC** | **5** | 1400.96 | 3567.47 | 3309.628 |
| | **6** | 10427.91 | 6311.57 | 6094.823 |
| | **7** | 18044.09 | 17332.73 | 16075.57 |
| | **8** | 19139.61 | 38854.95 | 37804.99 |

## 4.3. Conclusions

In response to the crucial need to build robust project schedules that are less vulnerable to disruptions caused by uncontrollable factors, in this chapter we have proposed three alternative models to formulate the robust DTCTP. In these models the uncertainty in the cost of activities is addressed. In order to solve the models, we have developed both exact and heuristic algorithms. The first model is solved exactly by using Benders Decomposition; the other two criticality-based models are rather complex and solved approximately by a tabu search algorithm.

To evaluate the performance of the algorithms under various problem settings, we have conducted computational experiments. We have assessed the robustness of the schedules generated by the algorithms by using several cost and time-based robustness measures. When worst-case cost of the generated schedules is considered, Model 1 is advantageous; but it is usually dominated by the criticality-based models regarding time-based robustness measures.

The major contribution of this chapter is the development of robust optimization models, which incorporates uncertainty into DTCTP, and the development of tailored solution approaches. To the best of our knowledge, the models in this section are the first implementation of robust optimization to the DTCTP. In that sense, results presented in this chapter serve as a useful base to fill the research gap in developing robust project schedules for multi-mode project networks.

# CHAPTER 5

# ROBUSTNESS MEASURES AND A SCHEDULING ALGORITHM FOR THE DISCRETE TIME/COST TRADE-OFF PROBLEM

## 5.1. Introduction

In this chapter, we focus on generating robust schedules for the DTCTP that exhibit the ability to protect performance against unexpected events. More precisely, we address the case where the processing times of some activities might be subject to uncontrollable disruptions, and the problem is to select a mode for each activity so that the project is likely to be completed within a preset deadline and the total cost is minimized. We propose some models and algorithms to generate project schedules that are not affected largely by disruptions. These scheduling algorithms will assist project managers to reach time and cost-based project objectives.

Firstly, we introduce formal surrogate measures that aim at providing an accurate estimate of the schedule robustness. The effectiveness of each proposed measure is thoroughly assessed through computational experiments. Using the insight revealed by the computational study, we propose a two-stage robust scheduling algorithm that requires successively solving two variants of the deterministic discrete time/cost trade-off problem. Finally, we address the relationship between budget amplification and robustness, and introduce an analytical model to support the decision makers in budget allocation decisions. We provide evidence that the proposed approach can be extended to solve a complex robust discrete time/cost trade-off problem with tardiness penalties and earliness revenues.

## 5.2. Robustness Measures (RM)

Developing formal quantitative metrics that provide a good estimate of schedule robustness is crucial to develop efficient scheduling algorithms. The baseline schedules that are created by utilizing these robustness measures are capable of absorbing unanticipated disruptions. Furthermore, these metrics should be calculated easily for a given baseline schedule. Optimizing direct measures in project networks is generally more difficult when compared to optimizing surrogate measures. The reason is that project networks consist of multiple paths that usually intersect and it is difficult in these networks to determine the effects of disruptions analytically. A reasonable approach to increase the computational efficiency is to use good surrogate measures and determine a scheduling algorithm optimizing these surrogate measure.

In project scheduling literature, there are only a few studies that propose measures to assess the robustness of project schedules. They address randomness in duration of the activities and suggest the use of surrogate measures due to the complexity of analytical determination of realized performances. Al-Fawzan and Haouari (2005) use the sum of free slacks as a surrogate metric for measuring the robustness of a schedule. Kobylański and Kuchta (2008) discuss a limitation of this measure and propose using the minimum of all free slacks or the minimum of free slack/duration ratios. However, focusing on the minimum values has the weakness that two schedules with the same minimum values could have different distributions and the measures proposed by Kobylański and Kuchta (2008) fail to differentiate between these schedules. On the other hand, Lambrechts et al. (2008) introduce a free slack utility function. Lately, Chtourou and Haouari (2008) propose twelve predictive indicators for resource constrained networks. Clearly there is a need for further work to develop new measures and test the quality and efficiency of these measures.

Now, we introduce nine slack-based measures that could be used to evaluate schedule robustness. Subsequently they will be analyzed and compared through simulation. As we concentrate on the quality robustness, we use total slack (TS) instead of free slack (FS), and propose and test the following surrogate measures:

**a) Average Slack**

The average slack, or equivalently, the sum of the slacks is commonly used to assess schedule robustness in scheduling literature (Leon et al. (1994), Al-Fawzan and Haouari (2005), Gören and Sabuncuoğlu (2008)). Experimental studies of Leon et al. (1994) on job shop scheduling reveal that there is a high correlation between robustness and the average slack value. However, it has not been experimentally tested for project scheduling problems. The average slack measure for project networks could be formulated as follows:

$$RM_1 = \frac{\sum_{i=1}^{n} TS_i}{n} \tag{5.1}$$

In this formulation $TS_i$ refers to the total slack of activity $i$.

**b) Weighted Slack**

Minimizing the average of slacks assumes that the contribution of slacks to robustness is the same for each activity. However, some of the activities are more likely to delay the project completion; hence more slacks should be allocated to these activities. One approach to accomplish this is to give larger weights to these activities and minimize the total weighted slack. Chtourou and Haouari (2008) propose to use the number of immediate successors as the weights since the activities having larger number of successors are more likely to affect the project makespan. They address resource constrained networks and use free slacks; however free slacks resulted in poor correlations in our pretests, we adapt the measure using total slacks as follows:

$$RM_2 = \sum_{i=1}^{n} NIS_i \times TS_i \tag{5.2}$$

In this formulation, $NIS_i$ refers to the number of immediate successors. Instead of concentrating on immediate successors, we suggest to use the number of all successors, $NS$, both immediate and indirect. As a result we formulate the robustness measure as follows:

$$RM_3 = \sum_{i=1}^{n} NS_i \times TS_i \tag{5.3}$$

**c) Slack Utility Function**

The average or weighted slack approach assumes the same return for every unit of slack assigned. This approach might unnecessarily inflate slacks for some of the activities. One alternative approach to eliminate this problem is to use a function that has diminishing returns per extra unit of slack.

For the resource constrained networks, Lambrechts et al. (2008) use a free slack based weighted utility function. Due to poor correlations in the pretests with free slacks, we use TS instead and assuming equal weights for each activity, we propose the following adapted measure:

$$RM_4 = \sum_{i=1}^{n} NS_i \sum_{j=1}^{TS_i} e^{-j} \tag{5.4}$$

In practice, it would make more sense to evaluate the activity slacks with respect to the activity's duration, since the higher the slack/duration ratio (SDR), the higher its capacity for preventing delay. The reason for this is that as the activity durations increase, the probability of observing longer delays while the activity is being performed increases. Thus, as an alternative approach, we propose to use SDR's to assess robustness as follows:

$$RM_5 = \sum_{i=1}^{n} NS_i \sum_{j=1}^{\lceil SDR_i \rceil} e^{-j} \text{ , where } SDR_i = \frac{TS_i}{p_i} \tag{5.5}$$

Chtourou and Haouari (2008) define a threshold level and assume zero return when the slacks allocated is more than the threshold. We adapt this measure as follows:

$$RM_6 = \sum_{i=1}^{n} Min\{TS_i, frac.p_i\} \tag{5.6}$$

The parameter *frac* refers to the expected percentage increase in activity duration. We set this parameter to 20% in our experiments.

**d) Dispersion of Slacks**

In addition to the magnitudes of activity slacks, their dispersion over the activities might be consequential to evaluate schedule robustness. Low variability of activity slacks is expected to be beneficial as it distributes the risk of delay among activities evenly. As a dispersion measure, we propose using the coefficient of variation (CV), which is the standard deviation divided by its mean, and formulate the robustness measure as:

$$RM_7 = \frac{\sqrt{Var\ (SDR_i)}}{\overline{SDR_i}}\text{, where } SDR_i = \frac{TS_i}{p_i} \qquad (5.7)$$

Since high slack quantities and small variability among activities are preferred regarding the robustness, the schedules with smaller CV are deemed to be more robust.

**e) Percentage of Potentially Critical Activities**

The conventional measure of an activity's criticality is its *TS* value, which in fact assesses the insensitivity of schedule performance with respect to the activity delay. Traditionally in the literature, the activities that have zero slacks are defined to be critical activities. As first mentioned in Section 4.2.2.1, we further enlarge the set of critical activities and use the SDR as a criterion to identify the criticality of activities and define the activities that have slack values less than $100\xi$ *%* of the activity duration as *potentially critical* activities, i.e. $CR = \{j: TS_j/p_j \le \xi\}$. In Section 4.2.3, we discussed the impact of the slack duration threshold on schedule construction and decided to set $\xi = 0.25$.

Since delays in potentially critical activities probably result in delays in the project completion time, schedules with less number of critical activities are preferred regarding robustness. We employ the ratio of the number of potentially critical activities to the total number of activities as a quality robustness measure and formulate as:

$$RM_8 = |CR| = \frac{\sum_{i \in N} I_i}{n}, \text{ where } \begin{cases} I_i = 1 \text{ if } TS_i \leq 0.25 p_i \\ I_i = 0 \text{ o/w} \end{cases} \tag{5.8}$$

**f) Project Buffer Size**

Project buffers as protectors of the project schedule against uncertainty in the duration of activities, are added to the end of projects to prevent possible delays. The critical chain project management (CCPM) emphasizes the importance of buffer management and proposes to insert project buffers. In CCPM, safety factors are eliminated from individual activities and aggregated at the end as a project buffer. Since these buffers aim to prevent the project from exceeding the due date, CCPM is a quality robust scheduling method. We refer the reader to Herroelen and Leus (2001), for an experimental evaluation of CCPM, and to Tukel et al. (2006) for analysis of several buffer sizing methods.

Schedules with larger project buffers are preferred regarding robustness. However, it may deteriorate other performance measures such as project cost. This trade-off makes analytical determination of buffer sizes crucial. In spite of the importance, there exist only a few analytical methods to determine the size of buffers in the literature (see section 2.2.2). In this research, we propose using the project buffer size as percentage of project deadline. This quality robustness measure is formulated below:

$$RM_9 = 100 \frac{\delta - C_{n+1}}{\delta} \tag{5.9}$$

In the next section, we perform an experimental study to assess the quality of proposed robustness measures and present the results of a computational study.

