Comprehensive CP Optimization for Dynamic Scheduling in Construction

by

Zinab Abuwarda Mohamed

A thesis

presented to the University of Waterloo

in fulfillment of the

thesis requirement for the degree of

Doctor of Philosophy

in

Civil Engineering

Waterloo, Ontario, Canada, 2018

Examining Committee Membership

The following served on the Examining Committee for this thesis. The decision of the Examining Committee is by majority vote.

External Examiner	NAME: Khaled El-Rayes Title: Professor, Civil & Environmental Engineering University of Illinois
Supervisor(s)	NAME: Tarek Hegazy Title: Professor, Civil & Environmental Engineering University of Waterloo
Internal Member	NAME: Carl Haas Title: Professor, Civil & Environmental Engineering University of Waterloo
Internal Member	NAME: Adil Al-Mayah Title: Professor, Civil & Environmental Engineering University of Waterloo
Internal-external Member	NAME: Frank Safayeni Title: Professor, Management Science University of waterloo

AUTHOR'S DECLARATION

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Abstract

Delays and cost overruns are common facts in construction projects due to its increasing complexity, the day-to-day dynamic changes, the stricter execution constraints, and the general lack of efficient scheduling tools to support the optimization of construction plans. Currently, many scheduling tools and techniques are available, in addition to a large body of literature that focus on schedule optimization. Such tools and techniques, however, do not adequately represent or incorporate various practical decisions and constraints, nor provide the project manager with the ability to examine the combinations of actions in order to either plan or bring the project back within the constraints.

This research enhances the schedule optimization research by efficiently modeling real-life decisions and constraints, and develops a framework to optimize planning and corrective-action decisions; dynamically before and during construction. The development of the proposed framework starts with a basic model that suits the schedule optimization decisions at the preconstruction stage. This model is then extended to a generic model that accommodates the dynamic schedule optimization needs during construction. The enhancements and extensions are formulated in a generic mathematical formulation to optimize the schedule's decisions at any stage. This formulation integrates a wide range of scheduling options (e.g., linear crashing, activity multimodes, overlapping, and multipath networks), and incorporates the project manager's preferences about the corrective-action decisions' implementation. The formulation also considers a variety of practical constraints (e.g., variable resource availability, correlated modes, and intermediate milestones); and uses a multi-objective optimization to tradeoff among the project time, cost, resources, and permissible schedule changes during construction. Based on the mathematical formulation, the proposed framework was then coded using the advanced

constraint programming tool “IBM ILOG CPLEX Optimization Studio”. To validate the model, multiple experiments on four case studies were used to prove the functionality, practicality, and its better representation of real-life construction challenges. Two of these case studies are taken from the literature to prove the ability of the comprehensive model to achieve better solutions. Construction experts were also consulted at multiple stages of this work to investigate the relevance of the framework.

Introducing the proposed framework as an add-on to standard project management software is expected to change the practitioners’ perception that optimization is a theoretical and complex tool. Therefore, it helps to present optimization as a useful decision support tool for construction scheduling.

Acknowledgements

First and foremost, I thank God (Allah) for giving me the knowledge, strength, and patience throughout my PhD journey. It was an exceptional experience that taught me a lot. I would like to express my sincere appreciation and gratitude to my supervisor, Dr. Tarek Hegazy, for his invaluable guidance and for enriching my research skills over the last five years. I am deeply impressed with his wealth of knowledge, excellence in teaching, and dedication to academic research. It has been a great honour to work with him and to learn from his experience.

I would like to thank my committee members Dr. Khaled El-Rayes, Dr. Carl Haas, Dr. Adil Al-Mayah, and Dr. Frank Safayeni, for their insightful comments.

I extend my appreciation to the Ministry of Higher Education in Egypt (MOHE) for providing funds to support my study in Canada. The very usefulness of the IBM Academic Initiative program is also acknowledged for offering access to the constraint programming optimization software.

Special thanks to my father, may God have mercy on him, and my mother; they are the reason for what I have become today. I also would like to thank my wonderful parents-in-law.

Finally, warm thanks to my beloved husband, Eng. Mohamed Eletr, for his endless endurance and encouragement and for always believing in me, in addition to his experience in managing Oil and Gas projects that he shared with me. Without his support, I would not have been able to reach this point. Thanks from the bottom of my heart for my kids: Mona and Abdelrahman, who have always been patient and understanding about the time I've dedicated to this degree and also for their encouragement. Thanks to my little one, Aser, your coming gave me the power to accomplish this work.

Praise to Allah

Dedication

This thesis is dedicated with love

To my parents

To my husband, Mohamed

To Mona, Abdelrahman, and Aser

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Background

While the construction industry makes up a sizable portion of the Canadian and American economy (approximately 8% of the GDP according to Statistics Canada, 2017; Bureau of Economic Analysis, 2016), cost overruns and schedule delays are common obstacles in construction projects, particularly large-scale and mega projects (Jergeas and Ruwanpura, 2010, Siemiatycki, 2009). According to the statistical reports of The Standish Group International (2015) over the last ten years, on average 68% of construction projects in the US and Canada miss their deadlines and/or budgets by 20% to 100% as shown in Figure 1.1. A trend from reports that continued in the latest survey is how larger projects have a much higher likelihood of failure than smaller ones. In their study among 50,000 projects around the world, Standish Group Survey (2015) reported that in 15% of projects that fails, ineffective and inadequate scheduling and planning are the top reasons of failure. However, the root cause reported behind the remaining 85% of failed projects is that "the traditional method used to plan and schedule projects is fundamentally flawed" which is also referring to scheduling problems.

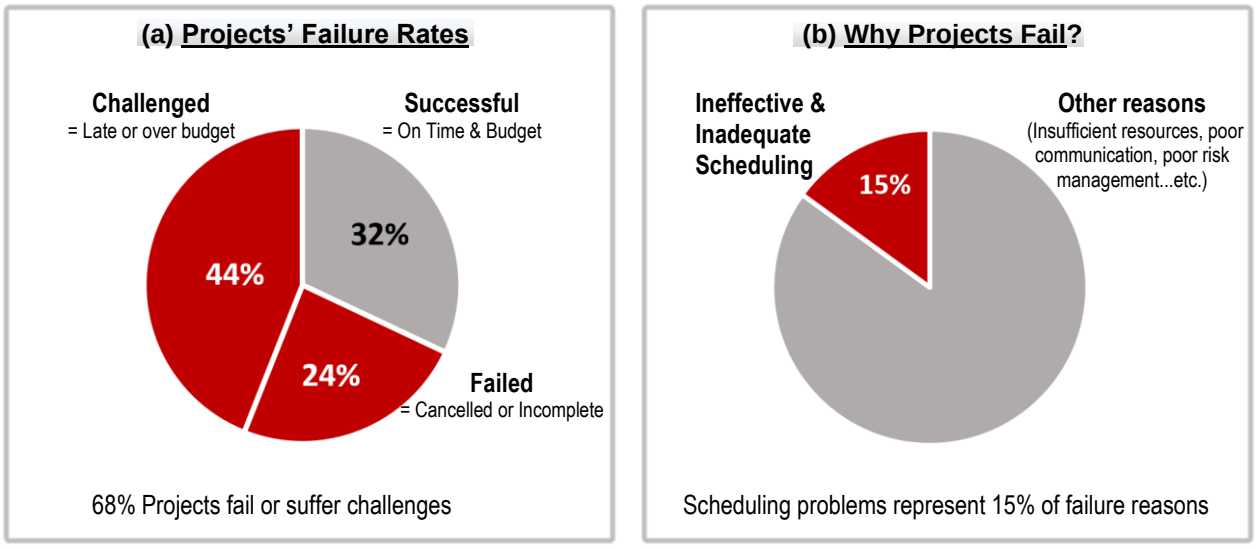


Figure 1.1 Percentage and reasons of construction project failure (Standish Group International 2015)

In 2016, the PMI reported that over 54% of construction projects were delivered late, over budget, or incomplete. The investigation of most of the primary causes reveals scheduling problems. In addition, different studies that are showing that construction industry is facing growing of constraints of tighter budgets, stricter deadlines, and resource shortages (Nasir and Haas, 2013). Further, a set of impractical situations that commonly occur during construction is due to the pitfalls of current scheduling techniques (Galloway 2006).

The above statistics and studies reveal that the construction industry lags behind other industries in efficient planning and scheduling (e.g. time scheduling, resource scheduling, cost scheduling, cash flow management, etc.). As scheduling is a key component of successful construction projects, professional organizations and commercial software continuously provide construction professionals with many practical insights to improve their developed schedules. However, the use of scheduling tools does not extend beyond creating a schedule with a neat appearance to satisfy the contract requirements, and these tools are not used for decisions impacting efficient project execution (Baweja 2006). There is still a significant need of improving the ability of construction firms to optimize their projects' schedules within strict cost, time, and performance guidelines in the current highly competitive and unstable construction business environment (Chen, and Tsai, 2011).

From the above discussion, it is clear that developing practical, flexible, reactive and optimized schedules is very much needed for effective project scheduling and management, thus improving the economics of project delivery, and the construction industry as a whole. Such needed improvements in project management will have a significant economic payoff, not only at the industry level, but also at the national level.

1.2 Research Motivation

Despite the substantial amount of previous and ongoing research related to schedule optimization in construction, there are serious drawbacks that this research will address, which represents the following research motivations.

1.2.1 Drawbacks of the Critical Path Method (CPM)

Current scheduling methods primarily rely on the CPM algorithm, which is based on two unrealistic assumptions: that the project deadline is not restricted and that resources are unlimited. To address these limitations, a number of techniques, such as time-cost trade-off analysis and resource leveling have been introduced since the 1960s, however, they deal with limited options for scheduling techniques. While these techniques can improve CPM scheduling, they can only be applied to a project one after another, rather than simultaneously (Hegazy 2002). Application of such techniques also renders the scheduling process longer and less comprehensible. In addition, most construction projects are exposed to a variety of practical requirements and constraints, such as fast-tracking, alternative construction sequence, and intermediate milestones. It is therefore often difficult to produce a realistic schedule because a solution to one constraint (e.g., limited resources) may interfere with the solution to another (e.g., deadline) because there is a lack of adequate procedures and models for resolving all constraints simultaneously. Therefore, practitioners often blame their inability to meet targets on the available tools because of their inability to: reflect various in-situ constraints relevant to the way the job is built (Thomas, 2001), optimize alternative schedule options to arrive at a cost/time effective schedule (Ahuja and Thiruvengadam 2004), and provide support for continuous changes of projects during construction (Liu and shih, 2009; and Vanhoucke, 2012);

1.2.2 Inadequate Modeling of Real-life Decisions

Almost all commercial scheduling software tools contain a limited set of features (e.g. resource allocation and leveling), and ignore other important decisions, leaving practitioners with insufficient support. These include Time-Cost Tradeoff (TCT) and other decisions such as fast-tracking, multimode activity execution, and optimizing the schedule to meet evolving constraints.

At the academic front, a large, but limited, body of literature exists in construction scheduling to introduce various individual scheduling options that were not addressed in CPM on small size projects. Some of these practical scheduling options include: activity multimode, alternative activity calendars, and fast-tracking. Existing models in the literature, however, do not represent an integrated scheduling model that incorporate these issues collectively. Yet, these models do not consider the dynamic nature of project constraints encountered throughout the execution of real size projects; they are not efficient in optimizing large-scale project networks (Golzarpoor, 2012). Most of these models, in essence, focus on using new methods to solve the optimization problem or refining an already used method to improve the optimization results and processing time, more than presenting new scheduling options.

Accordingly, existing commercial tools and scheduling research do not represent the various options that the project manager can exercise in real projects. Due to the application of these options such as fast-tracking, which becomes an increasing trend for construction, without understanding the long-term impact, contractors are left with unrealistic compressed and over-budget schedules (Jergeas and Ruwanpura, 2010). In fact, due to the lack of such decision support tools, the horizon of the detailed schedule of many contractors is very limited, thus they are being able to perceive the impact of recent execution events and delays (Thomas, 2001).

1.2.3 Need for Dynamic Corrective-action Planning

Along the course of construction, projects experience many changes on a daily basis which can cause significant deviations to project time and cost. Construction, therefore, requires frequent schedule updating and corrective-action to bring the project back on track. Both of the schedule updating and the corrective-action planning are challenges.

Schedule updating is a challenge, because most existing software tools make the unreasonable assumption that the remaining work of ongoing activities will follow the planned progress rates, even if the work so far has proceeded at a much slower rate. Such an assumption underestimates project deviation and results in late corrective actions. Determining the appropriate corrective actions in terms of which activities to expedite, and at what cost, is also a big challenge. In the market, the same traditional commercial software that is used for scheduling planning before construction is used for progress analysis and corrective-planning during construction. This does not take into consideration the lack of support for considering unforeseen changes during construction, experimenting with alternative construction methods, or optimizing the corrective-action plans to achieve a cost-effective schedule. In the academia though, few researchers have introduced very few models for dynamic schedule optimization (Hegazy and Petzold, 2003; and Liu and Shih, 2009). These models exercise very limited set of corrective actions to minimize the total cost while adhering to the contracted deadline, regardless of how these actions change the base plan. Lack of control of the deviations from the base plan is one of the primary reasons of cost overruns (Jergeas and Ruwanpura, 2010). There is a great need for processes or methodology to help bring the construction project back on track and getting stakeholders to accept the needed changes (PM Solutions, 2015).

1.2.4 Scheduling Advances (in other domains), not Utilized in Construction:

Similar to construction, other domains like manufacturing require efficient scheduling to react dynamically to changing events such as risks, change orders, scope change, etc. (Vieira et al., 2003). However, new scheduling techniques have evolved in these domains. For example, online scheduling, real-time scheduling, etc. all continuously optimize their schedules (Sabuncuoglu and Kizilisik, 2003). These techniques address important issues such as minimizing schedule disruption, and maximizing schedule stability as per decision maker preferences. Such techniques motivate the incorporation of project manager's preferences in optimizing the corrective-action decisions, which is not properly addressed in construction.

On the other hand, recent advances in optimization tools and computing technologies are used to solve complicated problems in several scheduling domains (Hentenryck et al., 2002). Recent optimization techniques such as Constraint Programming has opened up new opportunities for optimization of complex and large-scale scheduling problems, however, its application in construction is still very limited (Menesi et al., 2013; Abuwarda and Hegazy 2016 a,b; Abuwarda and Hegazy 2018).

Based on the above discussion, it is clear that there is a great need for efficient scheduling models that enable the project manager to experiment with and decide the proper combination of actions that will keep a project within constrained borders throughout the project's execution life cycle.

1.3 Research Objectives and Scope

The goal of this research is to enhance schedule optimization research by efficient modeling of real-life decisions and constraints, and develop a framework to optimize planning and corrective-action

decisions dynamically before and during construction for construction projects. A comprehensive decision support tool is expected to provide the ability to achieve real cost savings and better management of projects within constraints. To achieve this goal, the research objectives are as follows:

- Investigate the gaps in schedule optimization research, and identify the real-life parameters, decisions, and constraints which need to be incorporated into a generic scheduling model that suit the two distinctive project scheduling phases in construction: "Phase I: preconstruction", and "Phase II: during construction";
- Develop a mathematical framework to optimize decisions before construction Phase-I which requires enhancements of the representation of the project network, activities, and relations, in addition to a set of new practical constraints. The new framework will enable the project manager to exercise a wide range of practical options (e.g., fast-tracking and multipath schedules), within a strong optimization system, and avoid the rigid forms of traditional scheduling tools;
- Extend the framework of Phase-I to optimize decisions during construction Phase-II, considering progress details, corrective-action options, and project manager's preferences. The extended framework formulation becomes a comprehensive schedule optimization system that works at any project stage; and
- Implement the proposed framework into the advanced Constraint Programing (CP) Optimization Language. Accordingly, develop an automated computer prototype, and validate the system using several case studies.

1.4 Research Methodology

The methodology for achieving the above objectives is described below, and illustrated in

1) **Extensive review of CPM drawbacks and existing schedule optimization research:** to identify the limitations that prevent CPM from satisfying the requirements of construction practitioners, and conducting an extensive survey of the schedule optimization studies conducted to amend the CPM flaws and solve the schedule optimization problem. The reviewed literature will include other scheduling domains (e.g. manufacturing) rather than construction. The understanding of schedule optimization efforts and advances in these domains and comparing them with the construction literature is critical.

2) **Identification of real-life decisions at different construction phases:** Based on the literature review and identified research gaps, an investigation of the real-life parameters, decisions, and constraints, which are essential and have been overlooked by current researchers, will be performed before and during construction. Experts in the construction field (from local construction companies) will be consulted along this step.

3) **Development of comprehensive dynamic schedule optimization framework:** the structure of the main framework will combine the framework of schedule optimization of two distinctive phases to suit the distinctions between the required decisions of "Phase I: preconstruction", and "Phase II: during construction phase".

4) **Development of the schedule optimization framework of "Phase I: Preconstruction**": to improve decision making before construction, this framework involves five main enhancements to

the representation of the basic scheduling model options on the network level, activity level, relation level, and project level.

5) **New representation of project network with alternative paths:** the proposed framework will support activity decisions to be made with respect to optional activities. The project network with multiple groups of alternative branches (paths) will be mathematically presented.

6) **New representation of activity execution options:** The proposed framework will introduce "Activity Time-Cost-Resource (TCR) Spectrum" represent the trade-off among time, cost, and resource. The visual representation of the spectrum, as well as, the automated detailed calculations of each branch in the spectrum will be introduced.

7) **Representation of flexible activity relations in the CPM:** To accommodate the fast-tracking within the traditional CPM, new formalization of flexible relation will present a generic logical relationship (hard or soft) of any type (finish-to-start, etc.) between any two activities and indicate the permissible overlapping range between the activities.

8) **Introduction of new constraints and multi-objective decision environment:** A time-dependent renewable resource capacity will be incorporated in the framework in addition to numerous enhancements relevant to constraints on: correlation between activities' modes, multiple intermediate milestones, advanced integration constraints, and rework limits. Also, the formulation of the objective function includes time, cost, and resource objectives separately or simultaneously.

9) **Development of the schedule optimization framework of "Phase II: During construction**": This framework will extend Phase-I framework by considering the baseline schedule decisions,

evolving constraints, and objectives. These extensions will include preparation of a generic list of possible corrective-actions for each activity, and the incorporation of project manager's preferences regarding the potential changes to the baseline schedule due to the corrective-action decisions.

10) **Mechanism for preparing corrective-action alternatives:** To define a comprehensive list of corrective actions for each activity, an automated mechanism is developed to be run each reporting period. The mechanism will use programmed spreadsheets to examine the daily site reports, the information about progress and working conditions to automatically prepare the required inputs for the scheduling optimization model during construction. In addition, the spreadsheet will be used to define the list of potential corrective actions for each activity, including the improvement of working conditions and workers' morale.

11) **Defining satisfaction functions:** Introducing the concept of satisfaction functions to integrate explicitly the individual preferences of the project manager regarding the adherence to the baseline schedule during construction. Three scenarios of corrective-action preferences are presented: short-term corrective-actions, long-term corrective-actions, and the traditional minimum cost scenario. Another dimension of project manager preference will be included to integrate the project manager's preferences of the activity individual flexibility to change their mode during construction. Consideration of the project managers' preferences about the changes of the baseline after starting the real construction has not been addressed in other scheduling research.

12) **Comprehensive mathematical formulation to optimize schedule decisions at any stage:** The formulation will integrate all of the new options (e.g. fast-tracking, and multipath network), new

constraints (e.g. variable resources availability, correlation, and intermediate milestones), and the satisfaction functions.

13) **Coding of the mathematical formulation and computer implementation:** the mathematical formulation of the comprehensive framework will be coded using the advance constraint programming tool "IBM ILOG CPLEX Optimization Studio". Coding in the CP environment will be a critical step to develop the model.

14) **Validation:** Four different case studies were used in order to validate the proposed optimization models and prove their functionality, practicality, and usefulness for better efficiency in developed schedules. For simplicity, the case studies are designed to test some of the enhanced options and decision of the new framework separately, where, collectively all developed options are experimented.

1.5 Thesis Organization

The remainder of this thesis is organized as follows:

Chapter 2: This chapter provides a brief insight into the use of traditional scheduling techniques and their limitation in construction from the practitioner prospective, followed by a comprehensive analysis of the literature extensions of the traditional scheduling problems;

Chapter 3: This chapter introduces the proposed framework of two distinctive components to optimize scheduling decision before and during construction. It focuses on the new representation of the scheduling options, new constraints, and project manager's preferences in the middle of a project;

Chapter 4: This chapter presents the detailed mathematical formulation of the two optimization algorithms. First, the mathematical formulation of the preconstruction model introduces the decision variables, constraints and objective functions. Afterwards, the extended formulation was introduced to facilitate dynamic scheduling during construction, followed by the working procedure and detailed steps to implement the formulation into the IBM ILOG CPLEX Optimization Studio.

Chapter 5: This chapter focuses on the framework validation. Four case studies are presented in order to demonstrate the applicability of the modeling concepts and scheduling algorithms discussed in this work. One case study optimizes the schedule at the early planning stage and immediately before construction. The second and third case study demonstrates the model capability to optimize schedule compression before construction. The fourth case study shows the models effectiveness in optimizing corrective-action planning and recovery plans with significant support to decision making by considering the individual preferences of the project manager. For validation purposes, two of the case studies are from the literature to compare the results, and meetings were arranged with multiple industry experts to examine the practicality and proximity of the model results.

Chapter 6 Finally, Chapter 6 provides a summary of the main results obtained in this work, highlights its contributions, and outlines opportunities for future research on project scheduling.

Chapter 2

Literature Review

2.1 Introduction

This chapter starts with identifying the drawbacks of the current technique that has been primarily used by construction companies for project scheduling: the Critical Path Method (CPM). The rest of this chapter presents a comprehensive review of the schedule optimization models that have been developed to amend the CPM shortcomings. This review is organized as follows: introducing the various representation parameters of the schedule optimization problem, discussion of the existing solution methods for the scheduling problem as combinatorial optimization problems, and presentation of existing literature on extensions of the standard schedule optimization model which make the model more suitable for practical applications. The chapter ends by introducing past research related to progress tracking, schedule updating, and corrective-action planning as important inputs to the scheduling model during construction. The information presented serves as a precursor to establish this research framework as discussed later in Chapter 3.

2.2 Drawbacks of Traditional CPM Scheduling Techniques

The basis of current scheduling and project management tools is the Critical Path Method (CPM) which was established in the 1950's. However, after 50 years of practice, serious drawbacks have been identified both in academic publications and in practitioners' notes (e.g. Suhail and Neale, 1994; Hegazy and Wassef 2001; Hegazy 2005; Street 2000; Lam and Lu, 2006; Lowsley and Linnett, 2006; Menesi, 2010). Primarily, CPM is not designed to respect a combination of deadline and resource limits, nor incorporate any optimization method to determine an economic construction plan (Lu and Lam

2008). Moreover, project networks with multiple activity relationships (e.g. Start-to-Start, and Finish-to-Finish with negative lags) are challenging to analyze using CPM due to their complexity (Lu and Lam 2008). By turn, resolving these primary drawbacks and improving scheduling capabilities has captured the interest of researchers over the years. Early research efforts have used three individual techniques, which have been commonly used by practitioners to this point, as follows:

1- **Using PDM**: To amend CPM incapability to describe activity interdependencies in an adequate way, an enhanced version of CPM, the precedence diagram method (PDM), was developed to allow additional relationships between activities rather than the traditional Finish-to-Start relationship (e.g. Start-to-Start, Finish-to-Finish, and Start- to-Finish) and enables the definition of either "Lead" or "Lag" for each relation.

2- **Time-Cost Trade-off:** To overcome CPM's inability to confine the schedule to a specified duration, Time-Cost Trade-off (TCT) analysis is used. The objective of TCT analysis is to reduce the original CPM duration of a project in order to meet a specific deadline with the minimum cost (Chassiakos and Sakellaropoulos 2005), or to accelerate a project so that time delays can be recovered. The TCT analysis involves reducing the duration of some activities on the critical path of the project at the expense of increasing the project direct cost, starting with the activities of the least crashing cost. These actions have different implications on overall resource usage of the project which might exceed the resource limits, however, these limits are not considered in the traditional TCT analysis

3- **Resource Allocation and Levelling:** CPM assumes that the resources required for activities are unlimited, while in most practical situations, resources are available only in limited

amounts, particularly when resources are used for multiple activities or even for multiple projects (Lu and Li, 2003). Typically, the original CPM is used for scheduling resource-constrained projects by creating resource-driven relationships in addition to the logical relationships (Hajdu, 1996). However, it may result in wrong activity float calculation (Kim and de la Garza 2003; Lu and Li 2003) an incorrect critical path is therefore produced (Menesi 2010). Also, techniques like "Resource Leveling/Allocation" are used to fix over-allocations of resources in CPM schedules by shifting activities further in time (Hegazy, 2002; and Vanhoucke, 2012).

However, techniques like TCT analysis and Resource Levelling are applied separately after performing the traditional CPM analysis in order to deal with deadlines and consider limited resources. Therefore, applying such techniques solve only one constraint, e.g., resource limits, may violate the solution to another, e.g., the deadline. Consequently, applying these techniques does not present a radical resolution to the CPM limitations. Moreover, the most inherent drawback of CPM as a time analysis tool, and its inability to incorporate cost analysis which is a key objective in construction projects. To integrate the two objectives of least time and cost in construction scheduling and consider other constraints of time, resources, and various precedence relations, a huge body of literature has been introduced regarding scheduling optimization models over the past two decades as discussed in detail in Section 2.3. Most of these schedule optimization model consider the formulation of CPM in their mathematical formulation to solve the combinatorial optimization problem.