## 5.3. Experimental Analysis of the Robustness Measures

### 5.3.1. Experimental Methodology

We use Monte-Carlo Simulation to generate a random set of realizations of activity durations to test robustness measures. Project Management Institute (2004) defines *Monte-Carlo Simulation* in the context of project management as "a technique that computes or iterates the project cost or schedule many times using input values selected at random from probability distributions of possible costs or durations, to calculate a distribution of possible total project cost or completion dates". Simulation is frequently used both in machine and project scheduling to model the stochastic and dynamic nature of scheduling systems and to test the performance of the proposed algorithms.

Given a baseline schedule for a benchmark project and realizations of activity durations, we evaluate the effect of disruptions on project performance by the use of some performance measures. These measures are functions of the difference between the given project deadline and the expected realized completion time. Having simulated the projects, the robustness measure that has the highest correlation with the performance measures (PM) is selected as the best metric to represent robustness. The following performance measures are used in our simulation to evaluate the quality robustness of the schedules:

1. **$PM_1$:** The proportion of replicates in which the project ends before the deadline.
2. **$PM_2$:** Average delay in the project completion time as percentage of the project deadline (i.e. $100\frac{C_{n+1}-\delta}{\delta}$ for delayed projects).

As a test set for assessing the effectiveness of the robustness measures, we use the DTCTP instances generated by Akkan et al. (2005). For each problem instance the schedules are generated using the given scheduling policy. Then robustness measures of the generated schedules are calculated and afterwards performance measures are determined through simulation. The following algorithm is used to test the robustness measures using simulation.

1. **Schedule Generation**: Given the scheduling policy, generate an initial baseline schedule. Then calculate the $RM_i$, ( $i = 1,...,9$ ) of each schedule.

2. **Monte-Carlo Simulation**:
   - **a.** Set the processing time of each activity to a random number generated by using the activity time distribution (in other words, the activity durations are perturbed while executing the schedule in the simulation run).
   - **b.** Generate the early start schedule (ESS) by using the randomly generated durations and classical critical path method operations. Find out and record the project completion time.
   - **c.** Repeat step 2, *Nr* (Number of Replications) times.

3. **Correlation Computation**: Calculate the $PM_j$, ($j = 1, 2$ ). Compute and report the correlation between $RM_i$ and $PM_j$ ($i = 1,...,9;\ j = 1, 2$).

The following three scheduling policies are used in the algorithm above:

1) **Optimal DTCTP-D**: An initial baseline schedule is generated assuming no disruptions and solving the DTCTP-D exactly. Optimal mode assignments to this problem set are determined by using Benders Decomposition (see Section 3.2.3 for the details). After assigning the modes, corresponding ESS is determined by using classical CPM calculations.

2) **Project Buffer Insertion:** Buffer insertion policy assumes a smaller deadline, i.e. $\delta' = \delta (1- \tau)$, $0 < \tau < 1$, and inserts a project buffer at the end of the schedule. The project buffer size is proportional to the project due date. Modes are assigned to activities by solving the DTCTP-D exactly with $\delta'$ and then corresponding ESS is determined.

3) **Safety Time Insertion:** Safety time insertion introduces a safety time for each activity that is proportional to the nominal durations, i.e. $p_i' = p_i (1 + \Delta), \forall i \in N$. Modes are assigned to the activities by solving the DTCTP-D exactly, assuming the augmented durations. However, the so-called roadrunner mentality is used to generate the schedule, i.e. the non-gating tasks (activities with non-dummy predecessors) are started as soon as possible and safety times are ignored in this schedule.

For each problem instance, we generate 21 alternative schedules using the policies defined above (optimal DTCTP; $\tau \in \{0.01, 0.02, ... , 0.10\}$; $\Delta \in \{ 0.01, 0.02,..., 0.10\}$). To model the activity durations, we use a lognormal distribution with mean equal to the baseline duration and $CV = 0.5$. Many other project scheduling studies also suggest the use of this distribution (see Tavares et al. (1998), Herroelen and Leus (2001) and Tukel et al. (2006)). Tavares et al. (1998) list the following reasons to choose the lognormal distribution:

> (a) There is a lower bound which corresponds to the minimal feasible duration. This limit is due to contractual or technical reasons.
> (b) The lognormal distribution is an asymmetric distribution with the mode on the left side of the expected value, which is a very common feature of the duration of the activities.
> (c) The upper quantiles are unbounded. Actually, the occurrence of uncertain and inconvenient factors can always delay even further, the duration of the activity.

We generate a normal pseudo-random number, *X*, given a source of uniform pseudo-random numbers by using the polar form of the Box-Muller (1958) transformation. Then, the exponential function $Y = e^X$ returns a lognormal random variable. Since projects are temporary events, we employ terminating simulations; the processing of dummy finish activity terminates the simulation. To determine the number of replications required, we use the sequential procedure proposed by Law and Kelton (2000, see Chap. 9.4). This procedure inserts new replications one by one and determines the length of the simulation so that 95 % confidence level for the mean of the performance measures is constructed. In this procedure, a relative error of 5 %, which specifies a bound on the percentage error of the point estimate of the sample means, is used.

### 5.3.2. Computational Results

We use a subset of the random instances generated by Akkan et al. (2005) to test the proposed measures and algorithms. Table 17 summarizes the parameters and their levels in the test bed.

**Table 17. Experimental Setting of the Simulation Analysis**

| Parameter | Level(s) | Parameter | Level(s) |
|---|---|---|---|
| **CI** | 13 | **Deadline parameter** | 0.15 |
| **CNC** | 5, 6, 7, 8 | **Nominal cost** | ccv, cvx, hyb |
| **Number of modes** | U[2,10] | | |

In the computational study, for each setting three different instances are solved, hence each measure is tested with thirty six problems. To evaluate the relationship between robustness and performance measures, we run regression models and report the coefficient of determination ($R^2$) and the significance levels. Table 18 illustrates the average of $R^2$ over all problem instances that have the same network complexity figures. In this regression, each $RM_i$, ($i = 1, \dots , 9$) is modeled as the independent variable and $PM_j$ ($j = 1, 2$) as the dependent variable. Before running the regression model, the necessary assumptions of normality, equal variance and independence are checked. To ensure the sampling independence, different seeds are used for different scheduling policies and Box-Cox transformation is utilized to assure the constant variance assumption.

Table 18 demonstrates that when $R^2$ are compared, the buffer size ($RM_9$) is the best robustness measure regardless of the network complexity. Furthermore, $RM_1$, $RM_2$, $RM_3$, $RM_5$, $RM_8$ also have high correlations with the PM. For all the measures, the $R^2$ figures are found to be insensitive to the changes in CNC. Proposed transformation of the slack utility function has significantly improved the correlations ($RM_5$ vs. $RM_4$). Furthermore, weighting with the number of the entire successors instead of the immediate successors improves the $R^2$ figures ($RM_3$ vs. $RM_2$).

**Table 18. Results of Regression of Robustness Measures on Performance Measures**

| | $R^2$ CNC = 5 | | $R^2$ CNC = 6 | | $R^2$ CNC = 7 | | $R^2$ CNC = 8 | | $R^2$ Average | |
|---|---|---|---|---|---|---|---|---|---|---|
| | **$PM_1$** | **$PM_2$** | **$PM_1$** | **$PM_2$** | **$PM_1$** | **$PM_2$** | **$PM_1$** | **$PM_2$** | **$PM_1$** | **$PM_2$** |
| **$RM_1$** | 0.9104 | 0.8717 | 0.9486 | 0.9301 | 0.9611 | 0.9446 | 0.9812 | 0.9579 | 0.9503 | 0.9261 |
| **$RM_2$** | 0.9468 | 0.9191 | 0.9681 | 0.9503 | 0.9542 | 0.9432 | 0.9778 | 0.9634 | 0.9617 | 0.9440 |
| **$RM_3$** | 0.9538 | 0.9251 | 0.9701 | 0.9552 | 0.9582 | 0.9460 | 0.9754 | 0.9627 | 0.9644 | 0.9479 |
| **$RM_4$** | 0.2239 | 0.2541 | 0.1928 | 0.2194 | 0.2318 | 0.2454 | 0.2284 | 0.2537 | 0.2192 | 0.2429 |
| **$RM_5$** | 0.8529 | 0.8376 | 0.9349 | 0.9239 | 0.9131 | 0.8972 | 0.9402 | 0.9299 | 0.9119 | 0.8989 |
| **$RM_6$** | 0.6411 | 0.6626 | 0.5867 | 0.6270 | 0.5347 | 0.5378 | 0.5814 | 0.5364 | 0.5860 | 0.5889 |
| **$RM_7$** | 0.3789 | 0.3803 | 0.4016 | 0.3952 | 0.6597 | 0.6610 | 0.6765 | 0.6711 | 0.5292 | 0.5269 |
| **$RM_8$** | 0.8524 | 0.8381 | 0.8726 | 0.8608 | 0.8253 | 0.8164 | 0.7500 | 0.7470 | 0.8251 | 0.8156 |
| **$RM_9$** | 0.9603 | 0.9462 | 0.9703 | 0.9612 | 0.9707 | 0.9588 | 0.9802 | 0.9653 | 0.9704 | 0.9579 |

We compare the best four RM among each other, test the significance of the differences, report the t-test with 95 % confidence interval, and the corresponding *p* values in Table 19. This table illustrates that $RM_9$ and $RM_3$ are good estimates of schedule robustness. They have significant differences when compared to the other robustness measures. Based on the results of the computational study, we propose a two-phase approach for generating robust schedules in Section 5.4.

**Table 19. Individual 95% Confidence Intervals for All Pairwise Comparisons**

| | **$PM_1$** | | |
|---|---|---|---|
| | **$RM_3$** | **$RM_2$** | **$RM_1$** |
| **$RM_9$** | (-0.318, 1.518)<br>0.193 | (-0.212, 1.946)<br>0.112 | (0.385, 3.626) *<br>0.017 |
| **$RM_3$** | | (-0.063, 0.596)<br>0.109 | (0.106, 2.705)*<br>0.035 |
| **$RM_2$** | | | (0.015, 2.263)*<br>0.047 |
| | **$PM_2$** | | |
| | **$RM_3$** | **$RM_2$** | **$RM_1$** |
| **$RM_9$** | (-0.074, 2.052)<br>0.067 | (0.148, 2.624)*<br>0.029 | (1.455, 4.912)*<br>0.001 |
| **$RM_3$** | | (0.059, 0.735)*<br>0.023 | (0.820, 3.569)*<br>0.003 |
| **$RM_2$** | | | (0.650, 2.945)*<br>0.003 |

* *Statistically significant difference*

## 5.4. A Methodology to Generate Robust Schedules

Using the insight revealed by the simulation results, we generate the baseline schedule by maximizing the project buffer size ($RM_9$), the robustness measure that has the highest correlation with PM, so that the schedule involves sufficient safety time to absorb unanticipated disruptions. However, while maximizing robustness, the project cost should remain within acceptable limits. We propose to use the following two-phase methodology to generate a robust schedule:

**Phase 1: Exact solution of the DTCTP-D**: Given a project deadline, DTCTP-D is formulated and solved exactly. The objective value of the optimal solution sets a threshold budget value, $B_0$, for the subsequent phase. This is the minimum achievable cost under the assumption that each activity lasts as it is planned and the deadline constraint is satisfied. However, this generated schedule is not protected against disruptions. In the sequel, we refer to this schedule as the *non-protected schedule* and we will use it as a benchmark.

**Phase 2: Exact solution of the DTCTP-B**: Having set the budget to $B_0$, an initial baseline schedule is generated by solving the DTCTP-B exactly to assign modes to activities and generating the corresponding early start schedule (ESS) given the mode assignments. In doing so, the algorithm aims at inserting a maximal-size project buffer while controlling the project cost. Furthermore, given the mode assignments, ESS maximizes the sum of total slacks, equivalently $RM_1$.

In our experiments, we notice that small and medium-sized DTCTP-D and DTCTP-B instances could be efficiently solved using CPLEX 9.1. The exact solutions are based on the formulations given in sections 3.2.1 and 3.3.1. However, for solving large-scale instances exactly, we use Benders Decomposition (see Section 3.2.3 and 3.3.2).

As expected, we observe that increases in the budget result in increases in the project buffer obtained after solving an appropriate DTCTP-B. Consequently, in order to improve the schedule robustness, we set in Phase 2 an "inflated" budget $B = (1+ \eta)\, B_0$ (with $\eta > 0$). However, as this policy inflates the budget, it is crucial to address the trade-off between project cost and schedule robustness. For this purpose, an analytical model to set $\eta$ in a most profitable way will be introduced in Section 5.5.

Interestingly, a slight increase in the project buffer usually results in a significant improvement in the performance measures. We test the significance of the difference between the performance measures of the protected schedules with $\eta = 0.02$ and the non-protected schedules ($\eta = 0$) as well as the corresponding confidence intervals. The results are summarized in Table 20. We see from this table that when the proposed two-phase approach is used, it is possible to increase the probability of completing the project on-time significantly with small budget amplifications.

**Table 20. The Significance Test for the Differences**

| **Protected ($\eta = 0.02$)** / **Non-Protected ($\eta = 0$)** | **CV = 0.25** | |
|---|---|---|
| | **$PM_1$** | **$PM_2$** |
| **$PM_1$** | (18.570, 24.790)* | |
| **PM2** | | (-0.334, -0.229)* |
| | **CV = 0.5** | |
| | **$PM_1$** | **$PM_2$** |
| **$PM_1$** | ( 2.841, 4.609)* | |
| **PM2** | | (-0.972, -0.576)* |

* indicates that test statistic for is significant at 5% level

Figure 6 shows the relationship between budget amplification and the $RM_9$. This figure depicts the behavior of the buffer size as a function of the percentage increase of the budget (i.e. 100η). The averages over all of the project instances included in the aforementioned test bed are reported in this figure. Figure 6 clearly demonstrates the strong correlation between budget increase and buffer size increase.

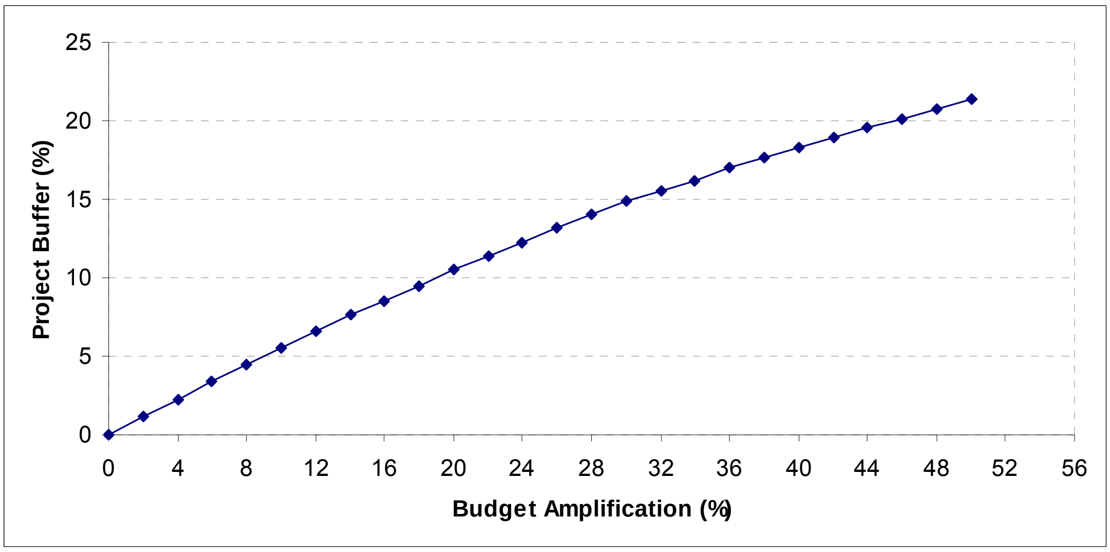

**Figure 6. The Relationship between Budget Amplification and Buffer Size Increase**

In order to assess the effectiveness of the proposed two-phase approach, we carry out an extensive simulation study. First, we set the budget amplification rate *η*. Then, for each instance of the test bed, we randomly generate perturbed processing times and compute the project completion time. Finally, we calculate the average values of $PM_1$ and $PM_2$ over all instances (recall that $PM_1$ represents the proportion of replicates that the project ends within the deadline, and that $PM_2$ represents the percentage excess of project completion time over the deadline). In our experiments, we use for each instance, a coefficient of variation (CV) of 0.25 and 0.5, respectively, to characterize small and moderate variability in the activity durations. Figure 7 and Figure 8 illustrate the simulation results and show the relationship between budget amplification and performances measures $PM_1$ and $PM_2$, respectively.