Despite the extensive existing research on schedule optimization over the past two decades, the model of these researches are not applicable to the market because of their complications. None of the existing

commercial software includes optimization algorithms, while the software developers are still depending on CPM calculations. Over the time, additional developed CPM drawbacks have been identified from a practitioner point of view, and are listed as follows:

- CPM's inaccurate schedule calculations due to the extensive use of leads and lags (Wickwire and Ockman, 2000) and inability to consider multiple activities' calendars (Scavino, 2003);
- CPM does not incorporate mechanisms for activity fast-tacking, activity linear crashing, or even responding to actual progress challenges (Lowsley and Linnett 2006);
- CPM provides little support for cost-effective corrective actions to recover from progress delays and cost overruns (Kuhn 2006), and it lacks ability to incorporate multiple baseline updates (Livengood and Anderson 2006);
- CPM is not responsive to the needs of field personnel and there is too much dependency on specialists to adjust CPM schedules (Liberatore et al. 2000; and Kelleher, 2004); and
- The intrinsic limitations of existing commercial tools for scheduling and planning (e.g. Primavera and MS Project software systems) are partly due to their rigid CPM formulation (Wickwire and Ockman 2000).

These findings agree with the results of a recent survey by Galloway (2006) revealing that CPM is useful in updating activities' data and analyzing progress status, but is not as beneficial in supporting decisions such as corrective actions to recover execution problems. CPM drawbacks are also most evident in the large cost overruns, and project delays due to serious drawbacks with current methods for scheduling (Jergeas and Ruwanpura, 2010; PM Solutions, 2011; Thomas, 2001; Standish Group 2015).

In conclusion, a comprehensive scheduling model that can consider both the early and the developed drawbacks of CPM, and can be incorporated into the commonly used commercial software (e.g. Microsoft Project) is still needed so that all practical aspects become available to the project manager and practitioners.

2.3 Schedule Optimization Problem

The standard scheduling problem assumes that an activity can only be executed, continuously once started, in a single mode which is determined by a fixed deterministic duration and fixed resource requirements. However, huge body of literature on schedule optimization extended the schedule representation by allowing various activity assumptions in their scheduling models. These extensions are not limited to construction domain, they exists in various domains of scheduling such as telecommunication, job shop scheduling, satellite scheduling, and time tables problems. In this work, the author categorizes this schedule optimization body of knowledge according to the main wide "*problem representation assumptions*" as follows:

1) *Activity Assumptions*

Typically, researches adopt more than one of the following assumptions simultaneously:

a. Multiple modes of activity execution: Each activity can have multiple methods of construction. Good surveys of scheduling problems that considers the existence of multiple construction mode for project activities are found in Demeulemeester and Herroelen, 2002; Peteghem and Vanhoucke, 2010; and Menesi et al., 2013

b. Linear crashing: The possibility of using linear activity crashing to reduce activity duration is extensively done in the early literature (e.g., Kelly 1961; Meyer and Shaffer 1963; Hendrickson

and Au 1989; Pagnoni 1990; Fondahl 1961; Fulkerson 1961; Elmeghraby and Salem 1981; Hajdu 1996)

c. Overlapping: Parallel execution between eligible activities (usually referred to as fast tracking) is considered extensively in the domain of construction (e.g. Roemer and Ahmadi 2004; Berthaut et al., 2011; Grèze et al. 2014; Hazini et al. 2014; Khoueiry et al. 2013; Hossain and Chua 2014; Dehghan et al., 2015)

d. The possibility of activity pre-emption: It is also known as allowing activity splitting where it is possible to split some activities into parts without altering their precedence relations and execute these parts discontinuously (e.g. Kaplan, 1988; Demeulemeester and Herroelen, 2002; Ballestin et al., 2009; Vanhoucke and Debels, 2008);

2) Activity duration' characteristics*: deterministic vs. probabilistic*

Some researches ignores any uncertainty in activity duration (e.g. Kelley, 1961; Siemens, 1971; Phillips and Dessouky, 1977; Talbot, 1982; Liang et al., 1995; Hegazy, 1999). On the other hand, other stochastic models have been developed to address time-cost trade-off problems with uncertain activity durations, such as Gutjahr et al. (2000); Feng et al., (2000); and Ke et al., 2009;

3) Project-scale:*: single vs. repetitive or multiple projects*

In practice, often not only one but several dependent projects have to be scheduled simultaneously. This is important if two or more projects share at least one resource and if they may be processed in parallel. Pritsker et al. (1969), Speranza and Vercellis (1993), and Wei-xin et al. (2014)

introduced models which simultaneously considers scheduling the activities of multiple projects, along with due dates and deadlines for the single projects;

4) ***Optimization scope****: during planning vs. during construction*

Generally, scheduling models in the literature of construction management optimize schedules before real site construction, few researchers have introduced models for corrective-action planning during construction (e.g. Herroelen et al., 1998; Chua et al., 2003; Hegazy and Petzold, 2003; Liu and Shih, 2009), however, no effort introduces a model for schedule optimization that suits both stages simultaneously;

5) ***Resource concept:*** *Resource levelling vs. Resource allocation*

Scheduling problems can have fixed project duration and the scheduling optimization targets improving resource profile (e.g. Hegazy, 1999; Leu and Yang, 1999; Akpan, 2000; Son and Mattila, 2004; Elrayes and Jun, 2009). On the other hand, Scheduling problems can have limited resource availability and some flexibility to lengthen the project duration while the scheduling optimization targets minimizing the project span (e.g. Hegazy, 1999; Lee and Kim, 1996; Leu and Yang, 1999; Senouci and Eldin, 2004);

6) ***Optimization objective function:*** *single vs. multi-objectives*

Minimizing cost or time has been the most common objective in the early studies of the single objective schedule optimization problems. However, the time-cost trade-off (TCT) problem can be treated as a multi-objective optimization process to minimize both duration and cost (Liao et al., 2011). Many studies have implemented a multi-objective approach to solve TCT problem such as

Feng et al. (1997); Zheng et al. (2004); El-Rayes and Kandil (2005); Ng and Zhang (2008); Xiong and Kuang (2008).

Considering any combination of these parameters, researchers use abbreviations to refer to the characteristics of their models. This abbreviation typically consists of three parts: the adopted activity assumption, objective function, and resource concept. Table 2.1 shows the most common examples of these abbreviations. The abovementioned classification of scheduling problems aims to show the diversity of scheduling problems, and the multiple dimensions involved. Despite the extensive research efforts in construction, they only cover some of these dimensions individual assumptions. A comprehensive model that integrates all the dimensions is still required in order to increase the practicality of the optimized schedule.

2.4 Solution Methods for Schedule Optimization Problems

This section covers the pros and cons of different methods that are used to solve schedule optimization problems: heuristic methods; traditional mathematical approaches; evolutionary-based techniques; and the advanced mathematical tools (e.g. constraint programming).

2.4.1 Heuristic Methods

Heuristic algorithms are easy to understand algorithms that are based on rules of thumb to find an acceptable near optimum solution for small problems. They do not belong to the category of optimization techniques. The main drawback in using heuristic methods to solve scheduling problems is that they typically provide fast and acceptable, but not optimum, solutions especially for medium size projects (Hegazy, 1999). They also lack mathematical consistency and accuracy and are specific

to certain instances of the problem. Fondahl, 1962; Prager, 1963; Siemens, 1971; and Moselhi, 1993 are some of the studies that have utilized heuristic methods for solving schedule optimization problems.

Table 2.1 Examples of schedule optimization problems' abbreviation in the literature

Acronym	Problem description	Examples
RCPSP	Basic **(RC)** Resource-Constrained **(PSP)** Project Scheduling Problems which require minimization the total project duration, while not exceeding resource limits	Hossain and Chua 2014; Dehghan et al., 2015; Menesi & Hegazy (2014)
MRCPSP	**(M)** Multimode **(RC)** Resource Constraint **(PSP)** Project Scheduling Problem is which require minimization the total project duration, while not exceeding resource limits, with multiple activity modes of execution	Kandil & El-Rayes (2005); Ng & Zhang (2008), Menesi et al. (2013)
PRCPSP	**(P)** Pre-emptive **(RC)** Resource Constrained **(PSP)** Project Scheduling Problem which require minimization the total project duration, while not exceeding resource limits, activities are permitted to have intermittent execution	Xiong & Kuang (2008); Afshar, Ziaraty, Kaveh, & Sharifi (2009)
PMRCPSP	**(PM)** Pre-emptive Multimode **(RC)** Resource Constrained **(PSP)** Project Scheduling Problem which require minimization the total project duration, while not exceeding resource limits, activities are permitted to have intermittent execution within their multiple activity modes of execution	Kaplan, 1988; Ballestin et al., 2009; Vanhoucke and Debels, 2008
RCTCT	**(RC)** Resource Constraint **(TCT)** Time-Cost Trade-off problem which require minimization the total project cost, while not exceeding resource limits or a deadline constraint	Hossain and Chua 2014; Dehghan et al., 2015
MRCTCT	**(M)** Multimode (**RC)** Resource Constraint **(TCT)** Time-Cost Trade-off problem which require minimization the total project cost, while not exceeding resource limits or a deadline constraint, coupled with multiple activity modes of execution	Lacouture et al. (2009).
PRCPSP-FT	**(P)** Pre-emptive (**RC)** resource-constrained **(PSP)** project scheduling problem with **(FT)** fast tracking	Grèze et al. 2014; Hazini et al. 2014;
TCROP	**(TCR)** Time-Cost-Resource Trade-off **(OP)** Optimization Problem which require simultaneous minimization of total project duration and total project cost, and resource variations	Vanhoucke and Debels, 2008; Ashuri and Tavakolan (2013)

2.4.2 Traditional Mathematical Optimization Methods

Mathematical programming implements mathematical representation to solve an optimization problem. Examples of mathematical methods are linear programming, integer programming, dynamic programming, and nonlinear programming. The usage of mathematical programming to solve optimization problems began when George B. Dantzig developed the simplex algorithm in 1947. The simplex algorithm, which is based on linear programming, demonstrated extraordinary computational efficiency and robustness for solving the linear optimization problems. The exceptional power of the simplex method, together with the availability of high-speed digital computers, made linear programming the most fundamental method and the starting point of mathematical optimization. Since then, many additional techniques have been developed, which relax the assumptions of the linear programming and broaden the applications of the mathematical programming approach (Bradley et al. 1977). Mathematical programming methods are typically implemented to solve small size time-cost trade-off problems, and they often fail (or reach local optimum) to solve nondeterministic polynomial-time hardness problems with large number of variables and non-linear objective functions. Examples of researches that have applied mathematical optimization methods for solving schedule optimization problems include Kelley (1961); Meyer and Shaffer (1963); Hendrickson and Au (1989); Pagnoni (1990); and Feng et al. (1997).

2.4.3 Meta-heuristic Methods: Evolutionary Algorithms

The difficulties associated with using heuristic methods and mathematical methods for solving large-scale optimization problems have contributed to the development of alternative solutions. Researchers have proposed evolutionary-based algorithms for searching near optimum solutions to problems. Evolutionary-based Optimization Algorithms (EOAs) are stochastic search methods that mimic the

natural biological evolution of species and/or their social behavior to solve large-scale optimization problems, for which traditional mathematical techniques may fail (Elbeltagi et al., 2005). However, the key drawback of the EOAs is the very long exponential time it takes to arrive at a good solution. Also, solutions are not guaranteed to be optimum. For example, consider a tiny project of only 5 activities, each has 3 optional construction methods and, 10 possible start dates. Then, the search space (i.e., no. of possible combinations of methods and start dates) is $3^5 \times 10^5 = 24{,}300{,}000$, which is very large. However, if the number of activities is doubled to only 10, the search space increases exponentially to $3^{10} \times 10^{10}$, which is 2.4×10^7 time more than the 5-activity project. Examining these combinations to find a single optimum solution is extremely difficult, and the exponential increase in the search space proves the problem is nondeterministic polynomial-time hard (NP-hard) and justifies the long processing time of EOAs models. Various research studies have discussed evolutionary-based optimization methods for solving schedule optimization problems including: Feng, et al., 1997; Elbeltagi et al., 2005; Zheng et al., 2004; Kandil and El-Rayes, 2005; Xiong and Kuang, 2008; Ng and Zhang, 2008; and Afshar et al., 2009.