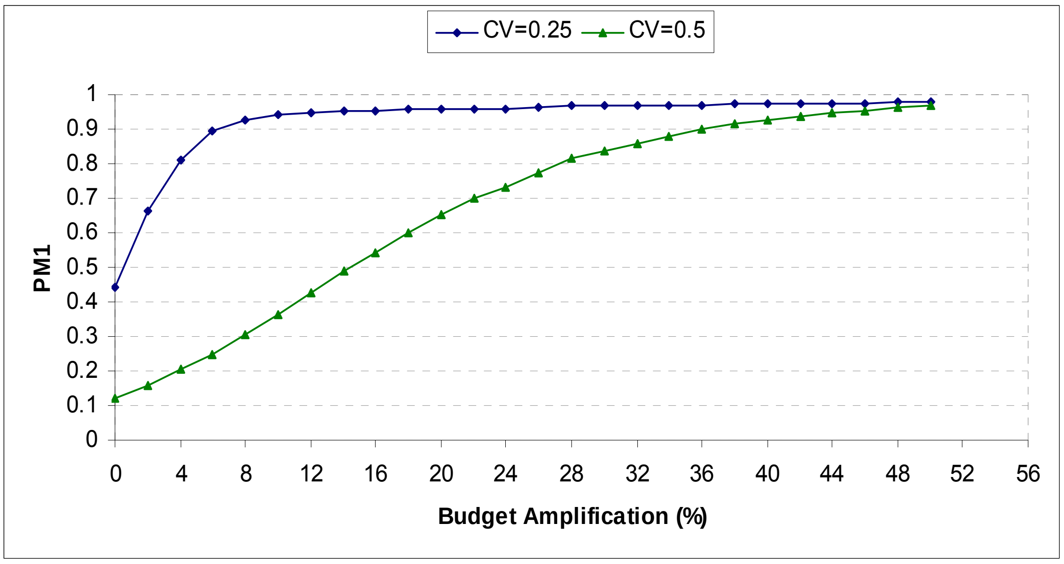

**Figure 7. The Relationship between Budget Amplification and PM $_1$**

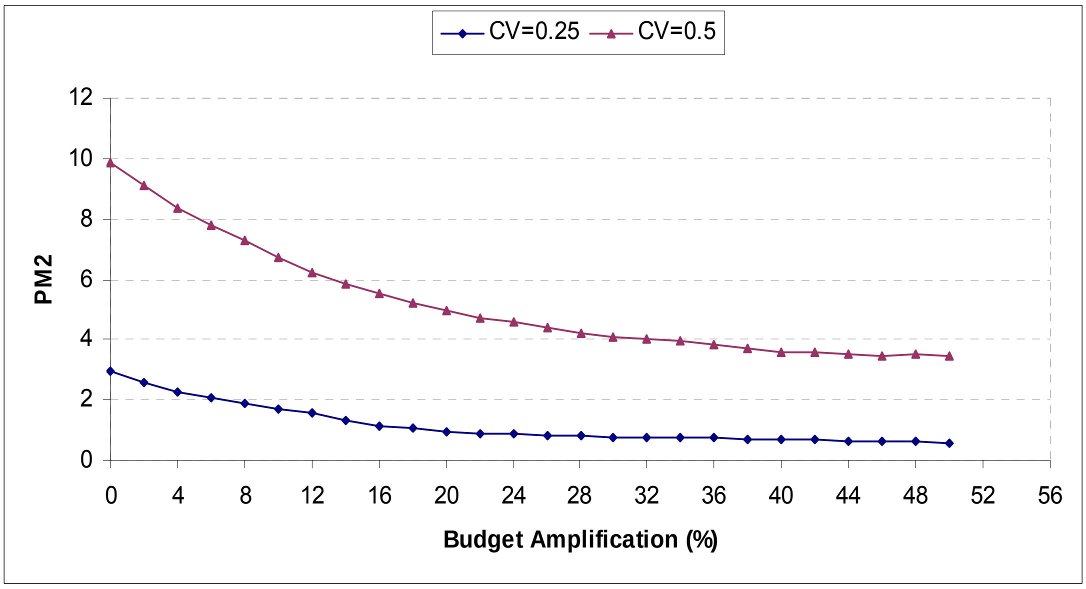

**Figure 8. The Relationship between Budget Amplification and $PM_2$**

Looking at these figures, we see that when variability is low, the performance measures could be significantly improved with small budget increases. Indeed, when *CV = 0.25*, with a 6% budget increase ($\eta = 6$), $PM_1$ increased from 44% to 89% and $PM_2$ decreased from 2.93% to 2.06%. The major effect of uncertainty in activity durations is that, when variability gets larger, much larger budget amplification is required to have the projects completed on time. For instance, when there is low uncertainty 7% is required to achieve 90% of projects completed on time whereas when there is high variability 36% amplification is required to have the same performance. We call the schedules obtained at the end of Phase 2 as *protected schedules*. When Figure 6 and Figure 7 is compared, it is evident that $RM_9$ and $PM_1$ (especially with CV= 0.5) give similar responses to budget amplifications and this similarity supports the strong correlation in between these measures (see Table 18).

In addition to the above mentioned performance measures ($PM_1$, $PM_2$), we also illustrate the average project completion time deviations from the project deadline in Figure 9. This measure evaluates the project lateness as it considers the negative deviations as well. Note that as uncertainty increases, much larger budget amplifications are required to complete the projects before the deadline. For example in Figure 5.4 when *CV = 0.25*, *3%* budget increase is sufficient, whereas when *CV = 0.5,* a budget increase of *16%* is required.

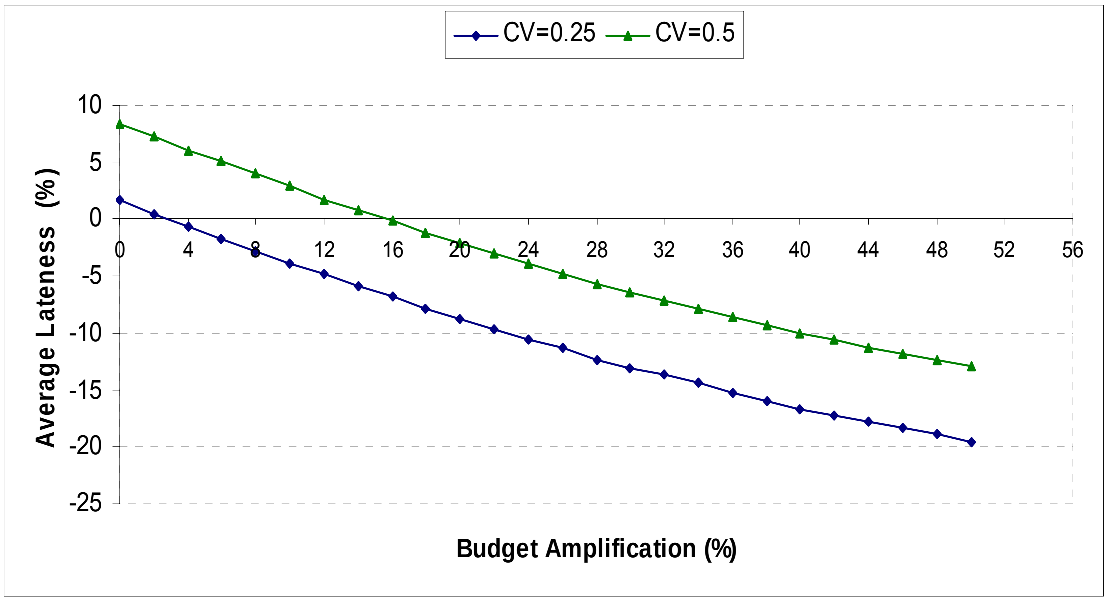

**Figure 9. The Relationship between Budget Amplification and Average Lateness**

So far, we have addressed the relationship between the allocated budget and various robustness measures. We show that using the two-phase approach, the larger the budget is and the more robust is the derived schedule. However, a crucial issue is to decide on the budget to be allocated and to determine the corresponding activity modes so that an optimal trade-off between cost and robustness is achieved. In the next section, firstly, we propose a model for this challenging issue and then we present the solution approach.

## 5.5. Analytical Study on Budget Allocation

### 5.5.3. A Model with Tardiness Penalties and Earliness Revenues

In order to model the trade-off between the budget and the performance measures, we use monetary units as a common basis to combine time and cost-based performance measures and we propose a model maximizing the net profit. A penalty cost is incurred in cases when the project finishes later than the due date. On the other hand, in cases of early completions, we assume that the enterprise can increase the profit by elongating the operating period. These additional profits are typical in BOT projects (characteristics of these types of projects are discussed in Chapter 3).

Thus, the problem is to determine the budget to be allocated (or equivalently, the increase coefficient $\eta$) such that the expected net profit:

$$h(\eta) = \rho \, Max\left\{0, \delta - C_{n+1}(\eta)\right\} - B_0(1+\eta) - \zeta Max\left\{0, C_{n+1}(\eta) - \delta\right\} \qquad (5.10)$$

is maximized. In the above function, $C_{n+1}(\eta)$ refers to the expected project completion time, $\rho$, and $\zeta$ denote the revenue rate and the tardiness penalty rate per unit time, respectively. The complexity of the model in (5.10) rests in finding out the expected completion times ($C_{n+1}(\eta)$). It is difficult to calculate exactly due to the complex network structures. For this purpose, we propose an analytical model to estimate the expected completion times. This approach aims to minimize computational effort by limiting the alternatives to be investigated. In the next section, a mathematical model is introduced to set the budget and generate the schedule so that the profit-based performance measure (5.10) is maximized.

### 5.5.4. Solution Approach

A striking observation in Figure 5.4 is that average lateness varies almost linearly with budget amplification. To evaluate this relationship, having checked the assumptions, we fit a linear regression model. For instance, when the data illustrated in Figure 5.4 is used, and *CV = 0.5,* the linearity assumption is valid for $0 \le \eta \le 0.20$ and $R^2$ = *0.997*. In this sequel, we have the following assumptions:

**A1:** The expected project completion time is a convex piecewise linear function of the budget amplification factor with *k* intervals. That is:

$$C_{n+1}(\eta) = a_0 + \sum_{j=1}^{i-1} b_{j-1}(\overline{\eta_j} - \overline{\eta_{j-1}}) + b_{i-1}(\eta - \overline{\eta_{i-1}}) \text{ if } \overline{\eta_{i-1}} \le \eta \le \overline{\eta_i} < \overline{\eta_k} \ , i = 1,\ldots,k-1 \qquad (5.11),$$

where $\overline{\eta_0} = 0$, $a_i$ and $b_i$ represents the intercepts and slopes of linear segments.

In practice, 2-4 segments would be reasonable to approximate the time/cost relationship in (5.11) well. It could be better approximated with more linear segments; however, it requires more computational effort as defining a segment requires simulating the project.

**A2:** The slopes are assumed to be negative and increasing with respect to budget increases, i.e. $b_i < b_{i+1} < 0$, $i = 1,\ldots,k$

This assumption expresses that the budget amplifications have diminishing rate of return. As the budget continues to increase, the marginal reduction in the project completion decreases. The following proposition defines the structure of the optimal solution.

**Proposition 5.1:** Given the profit function in 5.11 and assumptions A1, and A2 the optimal budget amplification policy is either one of the break points of the piecewise linear function or the critical point which makes the expected completion time zero.

**Proof:**

The proof rests on investigating all the possible parameter settings and finding out the optimal budget allocation in that setting. Given a budget allocation, the project could be either early or late and each case is considered separately due to the structure of (5.11).

In order to make a distinction between these two cases, we find the critical budget factor, $\eta_c$, such that $C_{n+1}(\eta_c) = 0$. If it is not possible to complete on time with the maximum budget, i.e. $C_{n+1}(\overline{\eta_k})a_o + \sum_{j=1}^{k} b_{j-1}(\overline{\eta_j} - \overline{\eta_{j-1}}) < \delta$, then we set $\eta_c = \overline{\eta_k}$. On the other hand if the project is expected to be early without any budget increase ($a_0 \leq \delta$), then we set $\eta_c = 0$.

The project is expected to be late when $0 \leq \eta \leq \eta_c$, hence in this case

$$h(\eta) = -B_0(1+\eta) - \zeta(a_o + \sum_{j=1}^{i-1} b_{j-1}(\overline{\eta_j} - \overline{\eta_{j-1}}) + b_{i-1}(\eta - \overline{\eta_{i-1}}) - \delta)$$

$$= \zeta\delta - \zeta(a_o + \sum_{j=1}^{i-1} b_{j-1}(\overline{\eta_j} - \overline{\eta_{j-1}}) - b_{i-1}\overline{\eta_{i-1}}) - B_0 - (B_0 + \zeta b_{i-1})\eta \qquad (5.12)$$

On the other hand, if $0 < \eta_c < \eta$, then project is expected to be early, hence

$$h(\eta) = \rho(\delta - a_o - \sum_{j=1}^{i-1} b_{j-1}(\overline{\eta_j} - \overline{\eta_{j-1}}) - b_{i-1}(\eta - \overline{\eta_{i-1}})) - B_0(1+\eta)$$

$$= \rho\delta - \rho(a_o + \sum_{j=1}^{i-1} b_{j-1}(\overline{\eta_j} - \overline{\eta_{j-1}}) - b_{i-1}\overline{\eta_{i-1}}) - B_0 - (B_0 + \rho b_{i-1})\eta \qquad (5.13)$$

Two additional parameters are required to be defined:

$d_1$ = Max$\{i\text{: } \overline{\eta_i} < \eta_c,\ i = 0, \ldots, k\}$, $d_2$ = Min$\{i\text{: } \overline{\eta_{i+1}} > \eta_c,\ i = 0, \ldots, k\}$. Three cases each containing three sub-cases could be identified such that each condition has specific optimality conditions:

**Case 1:** $\zeta \le \frac{-B_0}{b_0}$ :

In this case, $\zeta \le \frac{-B_0}{b_i}$ $i = 1,\ldots,k-1,$ due to A2.

**1a**. $\rho \le \frac{-B_0}{b_{d_2}}$ : Due to A2, $\rho \le \frac{-B_0}{b_i}$ $i = d_2,\ldots,k-1$.

This is the case that any budget amplification does not improve the profits. The optimal policy is $\eta^* = 0$.

**1b.** $\frac{-B_0}{b_{d_2}} < \rho \le \frac{-B_0}{b_{k-1}}$,

Define $f$ = Min $\{i\text{: } \rho < \frac{-B_0}{b_i},\ i = d_2+1,\ldots,k-1\}$. The optimal policy is $\eta^* = 0$ with profit: $h(0) = -\zeta(a_o - \delta) - B_0$ or $\eta^* = \overline{\eta_f}$ with $h(\overline{\eta_f}) = \rho(\delta - a_o - \sum_{j=1}^{f} b_{j-1}(\overline{\eta_j} - \overline{\eta_{j-1}})) - B_0(1 + \overline{\eta_f})$. The profits are compared and the comparison results in:

$$\eta^* = \begin{cases} 0 & \text{if } (a_o - \delta)(\zeta - \rho) - \rho \sum_{j=1}^{f} b_{j-1}(\overline{\eta_j} - \overline{\eta_{j-1}}) - B_0 \overline{\eta_f} \le 0 \\ \overline{\eta_f} & \text{o/w} \end{cases}$$

**1c.** $\frac{-B_0}{b_{k-1}} < \rho$ : due to A2, $\frac{-B_0}{b_i} < \rho$ $i = d_2,\ldots,k-1.$

In this case, optimal policy is either $\eta^* = 0$ with profit $h(0) = -\zeta(a_o - \delta) - B_0$ or

$\eta^* = \overline{\eta_k}$ with $h(\overline{\eta_k}) = \rho(\delta - a_o - \sum_{j=1}^{k} b_{j-1}(\overline{\eta_j} - \overline{\eta_{j-1}})) - B_0(1+\overline{\eta_k})$. Comparison results in:

$$\eta^* = \begin{cases} 0 & \text{if } (a_o - \delta)(\zeta - \rho) - \rho\sum_{j=1}^{k} b_{j-1}(\overline{\eta_j} - \overline{\eta_{j-1}}) - B_0\overline{\eta_k} \le 0 \\ \overline{\eta_k} & \text{o/w} \end{cases}$$

**Case 2:** $\frac{-B_0}{b_0} < \zeta \le \frac{-B_0}{b_{d_1}}$

Define $e$ = Min $\{i\!: \zeta < \frac{-B_0}{b_i}$, *i = 1,… , d₁*$\}$.

**2a.** $\rho \le \frac{-B_0}{b_{d_2}}$, $\eta^* = \overline{\eta_e}$

**2b.** $\frac{-B_0}{b_{d_2}} < \rho \le \frac{-B_0}{b_{k-1}}$:

Compare $h(\overline{\eta_e}) = -\zeta(a_o + \sum_{j=1}^{e} b_{j-1}(\overline{\eta_j} - \overline{\eta_{j-1}}) - \delta) - B_0(1+\overline{\eta_e})$ with

$h(\overline{\eta_f}) = \rho(\delta - a_o - \sum_{j=1}^{f} b_{j-1}(\overline{\eta_j} - \overline{\eta_{j-1}})) - B_0(1+\overline{\eta_f})$. Comparison results in

$$\eta^* = \begin{cases} \overline{\eta_e} & \text{if } (a_o - \delta)(\zeta - \rho) - \rho\sum_{j=1}^{f} b_{j-1}(\overline{\eta_j} - \overline{\eta_{j-1}}) + \zeta\sum_{j=1}^{e} b_{j-1}(\overline{\eta_j} - \overline{\eta_{j-1}}) - B_0(\overline{\eta_f} - \overline{\eta_e}) \le 0 \\ \overline{\eta_f} & \text{o/w} \end{cases}$$

**2c.** $\frac{-B_0}{b_{k-1}} < \rho$

Now compare $h(\overline{\eta_e}) = -\zeta(a_o + \sum_{j=1}^{e} b_{j-1}(\overline{\eta_j} - \overline{\eta_{j-1}}) - \delta) - B_0(1+\overline{\eta_e})$ with

$h(\overline{\eta_k}) = \rho(\delta - a_o - \sum_{j=1}^{k} b_{j-1}(\overline{\eta_j} - \overline{\eta_{j-1}})) - B_0(1+\overline{\eta_k})$. The optimal policy is:

$$\eta^* = \begin{cases} \overline{\eta_e} & \text{if } (a_o-\delta)(\zeta-\rho) - \rho\sum_{j=1}^{k} b_{j-1}(\overline{\eta_j} - \overline{\eta_{j-1}}) + \zeta\sum_{j=1}^{e} b_{j-1}(\overline{\eta_j} - \overline{\eta_{j-1}}) - B_0(\overline{\eta_k} - \overline{\eta_e}) \le 0 \\ \overline{\eta_k} & \text{o/w} \end{cases}$$

**Case 3:** $\frac{-B_0}{b_{d_1}} < \zeta$ : Due to A2, $\zeta \ge \frac{-B_0}{b_i}$ $i = 0,\ldots,d_1-1$

**3a.** $\rho \le \frac{-B_0}{b_{d_2}}$, The optimal policy is $\eta^* = \eta_c$.

**3b.** $\frac{-B_0}{b_{d_2}} < \rho \le \frac{-B_0}{b_{k-1}}$ , the optimal policy is $\eta^* = \eta_f$.

**3c.** $\rho \ge \frac{-B_0}{b_{k-1}}$, the optimal policy is $\eta^* = \overline{\eta_k}$.

The following table summarizes the cases and the corresponding optimal budget amplification policies:

**Table 21. Budget Amplification Policies**

| $\zeta$ \ $\rho$ | $\left[0, -\frac{B_0}{b_{d_2}}\right]$ | $\left(-\frac{B_0}{b_{d_2}}, -\frac{B_0}{b_{k-1}}\right]$ | $\left(-\frac{B_0}{b_{k-1}}, \infty\right)$ |
|---|---|---|---|
| $\left[0, -\frac{B_0}{b_0}\right]$ | $\eta^* = 0$ | $\eta^* \in \{0, \overline{\eta_f}\}$ | $\eta^* \in \{0, \overline{\eta_k}\}$ |
| $\left(-\frac{B_0}{b_0}, -\frac{B_0}{b_{d_1}}\right]$ | $\eta^* = \overline{\eta_e}$ | $\eta^* \in \{\overline{\eta_e}, \overline{\eta_f}\}$ | $\eta^* \in \{\overline{\eta_e}, \overline{\eta_f}\}$ |
| $\left(-\frac{B_0}{b_{d_1}}, \infty\right)$ | $\eta^* = \eta_c$ | $\eta^* = \eta_f$ | $\eta^* = \overline{\eta_k}$ |

Considering all the possibilities summarized in Table 21, the optimal budget amplification policy is defined with either one of the break points or the critical point which makes the expected completion time zero.

Q.E.D.

As an immediate result of Proposition 5.1, we propose the following algorithm to allocate the budget.

1. Initialization:
   - **a.** Define the minimum and maximum budgets, $B_0$, and $B_{max}$.

     Define the number of intervals, *k*, to approximate this time/cost. Partition $\left[0, \overline{\eta_k}\right]$ into *k* segments, where $\overline{\eta_k} = \frac{B_{\max} - B_0}{B_0}$.
   - **b.** Given $B = B_0$, generate the schedule by solving the DTCTP-B exactly.
   - **c.** Invoke Monte Carlo Simulation (the activity durations are perturbed while executing the schedule in the simulation run), and find out the expected completion time, set $a_0 = C_{n+1}(0)$.
2. For $i = 1, \dots, k$
   - **a.** Set the budget, $B = (1 + \overline{\eta_i})\, B_0$.
   - **b.** Given *B*, generate the schedule by solving the DTCTP-B exactly.
   - **c.** Invoke Monte Carlo Simulation and find the expected completion time, $C_{n+1}(\overline{\eta_i})$. Then, set $b_{i-1} = \frac{C_{n+1}(\overline{\eta_i}) - C_{n+1}(\overline{\eta_{i-1}})}{\overline{\eta_i} - \overline{\eta_{i-1}}}$, and $a_i = a_{i-1} + b_{i-1}(\overline{\eta_i} - \overline{\eta_{i-1}})$.

3. Having defined the parameters, $\rho, \zeta$, $B_0$, $a_i$, $b_i$ $i = 0,\dots,k$, use Table 21 to find out the optimal policy.

The major advantage of the proposed analytical approach is that it limits the number of computations of the expected project completion time. The expected completion time of a project is estimated by invoking the scheduling method and simulating the project. Now we only calculate it for at most $k+1$ budget settings. These are the break-points of the piecewise linear function. Furthermore, Table 21 illustrates that optimal policy is either one of the break points or the critical point that expected completion time is zero.

## 5.6. Conclusion

In this chapter, we have introduced several robustness measures and a two-phase robust scheduling algorithm for multi-mode project networks. We have investigated the pertinence of the robustness measures by using simulation. The test results indicate that the buffer size is the most appropriate robustness measure. Based on this finding, we have proposed a two-phase methodology for generating robust schedules, which are protected against variability in activity durations. Afterwards we have performed computational experiments to assess the effectiveness of the algorithm under various problem settings. The results revealed that a slight increase in the budget results in significant improvements in the performance measures. Furthermore, the budget amplification consistently improves the schedule robustness. Therefore, we have examined the trade-off between project cost and schedule robustness. To model this trade-off, we have developed an analytical model and proposed an appropriate solution strategy.