2.4.4 Advanced Mathematical Technique: Constraint Programming

To produce fast quality solutions for complex and large-scale optimization problems, an advanced mathematical optimization technique, Constraint Programming (CP), has increasingly being used in recent few years. CP derives its speed and power from being a combining technique of logic programming technique and operations research technique. It has been successfully used to solve complex combinatorial problems in a wide variety of domains, with particular advantages in scheduling problems (Brailsford et al. 1999; and Heipcke 1999) including: (1) its efficient solution search mechanism, (2) flexibility to consider variety of constraint types, and (3) convenience of model

formulation. CP also exploits the logical relationships between decision variables and model parameters to determine alternative feasible solutions (Chan and Zeng 2003), which is a unique characteristic of CP. Different researchers reported the recent increasing use of CP to address combinatorial optimization problems with combined resources, time, and cost constraints (e.g. Chan and Hu, 2002; Chan and Zeng, 2003; Gorman and Kanet, 2010). CP has special constraints that perfectly map scheduling constraints, and therefore there are advantages not only in solving the problem but also in modeling it (Hentenryck, 2002).

CP is distinguished by its inference techniques that involve reducing the domain of the variables. CP uses two techniques to find a solution: constructive search and constraint propagation (both initially and during search). The initial constraint propagation removes the possible variable values that will not take part in any solution, thus reducing the search space. The constraint propagation during search, on the other hand, removes all values that violate the constraints. CP then uses a constructive search strategy to guide the search for a solution in the remaining part of the search space. CP continues to search using constructive search and constraint propagation during search until a solution is found. Despite the heart of the CP is handling the Constraint Satisfaction Problem (CSP), However, CP can be used to solve resource constraint scheduling with an objective function involved of minimizing time or cost by implementing its propagation mechanism to the objective function. The objective function in the problem is treated as a constraint, and the upper or lower bounds of the constraint are replaced as soon as a better objective function value is found. While recording the current best solution, the optimization mechanism cuts off the feasible solution space until all of the decision variables have been searched. The current solution is then identified as the optimal solution (Liu and Wang, 2008).

The key advantages of CP are having techniques to reduce the computational effort required and promotes the search ability of solving complex combinatorial problems. CP provides users with different search strategies, including generate and test (GT), backtracking (BT), forward checking (FC), etc. (Apt, 2003). CP also permits to set appropriate ordering of initializing variable values to promote search ability (Liu and Wang, 2008) which contributes greatly to obtain efficient solutions and minimize the processing time. However, CP lacks the support of modeling relaxations that Mathematical Programming includes. Relaxation allows penalizing violations of some constraints, therefore, allowing an easier relaxed problem to solve complicated combinatorial optimization (Hentenryck, 2002).

To use the above methods (i.e. Metaheuristics, Mathematical programing, or Constraint programming) to find the solution of any optimization problem, various commercial optimization packages are available to model and solve optimization problems. A comprehensive review of the available optimization tools in the market for solving only, modeling only, or both of optimization problems is done by Golzarpoor (2012). Examples of the most commonly used optimization package are: Evolver developed by Palisade Corporation to optimize complex problems using GAs; Excel solver developed by Microsoft Corporation to handle simple Linear and nonlinear problems using mathematical optimization; and IBM ILOG CPLEX Optimization Studio, developed by IBM Corporation as a comprehensive platform for mathematical and constraint programming using CPLEX CP Optimizer. Despite the fact that Microsoft Solver and Evolver are widely used in the market because they are easy-to-use optimization tools packed as Add-ins in Microsoft Excel, however, they both inherit the limitations of their solver algorithms; thus they lead to effective solutions only for small to medium size of optimization problems. Lately, there has been an increasing use of The IBM ILOG CPLEX

Optimization Studio for development of optimization models in the academia and the market. IBM ILOG CPLEX Optimization Studio uses the CPLEX Optimizer for mathematical programming, the IBM ILOG CPLEX CP Optimizer for constraint programming, combined with powerful Optimization Programming Language (OPL), and comprehensive Integrated Development Environment (IDE). In this work, IBM ILOG Optimization Studio is used to model and solve the proposed scheduling optimization model using constraint programming technique as justified in Section 4.4.

2.5 Schedule Optimization Research

The classification of the scheduling models in section 2.3 shows the diversity and breadth of the scheduling problems in different scheduling domains. It also highlights the range of possible extensions in scheduling parameters that have been developed in the literature in a response to the scheduling challenges that practitioners have encountered in real word situations. This section focuses on the optimization literature efforts to resolve schedule modeling challenges in construction. It starts by defining the basic scheduling model, followed by the challenges of modeling schedule decisions and the commonly used extensions of the basic model to improve the practicality of the schedules of construction project.

The basic model of Resource Constraint Project Scheduling Problem (RCPSP) assumes that the project consists of activities. These activities are given and they are linked by two kinds of restrictions, namely precedence and resource constraints. For each activity, the duration, the resource requests, and the precedence relations with other activities are given. For each resource, the availability is given. All information on durations, precedence relations, and resource requests and availabilities are assumed to be deterministic and known in advance. Early efforts (e.g. Pritsker et al., 1966) presented mathematical formulation of the standard scheduling problem that is proven to be NP-hard problems even in its

simplest form (Blazewicz et al., 1983; and Garey and Johnson, 1979). Later, as a response to practical challenges that are not covered by this formulation, many researchers have developed more general project scheduling models, often using the standard RCPSP as a starting point as shown in the next subsections. Some of these developments are extensively discussed in the literature such as activity multi-modes, limited resources, resource levelling, and activity splitting. While, on the other hand, some other developments like fast-tracking, multipath scheduling, and dynamic scheduling have been addressed only a few in the literature as discussed in detail in the next subsections.

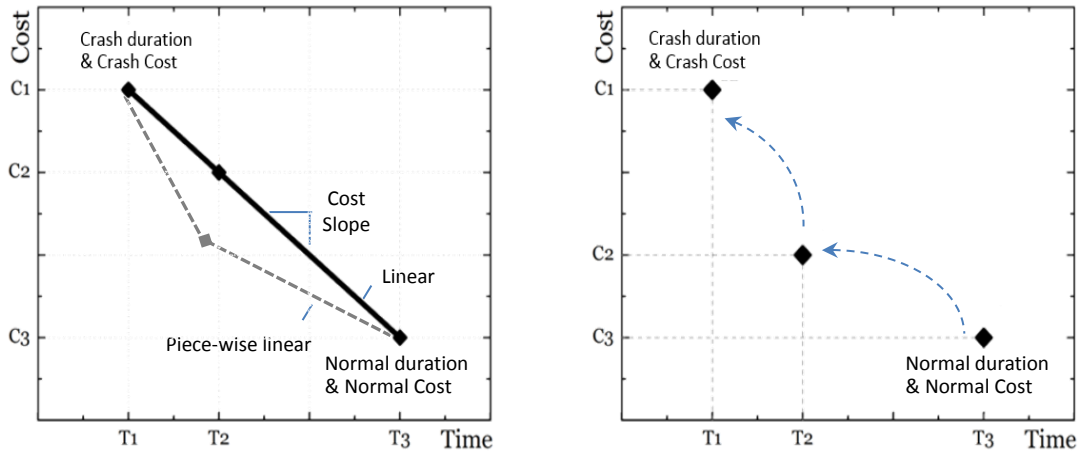
2.5.1 Activity Multiple Execution Options

Typically, the project manager decides the duration for each activity by selecting a time/cost/resource combination for each (Vanhoucke, 2012). However, there is a trade-off between the time, resource, and the direct cost to complete an activity; the less expensive the resources, the larger duration they take to complete an activity (Liao et al., 2011). The planner starts his/her estimation and scheduling process by assuming the least costly option for project activities, called the normal point, which can be crashed to the minimum possible time, called the crash point (Elbeltagi, 2009) at the expense of increasing in project direct cost which comprises: the cost of labor, equipment, and material. To transfer from normal point to crash point, various actions can be used. These actions have different implications on the activity time-cost relationship. The early attempts to incorporate time-cost trade-offs in the basic scheduling model assumed that the activity costs are linear function of the activity durations, and later several other forms of activity time-cost functions have been studied over the years such as concave, convex and discrete. However, linear and discrete relations are the most represented versions in the literature as shown in Figure 2.1, where:

- **Linear Time-Cost relationship (Linear TCT)**: represents a linear continuous relationship between activities time and their respective direct cost. The linear relationship between crash (upper point) and normal points (lower point) implies that any intermediate duration can also be chosen and the slope of the line connecting the two point is called the cost slope of the activity. The slope of this line can be calculated mathematically by using the coordinates of the normal and crash points (i.e. Cost slope = crash cost – normal cost / normal duration – crash duration). This line can contain a number of distinct lines, which is mathematically a piecewise-linear (Vrat and Kriengkrairut, 1986). This kind of time-cost relation allows the problem to be formulated as a linear programming problem. Early studies dealt with linear activity crashing (e.g., Kelly 1961; Meyer and Shaffer 1963; Hendrickson and Au 1989; Pagnoni 1990; Fondahl 1961; Fulkerson 1961; Elmeghraby and Salem 1981; Hajdu 1996). These studies used traditional mathematical tools to find exact optimum solutions for small size networks, but could not converge for large-size problems. Accordingly, heuristic methods (e.g., Siemens, 1971; Hajdu, 1996) were used and were suitable for larger size problems, although do not guarantee optimum solutions. With the surge of computational power and artificial intelligence-based metaheuristic tools (such as genetic algorithms, ant-colony, particle swarm, etc.), various models (e.g., Feng et al., 1997; Li and Love, 1997; Hegazy, 1999) were developed, which require exponential processing time for medium size problems, without guaranteeing optimality; and

- **Discrete Time-Cost relationship (Discrete TCT)**: represents actions like using alternate construction methods that range from cheap and slow to expensive and fast or offering incentive payments to increase the productivity. The discrete relationship between activity

time and cost involves discrete set of execution combinations of time, resource and cost. For each activity, this decision weighs between discrete options that range from a cheap (slow) mode to a fast (expensive) mode. In recent years, modeling activity options in the schedule using multimode Discrete-TCT analysis has been extensively studied, mainly in the context of the planning stage. Comprehensive literature surveys of these efforts can be found in Demeulemeester and Herroelen (2002), Peteghem and Vanhoucke (2010), and Menesi et al. (2013). Recent contributions are made by Ashuri and Tavakolan (2013) using Shuffled Frog-Leaping metaheuristic optimization; Shahriari (2016) using Genetic Algorithms; and Menesi et al. (2013); Menesi and Hegazy (2014); and Abuwarda and Hegazy (2016b) using Constraint Programming (CP).



a) Linear Time-Cost Relationship

b) Discrete Time-Cost Relationship

Figure 2.1 Linear and Discrete of Activity Time-Cost Relationships

From studying the above literature, a conclusion can be drawn that the recent scheduling research is mainly focusing on the Discrete TCT, considering it the most representative case of real projects, however, it is important to note that both types of TCTs are representative based on the available activity execution alternatives to choose between. For example, Discrete TCT represents choosing between alternative technologies to accomplish the activity, e.g. outsourcing the work instead of performing it with in-house resources (Abuwarda and Hegazy, 2016b), using a different type or special kind of software to accelerate the design process, engaging other installation machineries or more productive equipment (Gerk and Qassim, 2008), or sub-contracting a part of the work for faster performance are other examples of substitution (Hazini et al., 2013). While Linear TCT represents manipulating the activity resources through choosing the crew formation among options of normal crew formation and working hours, overtime hours, overmanning, outsourcing, applying multiple-shifts work, working on weekends and holidays, using additional workers or a combination of them (Abuwarda and Hegazy, 2016b, and Hazini et al., 2014). Therefore, enhancements are still required to adequately model the alternative options of performing construction activities that incorporate choosing between different technologies and different crew formation simultaneously, while satisfying the other constraints of resources, the deadline, and limited budget.

2.5.2 Limited-Resource Allocation and Levelling

The basic Scheduling problems with limited resources (RCPSP), typically called "*Resource allocation models*" are widely discussed in the literature. Examples of developed optimization models for a resource allocation problem using various techniques mathematical programming (Talbot 1982, Brucker et al. 1999, Lee and Kim 1996); genetic algorithms (Chan et al. 1996, Hegazy 1999b, Kim and Ellis 2008, Lee and Kim 1996, Leu and Yang 1999, Senouci and Eldin 2004), and particle swarm

optimization (Zhang et al. 2006b). This basic RCPSP features only renewable resources availability of the renewable resources with a predefined constant availability over the entire project duration, however, practitioners in construction projects stated that: 1) there are two more resource categories, nonrenewable and doubly constrained resources; 2) the availability of the renewable resources may vary over the project span (Hartman, 1999; and Vanhoucke, 2002). Moreover, the basic RCPSP generates construction schedules considering resource availability and overlooking resource fluctuations, while these fluctuations could be impractical, inefficient and costly to implement on construction sites. The fine-tuning of schedules in order to reduce significant fluctuations in resource utilization levels over the project duration is known as "*Resource Levelling*" in the literature. A number of resource leveling models and algorithms have also been developed to minimize the level of fluctuations in resource utilization and their negative impact on construction productivity and cost (e.g. Hegazy, 1999; Mattila and Abraham, 1998; Leu and Yang, 1999; Akpan, 2000; Son and Mattila, 2004; Elrayes and Jun, 2009). Some efforts optimized resource leveling, or resource allocation in multi-objective scheduling problems with time, cost, and other objectives such as quality (Hiroyasu and Watanabe, 2000; and El-Rayes and Kandil, 2005). In sum, considering different types of resources availability constraints, along with simultaneously minimizing resource fluctuation, is an important feature to enhance the practicality of construction schedules.