To the best of our knowledge, the research in this chapter is the first work that concentrates on robustness measures and robust scheduling algorithms for multi-mode project networks. As a future extension of this research, deriving robust schedules for the multi-mode resource constrained project scheduling problem with crashable modes (Ahn and Erenguc, 1998) could be addressed. This problem might be viewed as a natural combination of the well-known multi-mode resource constrained project scheduling problem and the time/cost trade-off problem. We expect that the results presented in this chapter might provide a useful base for investigating this challenging problem.

# CHAPTER 6

# GENERAL DISCUSSION AND CONCLUSIONS

In this dissertation, we have investigated the deterministic and robust discrete time/cost trade-off problems, formulated models and developed solution algorithms for these problems. These challenging project scheduling problems are practically relevant as it is often possible to reduce the duration of activities with additional expenses and there exists multiple activity processing alternatives in real life projects.

We have addressed three major research issues. Firstly, we have investigated the issue of how to solve large-scale deterministic DTCTP instances to optimality. The second issue deals with how to generate robust project schedules efficiently. Finally, the third issue deals with the development of new approaches and measures for the assessment of project schedule's robustness. The contribution of this dissertation lies in:

**i)** the algorithms developed to solve the large scale deterministic DTCTP instances exactly,

**ii)** the approaches and mathematical models developed to generate robust project schedules,

**iii)** the formal quantitative metrics to assess robustness of project schedules.

First, we have investigated deterministic project environments and have proposed Benders Decomposition-based algorithms to solve two variants of the deterministic DTCTP: the deadline and budget versions. Although Benders Decomposition is known to exhibit slow convergence, we have included several features to accelerate its convergence and make it feasible to solve large-scale instances to optimality. We have solved the deadline and the budget problems for project networks with up to 136 activities exactly. Extensive computational experiments have been performed to measure the efficiency of the algorithm under various problem settings and investigate the interactions in between problem parameters.

Secondly, we have incorporated uncertainty into the problem and have formulated robust optimization DTCTP models. Interval uncertainty is assumed for unknown cost parameters and three alternative robust optimization models have been proposed. We have compared the schedules that have been generated with these models on the basis of schedule robustness. Furthermore, we have assessed the performance of the proposed algorithms under various experimental problem settings. Until now, DTCTP has been studied under the assumption that problem parameters are completely known. To the best of our knowledge, the models introduced in this dissertation are the first attempt to optimize DTCTP under uncertainty.

In order to assess the robustness of project schedules, we have proposed some formal quantitative robustness metrics and experimentally tested the applicability of these metrics using simulation. Among various alternatives, the measures that are calculated easily for a given baseline schedule and that provide a good estimate of schedule robustness are emphasized. We have showed that the buffer size is the most appropriate robustness measure regardless of the network complexity. Based on this finding, we have proposed a two-phase methodology for generating robust schedules. In the first phase of the methodology, we set the minimum required budget. Next, in the second phase, this budget is slightly inflated by a specified amplification factor and then the buffer size is maximized. We have provided strong empirical evidence that the budget amplification consistently improves the schedule robustness. Therefore, we have addressed the important issue of determining the best trade-off between the project cost and the schedule robustness. To that aim, we have proposed an extended model involving both tardiness penalties and earliness revenues and we have described an appropriate solution strategy that requires a restricted number of simulations.

The scheduling algorithms developed in this dissertation respond to the crucial need to build robust project schedules that are less vulnerable to disruptions caused by uncontrollable factors. Furthermore, they serve as a basis to develop decision support systems (DSS) that will help project managers in planning under uncertain environments. As multi-mode scheduling is addressed, managers could plan considering multiple activity processing alternatives. Practical impact of the proposed models and algorithms could be observed via developing an integrated DSS. They comprise the optimization module of an integrated DSS. This DSS would counsel project managers to generate protected schedules that help to minimize the deviations from the time and cost-based project objectives proactively. Integrating the DSS into widely used commercial software packages would be very valuable in practice as it enhances the scheduling functions of these packages.

Furthermore, as a future extension of this research, robust optimization models and robustness metrics could be formulated for Multi-Mode Resource Constrained Project Scheduling Problem (MRCPSP), which allows the use of both renewable and nonrenewable resources. For this new generalized problem setting, in addition to the uncertainty in activity durations or costs, the uncertainty in resource requirements or in resource availabilities could also be addressed. These extension alternatives will serve to model the project environments more realistically and work to generate schedules that are protected against various kinds of uncertainty. We believe that the results presented in this dissertation might prove as a useful base for investigating multi-mode resource project scheduling addressing other sources of uncertainty.

# BIBLIOGRAPHY

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: 250–9.

Ahuja, R.K., T.L. Magnanti and J.B.Orlin. 1993. *Network Flows. Theory, Algorithms and Applications*.NJ, USA: Prentice-Hall.

Akkan, C., A. Drexl and A. Kimms. 2005. "Network Decomposition-Based Benchmark Results for the Discrete Time–Cost Trade-off Problem," *European Journal of Operational Research* 165: 339–358.

Al-Fawzan, M.A. and M. Haouari. 2005. "A Bi-objective Model for Robust Resource-Constrained Project Scheduling," *International Journal of Production Economics* 96: 175-187.

Atamturk, A. and M.W.P. Savelsbergh. 2005. "Integer-Programming Software Systems," *Annals of Operations Research* 140: 67-124.

Ballestín, F. 2008. "When It Is Worthwhile to Work with the Stochastic RCPSP?," *Journal of Scheduling* article in press.

Bein, W.W., J. Kamburowski and M.F.M. Stallmann, 1992. "Optimal Reduction of Two-Terminal Directed Acyclic Graphs," *SIAM Journal on Computing* 21: 1112–1129.

Bellman, R., and, L. A. Zadeh (1970). "Decision-making in a fuzzy environment," *Management Science*, 17, 141–161.

Benders, J. F. 1962. "Partitioning Procedures for Solving Mixed Variables Programming Problems," *Numerische Mathematic* 4: 238-252.

Ben-Tal, A. and A. Nemirovski. 1999. "Robust Solutions of Uncertain Linear Programs," *Operations Research Letters* 25: 1-13.

Ben-Tal, A. and A. Nemirovski. 2000. "Robust Solutions of Linear Programming Problems Contaminated with Uncertain Data," *Mathematical Programming* 88: 411-424.

Ben-Tal, A. and et al. 2004. "Adjustable Robust Solutions of Uncertain Linear Programs," *Mathematical Programming* 99: 351-376.

Bertsimas, D. and M. Sim. 2003. "Robust Discrete Optimization and Network Flows," *Mathematical Programming* 98: 49-71.

Bertsimas, D. and M. Sim. 2004. "The Price of Robustness," *Operations Research* 52: 35-53.

Birge, J.R. and F. Louveaux. 1997. *Introduction to Stochastic Programming.* New York: Springer-Verlag.

Box, G.E.P and M.E. Muller. 1958. "A Note on the Generation of Random Normal Deviates," *Annals of Mathematical Statistics* 29:610-611.

Blazewicz, J., J.K Lenstra and A.H.G. Rinnooy Kan. 1983. "Scheduling Subject to Resource Constraints: Classification and Complexity, " *Discrete Applied Mathematics* 5: 11-24.

Brucker, P. and et al. 1999. "Resource-Constrained Project Scheduling: Notation, Classification, Models, and Methods," *European Journal of Operational Research* 112(1): 3-41.

Charnes, A., and W.W. Cooper, 1959. "Chance-constrained programming," *Management Science*, 6, 73–79.

Chtourou, H. and M. Haouari. 2008. “A Two-Stage-Priority-Rule-Based Algorithm for Robust Resource-Constrained Project Scheduling” *Computers and Industrial Engineering* article in press.

Cohen, I., B. Golany and A. Shtub. 2008. "The Stochastic Time/Cost Trade-off Problem: A Robust Optimization Approach," *Networks* article in press.

Costa, A. M. 2005. "A Survey on Benders Decomposition Applied to Fixed Charge Network Design Problems," *Computers and Operations Research* 32:1429–1450.

Danna, E., E. Rothberg and C. Le Pape. 2005. “Exploring Relaxation Induced Neighborhoods to Improve MIP Solutions,” *Mathematical Programming Series.* A 102: 71–90.

De Meyer, A., C. Loch and M. Pich. 2002. “Managing Project Uncertainty: From Variation to Chaos,” *Sloan Management Review*, 43(2), 60-68.

De, P. and et al. 1995. "The Discrete Time/cost Trade-Off Problem Revisited," *European Journal of Operational Research* 81: 225-238.

De, P. and et al. 1997. “Complexity of the Discrete Time/Cost Trade-Off Problem for Project Networks," *Operations Research* 45: 302-306.

Demeulemeester, E., W. 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: 50–68.

Demeulemeester, E. and et al. 1998. “New Computational Results for the Discrete Time/Cost Trade-Off Problem in Project Networks,” *Journal of the Operational Research Society*, 49: 1153–1163.

Erenguc, S.S., S. Tufekci and C.J. Zappe. 1993. "Solving Time/Cost Trade-Off Problems with Discounted Cash Flows Using Generalized Benders Decomposition," *Naval Research Logistics Quarterly* 40: 25-50.

Fernandez, A.A., R.L. Armacost and J. Pet-Edwards. 1998. “Understanding Simulation Solutions to Resource-Constrained Project Scheduling Problems with Stochastic Task Durations,” *Engineering Management Journal* 10: 5–13.

Gal, T. and H.J. Greenberg, 1997, *Advances in Sensitivity Analysis and Parametric Programming*, Kluwer Academic Publishers, London.

Goldratt, E.M., and J.Cox, 1984. *The Goal.* North River Press, Croton-on-Hudson, NY.

Goldratt, E.M. 1997. *Critical Chain*. Great Barrington: The North River Press Publishing Corporation, MA.

Golenko-Ginzburg, D., and A. Gonik. 1997. "Stochastic Network Project Scheduling with Non-Consumable Limited Resources," *International Journal of Production Economics* 48: 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, 149–162.

Gören, S. and İ. Sabuncuoğlu. 2008. "Robustness and Stability Measures for Scheduling: Single Machine Environment," *IIE Transactions* 40: 66–83.

Gu, Z., G.L. Nemhauser, and M.W.P. Savelsbergh. 1998. "Cover Inequalities for 0–1 Linear Programs: Computation," *INFORMS Journal on Computing* 10: 427–437.

Gutjahr, W.J., C. Strauss and E. Wagner. 2000. "A Stochastic Branch-and-Bound Approach to Activity Crashing in Project Management," *INFORMS Journal on Computing* 12(2): 125-135.

Hall, N.G. and M.E. Posner. 2004. "Sensitivity Analysis for Scheduling Problems," *Journal of Scheduling* 7(1): 49–83.

Hapke, M. and R. Slowinski. 1994. "Fuzzy Project Scheduling System for Software Development," *Fuzzy Sets and Systems* 67: 101–117.

Hapke, M. and R. Slowinski. 1996. "Fuzzy Priority Heuristics for Project Scheduling," *Fuzzy Sets and Systems* 83: 291– 299.

Harvey, R.T. and J.H. Patterson. 1979. "An Implicit Enumeration Algorithm for the Time/Cost Trade-off Problem in Project Network Analysis," *Foundations of Control Engineering* 4: 107-117.

Hazır, Ö., Y. Günalay Y. and E. Erel. 2008. "Customer Order Scheduling Problem: A Comparative Metaheuristics Study," *International Journal of Advanced Manufacturing Technology* 37: 589–598.

Herroelen, W., E. Demeulemeester and B. De Reyck. 1998. "Resource Constrained Project Scheduling- A Survey of Recent Developments," *Computers and Operations Research* 25(4): 279–302.

Herroelen, W. and R. Leus. 2001. "On the Merits and Pitfalls of Critical Chain Scheduling," *Journal of Operations Management* 19: 559–577.

Herroelen, W. and R. Leus. 2003. "The Construction of Stable Project Baseline Schedules," *European Journal of Operational Research* 156: 550–565.

Herroelen, W. and R. Leus. 2005. "Project Scheduling Under Uncertainty – Survey and Research Potentials," *European Journal of Operational Research* 165: 289–306.

Hindelang, T.J. and J.F. Muth. 1979. "A Dynamic Programming Algorithm for Decision CPM Networks," *Operations Research* 27: 225-241.

İçmeli, O., S.S. Erengüç. and C.J. Zappe. 1993. "Project Scheduling Problems: A Survey," *International Journal of Operations and Production Management* 13(11): 80–91.

Karaşan, O.E, M.Ç. Pınar and H. Yaman. 2001. "*The Robust Shortest Path Problem with Interval Dat*a". Ankara: Bilkent University.

Kelley E.J., and M.R. Walker. 1959 *"Critical-Path Planning and Scheduling,"* Proc. Eastern Joint Computer Conference, Boston, Dec. 1-3, 160-173.

Kerzner, H. 2006. *Project management. A systems approach to planning, scheduling and controlling*. (9th ed.). New York: John Wiley and Sons.

Kobylański, P. and D. Kuchta. 2008. “A note on the paper by M. A. Al-Fawzan and M. Haouari about a bi-objective problem for robust resource-constrained project scheduling,” *International Journal of Production Economics* article in press.

Kolisch, R. and R. Padman. 2001. "An Integrated Survey of Deterministic Project Scheduling,” *Omega* 29: 249–272.

Kouvelis, P. and G. Yu. 1997. *Robust Discrete Optimization and Its Applications.* Norwell: Kluwer Academic Publishers.

Kulturel-Konak, S. and et al. 2004, “Exploiting Tabu Search Memory in Constrained Problems,” *INFORMS Journal on Computing*, 16(3): 241–254.

Kuyumcu, A. and A. Garcia-Diaz. 1994. “A Decomposition Approach to Project Compression with Concave Activity Cost Functions,” *IIE Transactions* 26(6): 63-73.

Lambrechts, O., E. Demeulemeester and W. Herroelen. 2008a. “A Tabu Search Procedure for Developing Robust Predictive Project Schedules,” *International Journal of Production Economics* 111(2): 493-508.

Lambrechts O, E. Demeulemeester, W. Herroelen. 2008b. “Proactive and Reactive Strategies for Resource-Constrained Project Scheduling with Uncertain Resource Availabilities," *Journal of Scheduling* article in press.

Law, A. M., and W. D. Kelton. 2000. *Simulation Modeling and Analysis* (3rd ed). New York: McGraw-Hill.

Leon, V.J., S.D. Wu and R.H. Storer. 1994. "Robustness Measures and Robust Scheduling for Job Shops," *IIE Transactions* 26(5): 32-43.

Leus*, R. 2003.* "The Generation of Stable Project Plans Complexity and Exact Algorithms,*" Ph.D. dissertation, Department of Applied Economics, Katholieke Universiteit Leuven, Belgium*.

Leus, R. and W. Herroelen. 2004. "Stability and Resource Allocation in Project Planning*,"* *IIE Transactions* 36: 667–682.

Linderoth, J. and M.W.P. Savelsbergh. 1999. “A Computational Study of Search Strategies for Mixed Integer Programming,” *INFORMS Journal on Computing* 11: 173–187.

Maccrimmon R.K and C.A. Ryavec, 1964 “An Analytical Study of the PERT Assumptions” *Operations Research* 12(1): 16-37.

Maniezzo, V. and A. Mingozzi. 1999. "A Heuristic Procedure for the Multi-Mode Project Scheduling Problem Based On Bender’s Decomposition." In J. Weglarz, ed., *Project Scheduling - Recent Models, Algorithms and Applications.* Boston: Kluwer Academic Publishers: 179-96.

McDaniel, D. and M. Devine. 1977. "A Modified Benders’ Partitioning Algorithm for Mixed Integer Programming," *Management Science* 24: 312-379.

Meredith, J.R. and S.J. Jr. Mantel. 2005. *Project Management A Managerial Approach* (6th ed.). New York: John Wiley and Sons.

Montemanni, R., L.M. Gambardella and A.V. Donati. 2004. “A Branch-and-bound Algorithm for the Robust Shortest Path Problem with Interval Data,” *Operations Research Letters* 32(3): 225-232.

Montemanni, R. and L.M. Gambardella. 2005. “A Branch-and-bound Algorithm for the Robust Spanning Tree Problem with Interval Data," *European Journal of Operational Research* 161(3): 771-779.

Mulvey, J.M., R.J. Vanderbei and S.A. Zenios. 1995. “Robust Optimization of Large-Scale Systems, ” *Operations Research* 43: 264-281.

Pascoe, T.L. 1966. “Allocation of Resources–CPM,” *Revue Française de Recherche Opérationelle.* 38: 31–38.

Project Management Institute 2004. A Guide to the Project Management Body of Knowledge:PMBOK Guide, (3rd ed.) Newton Square, Pennsylvania; Project Management Institute.

Raz T. et al. 2003. "A Critical Look at Critical Chain Project Management," *Project Management Journal* 34(4): 24-32.

Robinson, D.R. 1975. "A Dynamic Programming Solution to the Cost–Time Trade off for CPM," *Management Science* 22: 158–166.

Sabuncuoğlu, I. and M. Bayız. 2000. "Analysis of Reactive Scheduling Problems in Job Shop Environment," *European Journal of Operational Research* 126: 567-586.

Sabuncuoğlu, İ. and S. Gören. 2005. "A review of reactive scheduling research: proactive scheduling and new robustness and stability measures," Technical Report, IE/OR 2005-02, Department of Industrial Engineering, Bilkent University, Ankara.

Skutella, M. 1998. "Approximation Algorithms for the Discrete Time–Cost Trade-off Problem," *Mathematics of Operations Research* 23: 195-203.

Stork, F. 2001. "Stochastic Resource-Constrained Project Scheduling," *Ph.D. thesis, Technische Universität Berlin*, Berlin, Germany.

Tavares, L.V., J.A.A. Ferreiraand and J.S. Coelho. 1998. "On the Optimal Management of Project Risk," *European Journal of Operational Research* 107: 451-469.

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: 129–141.

Tukel, O.I., O.R. Walter and S.D. Ekşioğlu. 2006. "An Investigation of Buffer Sizing Techniques in Critical Chain Scheduling," *European Journal of Operational Research* 172(2): 401-416.

Valls, V. and et al. 1998. "Project Sheduling with Stochastic Activity Interruptions." In J. Werglarz, ed., *Project Scheduling: Recent Models, Algorithms and Applications*. Boston: Kluwer Academic Publishers, 333-353.

Van de Vonder, S. and et al. 2005. "The Use of Buffers in Project Management: The Trade-Off Between Stability and Makespan," *International Journal of Production Economics* 97: 227-240.

Van de Vonder, S. and et al. 2006. "The Trade-Off Between Stability and Makespan in Resource-Constrained Project Scheduling," *International Journal of Production Research* 44(2): 215-236.

Van De Vonder, S. and et al. 2007a. "Heuristic Procedures for Reactive Project Scheduling," *Computers and Industrial Engineering* 52(1): 11-28.

Van De Vonder, S., E. Demeulemeester and W. Herroelen. 2007b. "A Classification of Predictive- Reactive Project Scheduling Procedures," *Journal of Scheduling* 10(3): 195-207. Special Issue on Project Scheduling under Uncertainty (Demeulemeester, E.L. and Herroelen W.S. (eds.)).

Van De Vonder, S., E. Demeulemeester and W. Herroelen. 2008. "Proactive Heuristic Procedures for Robust Project Scheduling: An Experimental Analysis," *European Journal of Operational Research*. 189(3): 723-733.

Vanhoucke, M. and D. Debels. 2008. "The Discrete Time/Cost Trade-Off Problem under Various Assumptions Exact and Heuristic Procedures" *Journal of Scheduling* article in press.

Walker, C., and A.J. Smith. 1995. *Privatized Infrastructure: The Build Operate Transfer Approach*. London: Thomas Telford.

Wang, J. 2002. "A Fuzzy Project Scheduling Approach to Minimize Schedule Risk for Product Development," *Fuzzy Sets and Systems* 127 (2): 99–116.

Wang, J. 2004. "A Fuzzy Robust Scheduling Approach for Product Development Projects," *European Journal of Operational Research* 152: 180–194.