2.5.3 Integrated Efforts for Schedule Planning

While the above research efforts dealt with individual consideration of activity assumptions, few other efforts addressed their integrations as shown in Table 2.2. Roemer and Ahmadi (2004) presented a mathematical model that combined activity linear crashing and overlapping. Their study in the manufacturing domain, however, considered one set of sequential activities, which is not suitable

projects with activities that have multiple successors and predecessors. In other research, Gerk and Qassim (2008) formulated linear crashing, overlapping, activity mode substitution, and resource constraints into a mixed-integer linear model that uses the less common activity-on-arrow representation with FS relations only.

Table 2.2: Recent efforts on integrated schedule optimization efforts

Recent Studies	Scope	Technique	Strategy*				Precedence Relations
			C	O	S	R	
Srour et al. (2013)	Overlapping Design and Construction	System Dependency Matrix		✓			FS **
Khoueiry et al. (2013)	Construction	Mathematical Model		✓			FS
Hossain and Chua (2014)	Overlapping Design and Construction	Simulation and Genetic Algorithms		✓			All
Berthaut et al. (2011)	Overlapping Design and Construction	Mixed-integer linear model		✓		✓	FS
Koyuncu and Erol (2015)	New Product Development	Particle Swarm Optimization		✓		✓	FS
Roemer and Ahmadi (2004)	Manufacturing	Mathematical Algorithm	✓	✓			FS
Hazini et al. (2013)	Construction	Heuristic	✓	✓			All
Hazini et al. (2014)	Construction	Genetic algorithm	✓	✓			All
Ashuri and Tavakolan (2013)	Construction	Shuffled Frog-Leaping			✓	✓	All
Hegazy and Menesi (2013)	Construction	Heuristic			✓	✓	All
Kandil and El-Rayes (2005)	Construction	Genetic Algorithms			✓	✓	All
Menesi et al. (2013)	Construction	Constraint Programming			✓	✓	All
Gerk and Qassim (2008)	General	Mixed-integer linear model	✓	✓	✓	✓	FS
Proposed Model	Construction	Constraint Programming	✓	✓	✓	✓	All

* C = Crashing (linear TCT); O = Overlapping (logic change); S = Substitution (mode change); and R = Resource Constraints. ** FS = Finish-to-Start relationship.

Another effort by Hazini et al. (2013) developed a heuristic schedule acceleration model that integrates activity linear crashing, overlapping, and mode-substitution. The model decides, each day, the cheapest activity to crash or overlap, which suits projects with small number of activities that have linear time-

cost relations. This model was later enhanced in Hazini et al. (2014) using multi-objective evolutionary optimization. Both models, however, do not deal with resource limits.

Based on the above literature, determining an optimal mix of activity crashing, overlapping, and activity modes is still a large challenge, particularly for practical size projects and considering deadline, resource limits, and activity relationship constraints, simultaneously.

2.5.4 Handling Large-Scale Scheduling Problems

One of the main challenges in scheduling projects within the current constraints, is that the optimization problems are known to be NP-hard (Lee et al. 2010; Konak et al. 2006), which are very difficult to solve, particularly for large-scale problems. For large projects, the enumeration of alternative project plans is computationally hard, particularly because the number of alternatives grows exponentially with the increase in the number of activities of the project. For example, the number of possible alternative project plans for a project consisting of 180 activities, each with only 3 modes of construction can reach 3^{180} ($7.6x10^{85}$) (Kandil and El- Rayes, 2005). The number of possible alternatives increases $3.5x10^{9}$ times ($2.7x10^{95}$), if the number of activities in the project network rises only to 200 activities (Golzarpoor, 2012).

Most scheduling efforts in construction discuss small scale examples of 10 or 20 activities, and the few that have handled medium-size problems (i.e., several hundred activities) took an unreasonably large amount of time to provide a solution. Kandil and El-Rayes (2005), for example, have reported GA processing time of 55 hours for a case study of 360 activities, which was reduced to 9.3 hours using a system of parallel computing with 50 processors. In contrast, a heuristic model, which has been proposed by Hegazy and Menesi (2012) for resolving both resources and deadline constraints simultaneously, took 32 minutes to solve a case study of 360 activities, which is much faster than GAs,

for the same size problem. However, even this heuristic method is expected to take many hours to solve problems with more than 360 activities, making schedule optimization a difficult objective to achieve for large-scale projects. One of the latest efforts to handle large-scale problems of Menesi et al. (2013) is the proposed constraint programming approach to solve the Resource Constrained Scheduling Time Cost Tradeoff problem to resolve both time and resource constraints simultaneously for large-scale projects of up to 2,000 activities and proved the suitability of CP models to handle large-scale multi-mode resource levelling problems, however, their proposed model does not represent other practical scheduling features such as alternative path, and fast-tracking.

2.5.5 Fast-tracking and Schedule Crashing

The standard project scheduling problems assumes a fixed/hard logical relationship between project activities, however, in real projects there are some activities have soft logic in their relationships with other activities. These discretionary dependencies can be considered for modification when overlapping techniques are required (PMBOK, 2017). Activity overlapping (often referred to as fast-tracking) exploits the soft dependency relations between eligible activities and changes their work sequence from being in-series to being partially parallel, to save time as shown in Figure 2.2 (Abuwarda and Hegazy, 2016b). One example is starting to run electrical conduits on a slab when the rebar work is halfway done. Overlapping decisions, however, might involve risks which lead to possible rework and resource over-allocation in case both activities use the same type of resource.

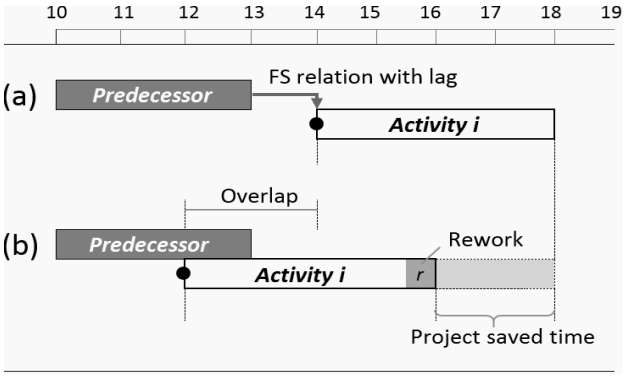


Figure 2.2 Two activities in series (a) can be overlapped (b) to save time

Therefore, although overlapping does not result in an increase in the direct cost, the time and cost of rework need to be considered. To address the time-cost trade-off of overlapping decision in scheduling problems, a large body of literature, mostly found in the manufacturing domain does exist. Sample efforts such as Krishnan et al. (1997); Eppinger 1997; Roemer and Ahmadi 2004; and Koyuncu and Erol (2015) examined the sensitivity of various tasks to overlapping and rework evolution. Good comparisons among overlapping research efforts can be found in Grèze et al. (2014) and Dehghan et al. (2015). In the construction industry, while Pena-Mora and Li (2001) developed a generic framework for fast-tracking of design-design, design-construction, and construction-construction activities, Khoueiry et al. (2013) optimized fast-tracking of design-construction activities; however, studying overlapping of design-stage activities is more common (e.g. Bogus et al. 2006 and Srour et al, 2013). In all efforts, overlapping is assumed to expedite the project at the expense of potential rework, however, they adopt different assumptions in assigning this amount of rework. Many studies in the literature (e.g. Roemer and Ahmadi 2004; Berthaut et al., 2011; Grèze et al. 2014; Hazini et al. 2014; Khoueiry et al. 2013; Hossain and Chua 2014; Dehghan et al., 2015) assign rework as an extension to

downstream activities' durations. They estimate rework time and cost as a linear function of the overlapping time. Other interesting efforts by Berthaut et al. (2011) and Grèze et al. (2014) also assign the rework to the downstream activity by defining overlapping options as discrete modes with a specific amount of rework time and cost, then determine the best overlapping modes using linear optimization. Rather than assigning rework only to the downstream activity, Gerk and Qassim (2008) extend the overlap period of both downstream and upstream activities by a factor that is linearly proportional to the degree of overlapping.

Yet, schedule exercises, that incorporating crashing and overlapping in construction projects, are not well structured. Hence, industry practitioners prefer to use crashing rather than overlapping, particularly with no detailed analysis to avoid raising risks (Hazini, 2012). Moreover, in a large project, expert practitioners are not able to simultaneously consider the dynamics of critical path changes, and all the activities' acceleration options, while satisfying the many constraints related to short-term and long-term milestones, the deadline, and resource limits. Therefore, a systematic process is needed to produce least-cost schedules that account for all the implications for the critical path(s), resource use, rework, project duration, and cost (Abuwarda and Hegazy, 2018).

Another dimension to enhance the basic scheduling model addressed in the literature is considering multipath networks. The basic resource-constrained project scheduling model assumes that the activities to be implemented -and hence the project structure- are given. However, real construction projects cases require decisions to be made with respect to optional activities. For example, during the early planning stage (e.g., the bidding stage or the early stage of project design), the planner focuses on work packaging considering different options (e.g., cast-in-situ concrete versus prefabricated elements as shown in Figure 2.3). Each option has its set of unique activities and sequence of work. The cast-in-

situ option, for example, mandates activities for formwork erection and steel fixing, which are not needed for prefabricated elements. Precedence relationships that lead to alternative path in the network are depicted by broken arrows in Figure 2.3. These relationships are only imposed if their path is selected. In terms of network scheduling, these options represent alternative branches with different relations.

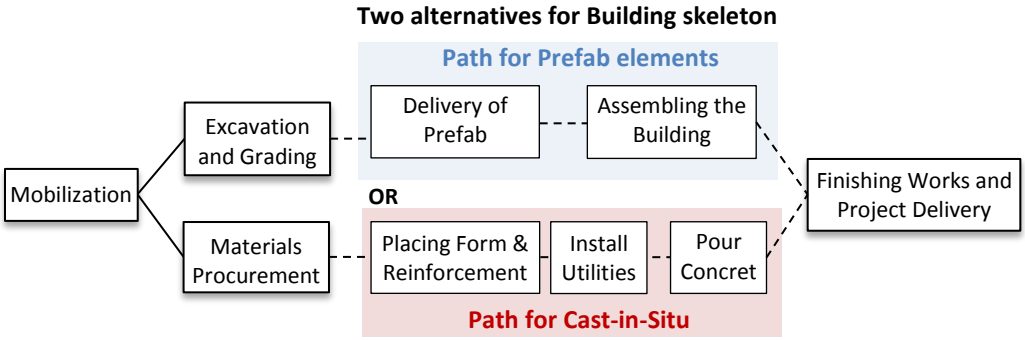


Figure 2.3 Example of two alternative paths of building construction

Multipath scheduling has been researched in different domains such as computer science, job-shop scheduling, and telecommunications (Cherian and Gopalakrishnan Nair 2011; Tsamardinos et al. 2003; Focacci et al. 2000)). Also, the interesting research by Gasparini and Qassim (2003) utilized a superstructure project consisting of several optional interdependent activities and introduced a mixed-integer linear programming model to obtain the optimum set-up, however, this work only considered finish-to-start relationship between activities. More recently, Bartak and Cepek (2007) proposed a network model with parallel alternatives and showed that deciding the final network is nondeterministic polynomial-time hard (NP-hard) and very difficult to solve. In construction, very little research efforts offer alternative logic execution in scheduling problems. Among these very few efforts, Hegazy and

Ersahin (2001) used contingent activity groups in their simple scheduling model to consider alternative sets of activities, but did not change the logical relations between the activities.

While this feature is an important aspect to consider in practical scheduling, the existing efforts do not represent alternative paths adequately or integrate it with other important features such as multimode activities and fast-tracking (Abuwarda and Hegazy 2016b, and 2018) .

2.5.6 Dynamic Optimization during Construction

The statistics regarding troubled projects during its execution are sobering. According to the survey done by Project Management Solutions (2011), companies on average manage $200 million in projects each year and more than a third of their projects, $74 million worth, are at risk of failing and in jeopardy if nothing is done to recover these troubled projects. A process or methodology to optimize decisions is needed in construction projects to recover delays (PM Solutions, 2011). To overcome these challenges, a systematic process accepted by all project stakeholders is necessary to guide the corrective-action planning for project recovery. Corrective-action plans require modifying the initial schedule by rearranging activities and resources in response to project delays. In construction, corrective-action results in a recovery plan, if no extension is permitted to the end of the project, or a revised schedule if there is a possibility to extend up the project end. As the schedule is a part of the project contract, both of revised schedule and the recovery plan requires the consultant approval before being put-into-effect (PMI, 2014). Optimizing decisions when determining which tasks will be rescheduled and in which corrective-action method to use are very important for practitioners in real construction projects when producing their new schedules.