Wolsey, L.A. 1998. *Integer Programming*. New York: John Wiley and Sons.

Yaman, H., O.E. Karasan and M.C. Pinar. 2001. "The Robust Spanning Tree Problem with Interval Data," *Operations Research Letters* 29:31-40.

Yamashita, D.S., V.A. Armentano and M. Laguna. 2008. "Robust Optimization Models for Project Scheduling with Resource Availability Cost, "*Journal of Scheduling* article in press.

Yang, K.K. 1996. "Effects of Erroneous Estimation of Activity Durations on Scheduling and Dispatching A Single Project," *Decision Sciences* 27(2): 255-290.

Yen, J.Y. 1971. "Finding the K Shortest Loopless Paths in a Network". *Management Science* 17: 712–716.

Zimmermann, H.J. 2001. *Fuzzy set theory and its applications* (4th ed.). . Kluwer Academic Publishers Boston.

Zhu, G., J.F. Bard and G. Yu. 2005. "Disruption Management for Resource-Constraint Project Scheduling," *Journal of the Operational Research Society* 56: 365–381.

Zhu, G., J.F. Bard and G. Yu. 2008. "A Two-Stage Stochastic Programming Approach for Project Planning with Uncertain Activity Durations," *Journal of Scheduling* article in press.

# APPENDIX

## Illustrative Examples for Solving Discrete Time Cost Trade-off Problems with Benders Decomposition

## AN ILLUSTRATIVE EXAMPLE FOR DTCTP-D:

Consider the simple project network in Figure 10 (the example of De et al., 1995). Each activity has two mode alternatives and for each mode alternative, $(p_{i1},c_{i1})$ and $(p_{i2}, c_{i2})$ are given above the nodes in the figure. The project has a deadline of *6* units, i.e. $\delta = 6$.

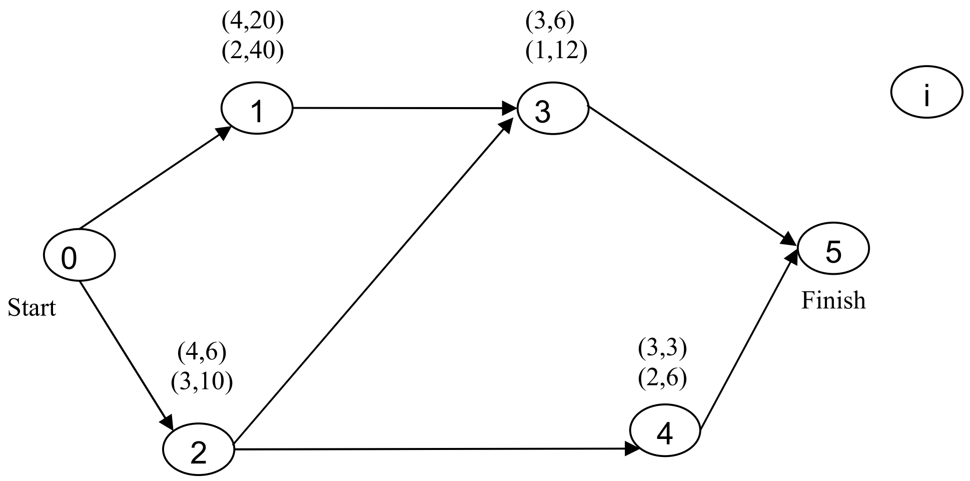

**Figure 10. The Example Network (Deterministic Problems)**

For the given problem, the primal and dual solutions are represented with vectors $x^T = [x_{11},x_{12},x_{21},x_{22},x_{31},x_{32},x_{41},x_{42}]$ and $w^T = [w_{01}, w_{02}, w_{13}, w_{23}, w_{24}, w_{35}, w_{45}]$, respectively. The procedure to solve the illustrative example by using the algorithm outlined above could be summarized as follows:

**Initialization:** $\overline{x^1} = [0, 1, 0, 1, 0, 1, 0, 1]$, $LB = -\infty$ $UB = \infty$.

**Iteration 1:**

- Solve $SP(\overline{x^1}) : C_5 = 5$.
- Feasible and bounded primal and dual solutions: $UB = \sum_{j\in N}\sum_{m\in M_j} c_{jm}\overline{x^0_{jm}} = 68$.
- Solve MP$^1$:

Min $20x_{11}+40x_{12}+6x_{21}+10x_{22}+6x_{31}+12x_{32}+3x_{41}+6x_{42}$

subject to

$x_{11}+x_{12}=1$

$x_{21}+x_{22}=1$

$x_{31}+x_{32}=1$

$x_{41}+x_{42}=1$

$x_{11},x_{12},x_{21},x_{22},x_{31},x_{32},x_{41},x_{42} \in \{0,1\}$

- $x^1 = [1,0,1,0,1,0,1,0]$, $z^1 = \sum_{j\in N}\sum_{m\in M_j} c_{jm}x^1_{jm} = 35$,
- $t=2$, $\overline{x^2} = x^1 = [1,0,1,0,1,0,1,0]$ .

**Iteration 2:**

- Solve $SP(\overline{x^2})$: $C_5 = 7$.
- Infeasible primal, unbounded dual solutions: $\overline{w^2} = [1, 0, 1, 0, 0, 1, 0]$.
- Solve MP$^2$:

Min $20x_{11}+40x_{12}+6x_{21}+10x_{22}+6x_{31}+12x_{32}+3x_{41}+6x_{42}$

subject to

$x_{11}+x_{12}=1$

$x_{21}+x_{22}=1$

$x_{31}+x_{32}=1$

$x_{41}+x_{42}=1$

$4x_{11}+2x_{12}+3x_{31}+x_{32}\leq 6$

$x_{11},x_{12},x_{21},x_{22},x_{31},x_{32},x_{41},x_{42} \in \{0,1\}$

- $x^2 = [1,0,1,0,0,1,1,0]$, $z^2 = \sum_{j\in N}\sum_{m\in M_j} c_{jm}x^2_{jm} = 41$.
- $t=3$, $\overline{x^3} = x^2 = [1,0,1,0,0,1,1,0]$.

**Iteration 3:**

- Solve $SP(\overline{x^3})$; $C_5 = 7$.
- Infeasible primal, unbounded dual solutions: $\overline{w^3} = [0, 1, 0, 0, 1, 0, 1]$
- Solve MP$^3$:

Min $20x_{11}+40x_{12}+ 6x_{21}+10x_{22}+6x_{31}+12x_{32}+3x_{41}+6x_{42}$

subject to

$4 x_{11} +2x_{12}+3 x_{31}+ x_{32} \leq 6$

$4 x_{21} +3x_{22}+3 x_{41}+ 2x_{42} \leq 6$

$x_{11}+x_{12}=1$

$x_{21}+x_{22}=1$

$x_{31}+x_{32}=1$

$x_{41}+x_{42}=1$

$x_{11},x_{12},x_{21},x_{22},x_{31},x_{32},x_{41},x_{42} \in \{0,1\}$

- $x^3 = [1,0,1,0,0,1,0,1],\ z^3 = \sum_{j \in N} \sum_{m \in M_j} c_{jm} x^3_{jm} = 44.$
- $t = 4,\ \overline{x^4} = x^3 = [1,0,1,0,0,1,0,1].$

**Iteration 4:**

- Solve $SP^2(\overline{x^4}),\ C_5 = 6$
- Feasible and bounded primal and dual solutions:

Terminate with $x^* = [1,0,1,0,0,1,0,1],\ z^* = \sum_{j \in N} \sum_{m \in M_j} c_{jm} \overline{x^4_{jm}} = 44.$

**AN ILLUSTRATIVE EXAMPLE FOR THE DTCTP-B:**

We solve the budget problem with $B_0 = 44$ .

**Initialization:** $\overline{x^1} = [1, 0, 1, 0, 1, 0, 1, 0],\ LB = -\infty\ \ UB = \infty$,

**Iteration 1:**

- Solve $SP\ (\overline{x^1})$: $UB = C_5 = 7.$
- $\overline{w^1} = [1, 0, 1, 0, 0, 1, 0]$
- Solve $\mathrm{MP}^1$ :

Min $z$

subject to

$x_{11}+x_{12}=1$

$x_{21}+x_{22}=1$

$x_{31}+x_{32}=1$

$x_{41}+x_{42}=1$

$20x_{11}+40x_{12}+ 6x_{21}+10x_{22}+6x_{31}+12x_{32}+3x_{41}+6x_{42}\leq 44$

$z\geq 4 x_{11} +2x_{12}+3 x_{31}+ x_{32}$

$x_{11},x_{12},x_{21},x_{22},x_{31},x_{32},x_{41},x_{42} \in \{0,1\}$

- $x^1 = [1,0,1,0,0,1,1,0]$, $LB = 5$
- $t =2$, $\overline{x^2} = x^1 = [1,0,1,0,0,1,1,0]$.

**Iteration 2:**

- Solve $SP(\overline{x^2})$: $UB = C_5 = 7$.
- $\overline{w^2} = [0, 1, 0, 0, 1, 0, 1]$.
- Solve MP$^2$:

Min $z$

subject to

$x_{11}+x_{12}=1$

$x_{21}+x_{22}=1$

$x_{31}+x_{32}=1$

$x_{41}+x_{42}=1$

$20x_{11}+40x_{12}+ 6x_{21}+10x_{22}+6x_{31}+12x_{32}+3x_{41}+6x_{42}\leq 44$

$z\geq 4 x_{11} +2x_{12}+3 x_{31}+ x_{32}$

$z\geq 4 x_{21} +3x_{22}+3 x_{41}+ 2x_{42}$

$x_{11},x_{12},x_{21},x_{22},x_{31},x_{32},x_{41},x_{42} \in \{0,1\}$

- $x^2= [1,0,1,0,0,1,0,1]$, $LB =6$.
- $t = 3$, $\overline{x^3} = x^2 = [1,0,1,0,0,1,0,1]$.

**Iteration 3:**

- Solve $SP(\overline{x^3})$; $UB = C_5 = 6$.
- $\overline{\mathrm{w}^3} = [0, 1, 0, 0, 1, 0, 1]$.
- $UB = LB$

Terminate with $x^* = [1,0,1,0,0,1,0,1]$, $z^* = 6$.