In the literature, extensive research has been carried out in the manufacturing industry to regenerate job schedules to recover order delays, for example, Vieira et al., 2003, and ElMekkawy and ElMaraghy, 1981. To solve their dynamic scheduling problem, Brucker et al. (1999) have defined branch-and-bound methods, Kelleher and Cavichiollo (2001) have used a constraint-based approach, while for small size projects, Yu et al. (2006) have applied an immune heuristic algorithm. Some of these researches used limited time horizon to optimize the dynamic process of production, for example, Ghoniem (2002) in manufacturing, Dishan et al. (2013) in earth-observing satellite planning, and Chong (2012) in plant operation under partial shutdown. Likewise, ElMekkawy and ElMaraghy (2003) have used a heuristic routine to reschedule some jobs rather than all jobs in their corrective-action planning algorithm.

In construction, few researchers have introduced models to optimize corrective-action decisions during construction to recover delays (e.g. Herroelen et al., 1998; and Chua et al., 2003). Hegazy and Petzold (2003) have proposed a genetic-algorithm-based scheduling model to optimize corrective actions during construction, taking into consideration time, cost, and resource constraints, concurrently. Based on concepts associated with manufacturing, Liu and Shih (2009) have developed a corrective-action planning optimization model using Constraint Programming (CP) techniques. These efforts start scheduling from scratch and consider a corrective-action problem as a new scheduling problem irrespective of the initial schedule, however, adhering to the initial schedule, maintaining the ability to recover from delays, speedy recovery from deviations are primary project goals to avoid contractual conflicts.

In conclusion, construction projects are in need of a corrective-action planning framework that is able to reflect and react to the occurrence of unexpected events that make the current schedule non-feasible. Then determining the optimum recipe of corrective-action planning options that will be used to produce a new efficient schedules while adhering to the initial schedule, and respecting the continuing and

evolving project constraints. This recipe has to include wide range of alternative options such as switching some activities to faster construction methods, adding additional resources for critical activities, applying certain strategies to improve workers' morale, and fast-tracking of eligible activities.

To produce a comprehensive model for scheduling optimization that can work during construction, three other components are required to prepare the schedule for optimization: progress tracking and data collection, schedule updating, and corrective-action planning. The following subsection introduce quick reviewing of the available research and existing challenges associated with schedule updating and project progress tracking.

2.5.6.1 Data Collection and Progress Tracking

To select appropriate corrective actions during construction, timely and accurate data collection and documentation of all site events becomes essential (Wang et al. 2007, and Ahsan et al. 2009). However, the documentation of as-built information is mainly a manual process that is time-consuming and error-prone (Trupp, 2004; and Navon, 2007), thus contributing to misunderstandings, incorrect assessment of project performance, and lack of early warnings. Typical paper-based forms or text-based mobile devices for progress data collection from diverse site personnel are deemed to be slow and error-prone. Using these methods, a single instance of progress tracking and schedule updating takes at least two weeks, which is a long time before problems are detected or resolved. McCullouch (1997) reported that, on average, 30–50% of field supervisor personnel's time is spent recording and analyzing site data. As a result, construction projects do not meet their objectives and encounter cost and schedule overruns. The longer it takes to identify discrepancies, the more complex and costly the corrective actions will be. Nowadays and with modern communication technologies, handheld computers and mobile phones

are used to offer interactive project management. These options are gaining popularity in the construction field and have been identified as important IT support for construction sites as they can enhance data collection and progress tracking (Jaselskis et al., 2010). The main types of mobile computing hardware available at construction sites are personal digital assistants (PDAs), handheld computers, pen tablet/touch PCs, and rugged notebook PCs. PDAs offer superior mobility because they are small allow hands-free use and can be integrated with other technologies, such as digital cameras, Global Positioning System (GPS), barcodes, and Radio-frequency identification (RFID). In this regard, an interesting effort to automate data collection by Abdel-Monem and Hegazy (2013) has presented a low-cost framework, utilizing Email and Interactive-Voice-Response (IVR) technologies. The combination of Email and IVR proved to have great potential to minimize the time and cost associated with site data collection by asking relevant and dynamic questions related to each activity. Using structured email also facilitates collecting of complete site data. The developed framework has been applied as an add-on program on existing commercial scheduling software (Microsoft Project), so that the activities themselves call for progress, receive data, and update the schedule by documenting the collected information directly on the daily segments of each activity. Such a system supported the hypothesis of this research that the data related to worker satisfaction levels and morale can be automatically collected is ready to within the proposed framework for corrective action optimization proposed in this study.

2.5.6.2 Schedule Updating

Schedule updating is typically performed to monitor and control project progress. The following tasks are essential in schedule updating: (1) use the progress tracking data to recognizing the actual progress; (2) compare the initial schedule with actual project progress; (3) identify all delayed activities; and (4)

forecast and modify projected work progress based on actual progress (Vieira et al., 2003). In some contexts, schedule updating involves all scheduled inspections and even includes recovery planning; in this case, the task of schedule updating is primarily extended to determine an applicable corrective-action planning policy to the initial schedule by rearranging activities and resources in response to project delays.

Typically, schedule updating forecast the remaining project duration and cost based on one of two assumptions relevant to how the project manager expects the work to continue from now on. The two cases are:

- Case-1: Past actual performance is not a good predictor of future performance. In this case, all remaining work is expected to follow the planned speed of the activity's current methods; or
- Case-2: Past actual performance is a good predictor for the remaining work. In this case, the remaining work will follow the actual rate of progress.

At the schedule updating level, on the other hand, existing scheduling software computes the remaining duration of each activity using the first assumption (Case-1), which results in underestimation of project deviations. To a much lesser extent, a few software systems include hidden features to allow the user to apply the second assumption, but only to all ongoing activities, not to specific activities. Such an assumption also results in the overestimation of project deviations. As such, all software systems make the unreasonable assumption that the remaining work will either follow the plan or will follow the actual so far. In reality, however, neither may turn out to be true. A crew that has exhibited slow past performance cannot be expected to suddenly follow the plan in the remaining days, particularly if morale is low. As such, this work will examine a combination of the two assumptions, as a function of the work progress and worker's morale. These calculations of the remaining work and costs are

essential inputs to optimize the corrective-action decisions during construction, therefore, adapting unrealistic assumptions could mislead the project outputs.

2.5.6.3 Corrective-action planning

Taking corrective action is a very critical a skill set for delivering successful project outcomes. A successful project manager must have the skills and abilities to tackle significant project problems, and this requires a willingness to lead the team through the unknown. However, leading the team through the unknown becomes significantly easier with the right tools and processes (Kastner, 2013). Despite the importance of corrective action planning in construction, a small number of studies present tools to optimize corrective action selection. In one example, Russell and Fayek (1994) presented a framework to automatically suggest the potential corrective action based on the daily site records, however, their framework selection depended only on the reason for difficulties that the activities are experiencing and does not incorporate any cost analysis. Veronika et al. (2006) propose a list of recommended corrective actions by observing the risk level of material cost change acquired from construction experts, nevertheless, their actions are more preventive than corrective. Vanhoucke (2012) conducted a focus meeting with the project managers of eight prominent construction companies in Belgium to explore the various corrective actions have been taken in their project. The results introduce three different classes of corrective actions: 1) reducing activity durations, known in as activity crashing; 2) parallel execution of precedence related activities, known in literature as fast tracking; and 3) re-baselining parts of the original baseline schedule in case of significant changes in scope.

Boosting workers' morale as a corrective action: Morale refers to the positive feelings of an employee towards his work, coworkers, employer, and the company. The efforts in literature to study employee morale have generally been directed toward fostering group rapport in manufacturing or

merchandising industries. In construction, few efforts studied the workers' morale, e.g. Borcherding (1981) effort, to investigate the causes of lost man-hours and its relationship to morale. Markos (2010) has also examined worker engagement as a key to improving performance and Zia (2011) studied the effects of organizational team building on morale and job retention. Goldenhar (2010) modeled the relationship between job stressors and the injury potential and performance deviation of construction laborers. Morale has been looked at by some managers as independent from performance and needs to be addressed and improved over time (Fortrock Construction, 2013). A manager can push for high productivity by using scientific management, time studies, and close supervision, which may cause high production and low morale. The opposite can occur as well since productivity and morale are believed to be independent, a manager has to work on them both to improve overall performance (Rao, 2010). There is a general interest within the construction industry to ensure high morale among workers, since construction is by nature a stressful environment with many potential hazards (Hurst, 2011; and United Construction, 2014). Nasir et al. (2013) developed an index of best productivity practices and one of the important elements that considered is the presence of incentive programs and social activity. In conclusion, boosting worker's morale on construction sites to recover project delays is an intuitive and common practice by project managers. Therefore, it is worthy to add the implementation of worker's morale to the list of potential corrective actions to optimize scheduling decisions during construction.

2.6 Summary

This chapter reviews some commonly used extensions to the basic resource-constrained project scheduling techniques have been briefly discussed. The review revealed the wide range of possible extensions to the basic resource-constrained project scheduling problems that had been developed as a

response to the needs from industry. However, this review shows that these improvements are far from being comprehensive, complete, and representable model of real-life project scheduling. To the author knowledge, a generalized scheduling model suits different project stages and real-life projects size, let alone, Constraint programming model to optimize scheduling during construction.

Based on the literature review represented in Chapter 2, there is a critical need to a comprehensive schedule optimization model that suits the dynamic nature of scheduling in construction projects, starts before the actual construction to issue the initial schedule, and continues during construction to produce the corrective-action plans.

Chapter 3

Proposed Dynamic Schedule Optimization Framework

3.1 Introduction

This chapter starts with a brief description of the two main components of the proposed dynamic schedule optimization framework: "Pre-construction component", and "During construction component" which is considered an extended version of the first component to accommodate real construction needs. Then, this chapter presents the enhanced representation of the scheduling options, constraints, and objectives. Later, this chapter introduces the proposed extensions to support corrective-actions' decisions during the execution of projects.

In this work, there are also some observations on this research study based on the personal experience of the author as she worked as project engineer for four years in Egypt. Other important input regarding this research are provided by professionals with experience on large-scale oil and gas projects. Meetings were arranged to explain the research goals and objectives. These interviews were important to obtain experts' opinion on the best practices and preferences; acquiring and confirming the assumptions used in developing various algorithms; and performing validation of the research results.

3.2 Components of the Proposed Framework

Scheduling is not a static process, it is a continuous dynamic process aimed at providing a plan to achieve the project target within the different existing limits and constraints, and while adapting to the continuous stream of changes along the project execution. This study proposes a comprehensive dynamic scheduling framework with two phases: "Phase I: Preconstruction" to optimize scheduling

decisions before real site construction starts and "Phase II: During Construction" to optimize corrective-action decisions during construction.

As shown in Figure 3.1, there are various dimensions related to the two distinctive phases which mandate various enhancements in the scheduling model to suit each of these phases. Figure 3.1 summarizes, for each phase, the typical decisions involved in the scheduling process, the available inputs and applied constraints, and the needed enhancements and extensions respectively. These aspects are explained for the two phases in the following sections as mentioned in Figure 3.1 as follows: Section 3.3 describes the nature of required decisions, challenges at the different stages of construction, considering the required inputs for each stage and practical pragmatic constraints. Based on the discussion in section 3.3, Section 3.4 introduces the proposed enhancements of the classical schedule optimization model to improve the decision making in Phase-I. While, Section 3.5 presents the proposed extensions of the enhanced model to incorporate the increasing challenges of reach optimum decisions during construction to suit decisions during Phase-II. Later, in Chapter 4, the mathematical formulation of the model considering these extensions and enhancements models will be demonstrated as a step towards producing the comprehensive CP model.

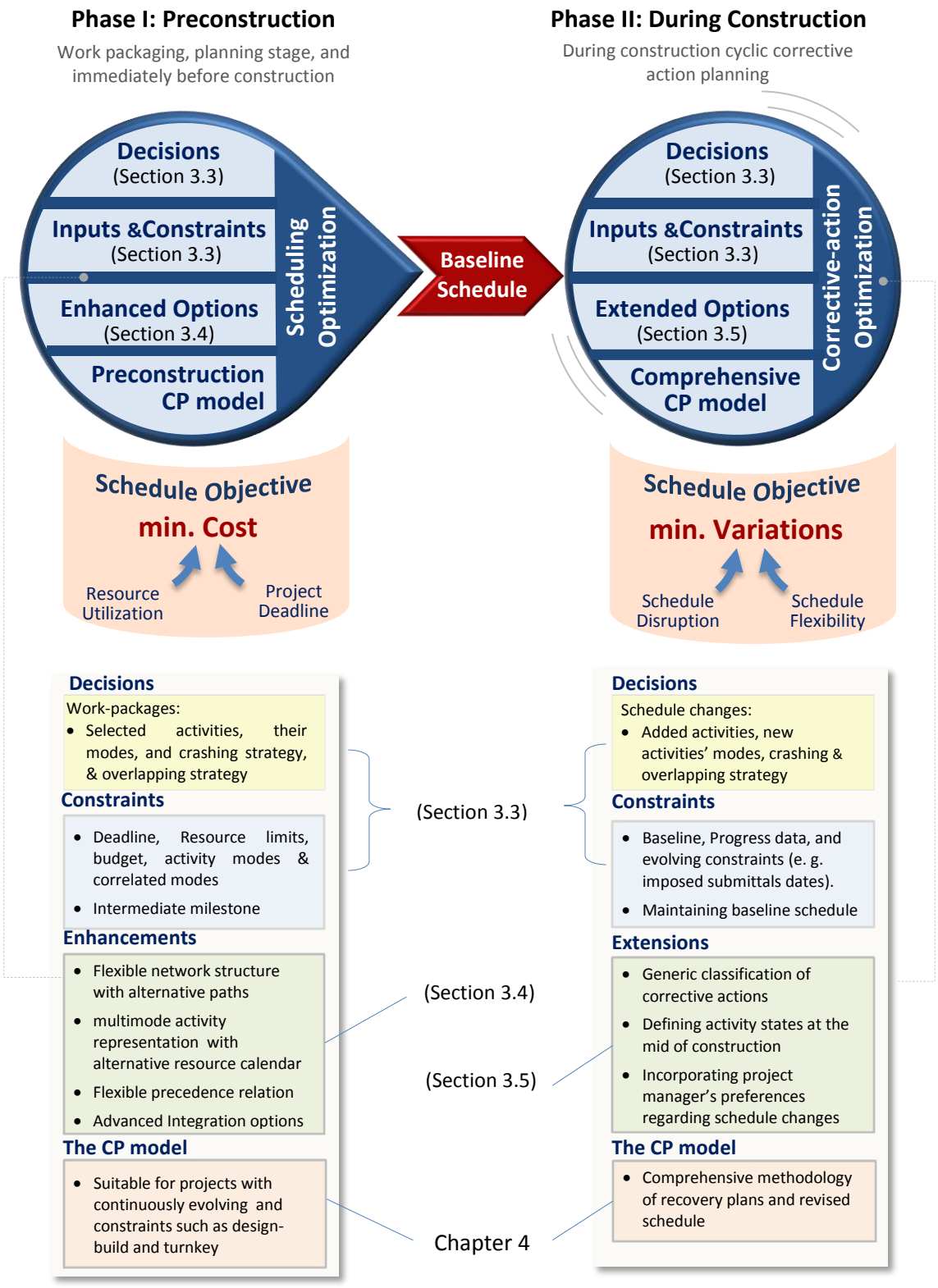


Figure 3.1 Components of the proposed framework

The framework starts at the beginning of the project when the specific project data related to deadline, budget, penalty, incentive, and resource limits are determined, as well as the data pertaining to activities' optional methods of construction. At this point, Phase-I optimization model is run to obtain the efficient project baseline, which is the optimum combination of the activities' methods of construction. After the construction starts, and at the end of each reporting period, the project status is evaluated to determine the need for preparing corrective-action plan. Then, Phase-II optimization framework starts by updating the current schedule, and preparing the potential list of corrective actions of project activities. Then Phase-II optimization model is run with alternative objective functions to yield to a set of the optimum corrective-action plan. This plan defines the new set of remaining activities, their modes of construction, and their start times. This entire process in practice can enable better decision making for project managers. The proposed framework with its two phases requires expertise to provide various options such as the possible alternative paths, the possible alternative modes of construction, possible activity overlapping, and the associated implications of these options on costs and resources.

3.3 Scheduling Decisions and Constraints at Different Phases

Typically, at different stages of projects, planners face various challenges that make it difficult to meet the deadline while satisfying other time, cost, and resource constraints. These challenges are exacerbated in the case of projects with evolving requirements, and constraints such as design-build and turnkey projects, where one company is involved in both the design and the construction of the project (Abuwarda and Hegazy, 2016b). As the design progresses, some changes and adjustments to the constraints become necessary, which may mandate scope changes and/or the use of alternative

construction methods. Overall, the pace by which project constraints change depends on various factors including the project delivery method (design-build, turnkey, etc.), the levels of project complexity and risk, and how early the construction experts are involved in the project. In general, however, there is a distinguishable difference between decisions of planning and scheduling needed during the early planning and work packaging phase (Phase-I in Figure 3.1 left side), and decisions of corrective-actions needed during real construction (Phase-II in Figure 3.1 right side).

During the early planning stage (e.g., bidding stage or the early stage of project design) the planner's focus is more on work packaging considering different options (e.g., cast-in-situ concrete versus prefabricated elements). Each option has a different set of unique activities and sequence of work. The cast-in-situ option, for example, mandates activities for formwork erection and steel fixing, which are not needed for prefabricated elements. In terms of network scheduling, these options represent alternative branches with different activities and relations. On the other hand, each activity has more than one mode of execution with different resource requirements and costs. The planner, in this case, can greatly benefit from support for decision to select the best path and for the selected-path activities, select the specific mode of construction that satisfies the early state of knowledge about the project deadline, milestones, and resource constraints as summarized in the left side of Figure 3.1. Later, as the project is committed to specific activities with specific construction methods (modes), immediately before starting construction, the planner's focus shifts to detailed crew-level scheduling. At this stage, the resources' availabilities, the milestones, and the deadline get refined and need to be accommodated as revised constraints on the scheduling process. In this phase, the planner needs to use specific techniques to fine-tune the adopted schedule to meet the revised constraints/conditions. These techniques, in terms of scheduling options, can be translated into three different classes: deciding on a

crashing strategy for the selected mode (e.g., overtime or overmanning), switching to another mode, and/or deciding overlapping strategy of qualified pair of activities (known for practitioners as fast-tracking) (Abuwarda and Hegazy, 2018). Considering the standard model for resource-constrained project scheduling in the literature, many of practical enhancements are required to develop a schedule optimization model that can support the challenging decisions of the preconstruction phase (Phase-I) as established in Section 3.4.

Once a project has begun, the schedule becomes essential to the successful coordination of day-to-day activities and acts as a baseline for measuring progress. When accurate site events are recorded and entered into the schedule, CPM analysis can help project managers anticipate what might occur in the future by the difference between actual and planned progress (i.e, schedule time and cost variances). Based on the negative schedule variances, if during the project execution the estimated budget or schedule will not be met, project will require appropriate corrective actions in order to recover delays as a recovery plan or revised schedule. Revised schedules or recovery schedule usually involves the submittal of a revised CPM schedule and a written plan. This requires deciding on the appropriate corrective actions that often includes switching some activities to faster construction methods, adding additional resources for critical activities, applying strategies to improve workers' morale, and fast-tracking of eligible activities (Figure 3.1 right side). Despite the fact that the required decisions during construction have the same options of the scheduling decisions that occurred before construction, the nature of the decision becomes more complex because it has a previous experience; the baseline schedule upon which the project contact is signed and subcontracting, outsourcing, and dealing with suppliers' practices are complete. Significant cost overruns is highly correlated to the deviations from the base plan during construction especially for mega projects. Also, construction industry lacks a

process or methodology to help developing a new schedule that matches up the original schedule and convincing project stakeholders to accept the necessary changes (PM Solutions, 2015) reported that the main obstacles to recover project delay are lack of are lack of a process or methodology to help developing a new schedule that matches up the original schedule and getting stakeholders to accept the needed changes. Overall, the goal when making scheduling decisions during construction is to maintain the initial schedule in order to avoid contract conflicts, however this can be exceptionally challenging. This work addressed these challenges through numerous proposed extensions the preconstruction optimization model of "Phase-II" as shown in Section 3.5.

3.4 New Schedule Optimization Parameters

To develop the preconstruction enhanced model that enables all schedule fine-tuning techniques that combine to respond to variety of constraints related to deadline, budget, milestones, and multiple resource limits, etc. simultaneously, a new mathematical representation has been developed. As shown in Figure 3.2, the new representation allows for five main enhancements to the basic scheduling model on four different levels as follows: (1) at the network level, defining project networks with multiple groups of alternative sub-paths (branches); (2) at the activity level, representing alternative linear resource assignments through the optional activity multi-modes; (3) at the relationship level, defining flexible (soft) activity relationships that allows overlapping between activities; and at the project level (4) allowing enhanced constraints such as generalized resource limits, activity-modes' correlation, advanced integration constraints, and Intermediate Milestone; and (5) allowing single and multiple alternative optimization objectives. These five enhancements are important for both the preconstruction and during-construction models, as explained in the next subsections (3.4.1 to Section 3.4.5).

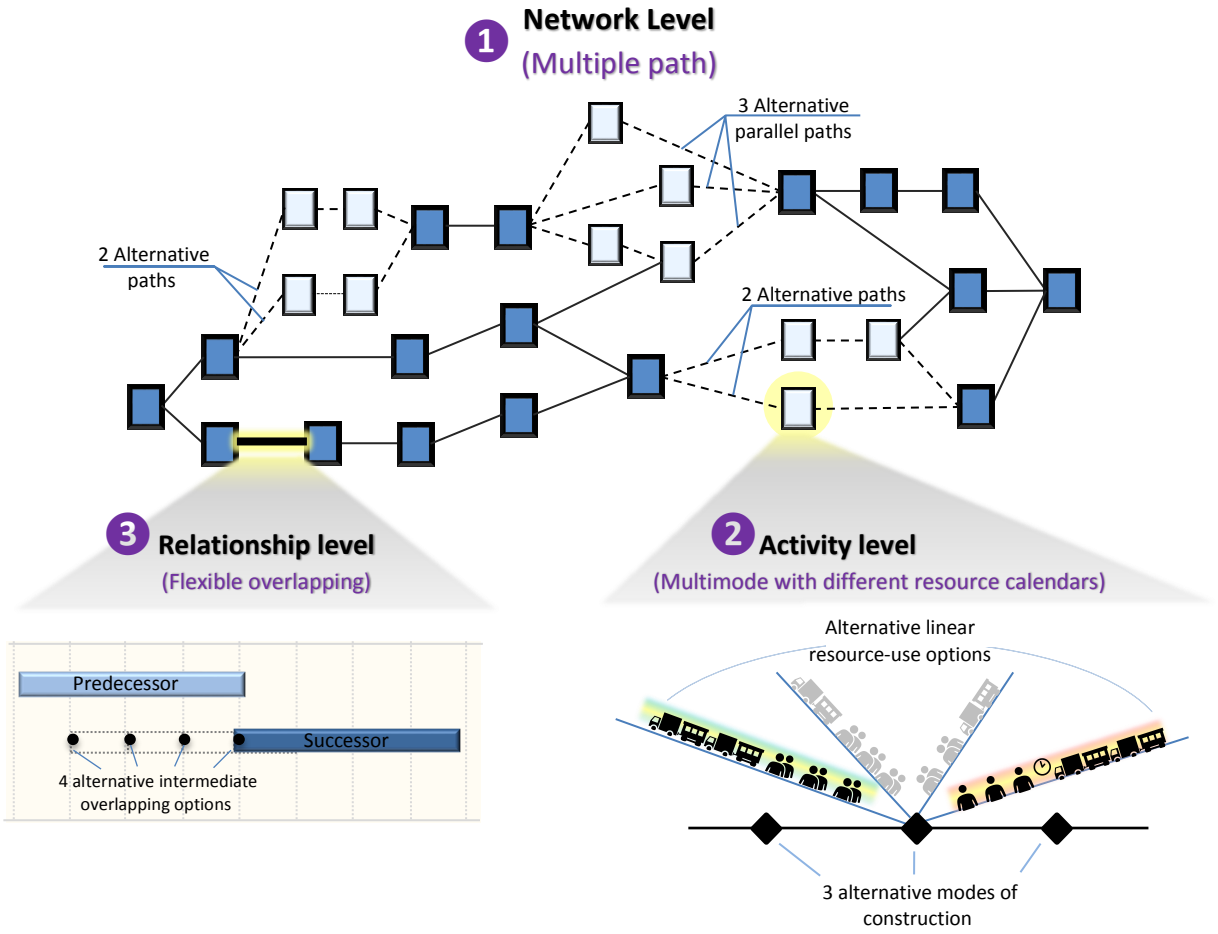


Figure 3.2 The model enhanced options of at different levels

3.4.1 Network-Level: Alternative Paths

As discussed in the literature, a standard assumption of resource-constrained project scheduling problems (RCPSPs) is that the activities to be implemented, and hence the project structure, are given. However, there are cases as discussed when, in the process of planning and scheduling a project, decisions have to be made with respect to optional activities. For example, during the early planning stage (e.g., bidding stage or the early stage of project design) the planner's focuses on work packaging considering different options (e.g., cast-in-situ concrete versus prefabricated elements). Each option has a set of unique activities and a sequence of work. The cast-in-situ option, for example, mandates activities for formwork erection and steel fixing, which are not needed for prefabricated elements. In terms of network scheduling, these options represent alternative paths with different relationships.

Therefore, in this work, enhanced modeling of project network with alternative branches can greatly benefit the planner to select the optimum project recipe that satisfies the early state of knowledge of Phase-I about the project deadline, milestones, and resource constraints (Abuwarda and Hegazy, 2016b). Multipath network is important as well during construction, for Phase-II optimization, to support path substitution decisions which involve the replacement of the early selected path by alternative ones as a corrective action to expedite late project delivery (Abuwarda and Hegazy, 2018). To illustrate the proposed enhancement, Figure 3.3 shows a schematic project network with multiple groups of parallel alternative paths. The proposed network representation handles having multiple groups of multiple alternative paths (branches) and can select between them. Precedence relationships that lead to alternative path in the network are depicted by broken arrows in Figure 3.3. These relationships are only imposed if their path is selected.

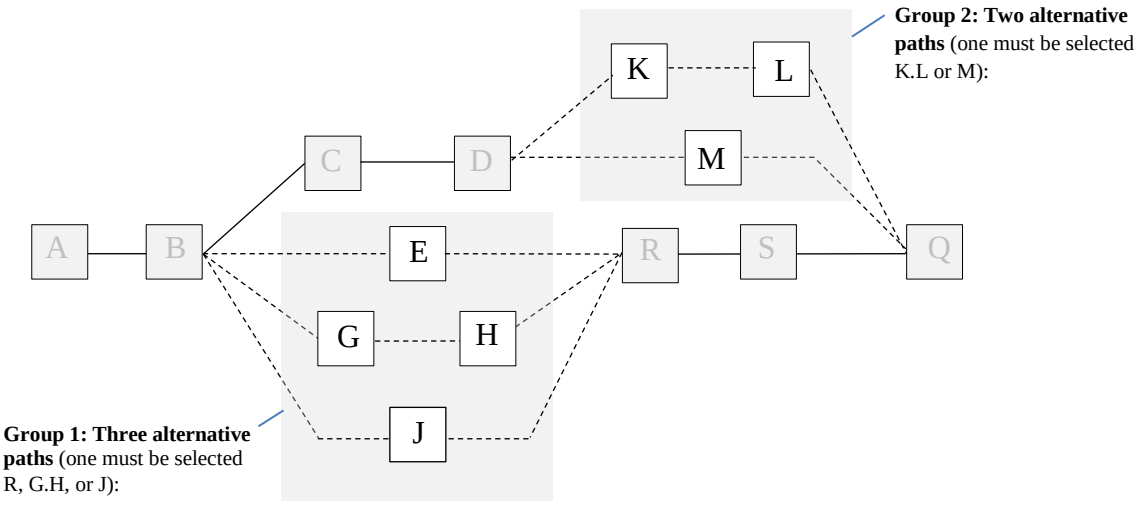


Figure 3.3 Example project network with multiple alternative paths

3.4.2 Activity-Level: Detailed Time-Cost-Resource (TCR) Spectrum

Typically, the representation of the activity option in the scheduling problem is either: Discrete TCT or Linear TCT options. As mentioned in the literature, some researchers mentioned that the linear representation might not be a practical or realistic enough for a number of construction activities, and the discrete is the most representative version of real-world construction projects (Esfahan, 2011; and Golzapoor, 2012). In reality, both representations are essentially based on the activity option. Discrete TCT options represent the discrete set of execution combinations of time, resource and cost that range from inexpensive and slow to expensive and fast due to the application of different technologies. The linear TCT represents the activities linear crashing options like changing the activity resource calendar among options of normal working hours, overtime hours, overmanning etc. that have linear relationship between crash and normal points of time and cost because it can be applied for a segment of the activity

only. Therefore, the author considers that the ideal case is to combine both representations to model the linear change of crew formation or resource calendar within each distinctive mode. Therefore, this research uses the novel approach of combining both relationships in one representation to embody the option of selecting the activity resource calendar/crew formation through the multimode of execution. This representation is called "*Activity Spectrum*".

New activity spectrum: combines the mathematical representation of activity discrete modes with the linear crew options that can be applied through each mode. Activity spectrum is a rich representation that defines crashing options in terms of time, cost, resource calendar (e.g., overtime, overmanning, etc.), and crashing time-segment that can be selected within each possible mode of construction considering the productivity loss due to the use of crashing techniques. An example of Activity Spectrum is shown in Figure 3.4. The figure visibly presents the full spectrum of activity discrete options that use different execution technologies, which are represented by the three diamonds (T_1&C_1), (T_2&C_2), and (T_3&C_3) on a horizontal axis. Within each mode, many linear crashing options are possible, e.g., overtime, overmanning, outsourcing, working weekends, working multiple shift, or a combination of both (as represented in Figure 3.4 for the (T_1&C_1) mode only for simplification). The TCR spectrum, as shown in Figure 3.4, visually represents all modes and crashing options to which facilitates decision and can readily be used in schedule optimization (as discussed later in section 4.2.2.2).

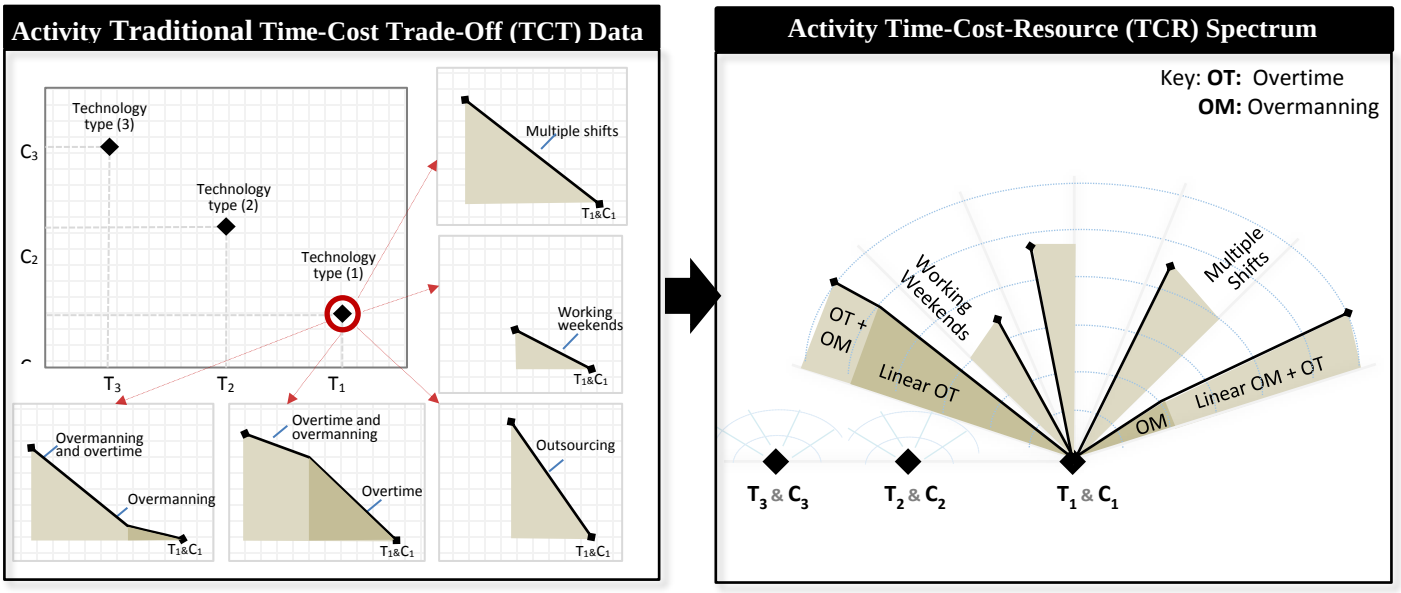


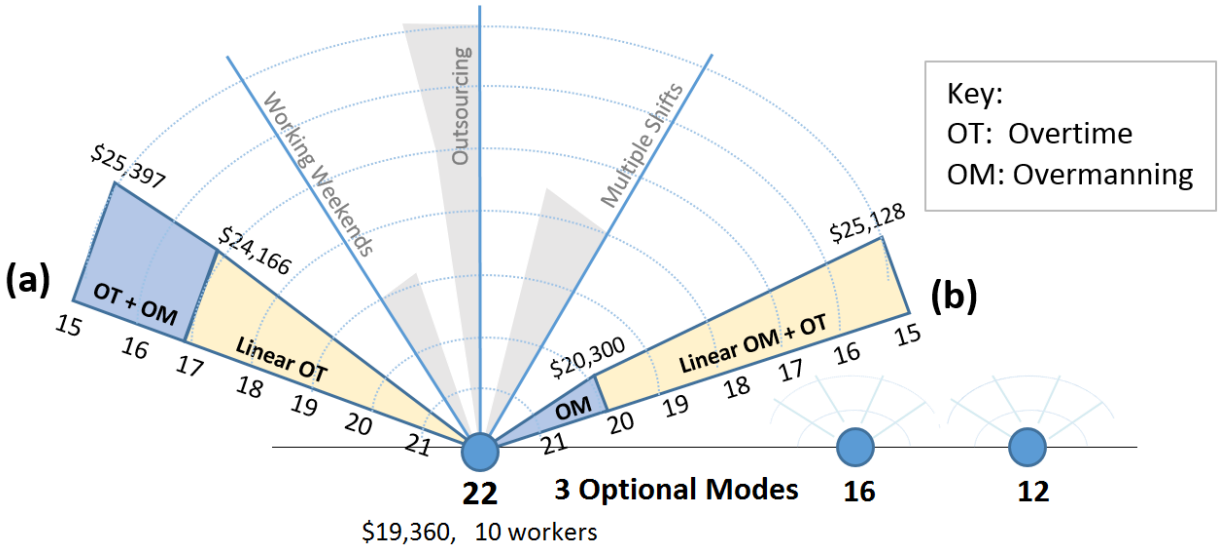
Figure 3.4 Activity traditional TCT data versus TCR Spectrum

A detailed example of TCR Spectrum of an activity (activity 5 in case study-3 discussed later in chapter 5) is shown in Figure 3.5. The inputs include the activity modes, represented in the top section of the figure, indicating modes with 22, 16, and 12 day durations. Within each mode, many crashing options are possible, e.g., overtime, overmanning, or a combination of both. The figure shows a piecewise-linear TC spectrum that visually represents all modes and crashing options of the activity. As shown in Figure 3.5, there are multiple ways to crash the normal 22-day duration, day-by-day to 15 days. As an input to this analysis, practical information is first specified (as shown in Figure 3.5), including the maximum overtime hours (4/day); maximum additional workers (2/day), linear productivity-loss function (as proposed by Hazini et al. 2013); and the hourly rates for overtime and overmanning. The bottom two sections in the automated calculations in

Figure **3.5** are related to two strategies for crashing: overtime first (in

Figure **3.5**a); or overmanning first (in

Figure **3.5**b).



Normal workers =	10	Max OT hrs=	4	Max extra workers =	2			
Normal rate/hr =	$11	OT rate/hr =	$15	Overmanning rate/hr =	$11			
Normal hrs =	8							

All Normal	Duration after Crashing							
22	**21**	**20**	**19**	**18**	**17**	**16**	**15**	
	2.4%	4.9%	7.3%	9.7%	12.1%	14.6%	17.0%	Prodctivity Loss (%)
	1,680	1,600	1,520	1,440	1,360	1,280	1,200	Normal hours
$19,360	$18,480	$17,600	$16,720	$15,840	$14,960	$14,080	$13,200	Normal cost
	123	245	368	491	614	736	859	Needed Extra Hours

(a) Overtime	**3**	**6**	**9**	**12**	**15**	16	15	Overtime days
first, then						4	11	Overmanning days
Overmanning	$20,321	$21,282	$22,243	$23,205	$24,166	$24,870	$25,397	Total Cost
	← Linear Overtime: Cost Slope $961; Segment = 3 days →					← Linear Overmanning →		

Overmanning	**8**	**15**	**19**	**18**	**17**	**16**	**15**	Overmanning days
(b) first, then			**1**	**4**	**7**	**10**	**13**	Overtime days
overtime:	$19,830	$20,300	$21,027	$22,053	$23,078	$24,103	$25,128	Total Cost
	← Linear Overmanning →		← Linear Overmanning + Overtime (segment = 3 days) →					

Figure 3.5 Sample "Activity Time-Cost-Resource (TCR) Spectrum"

Figure 3.5a shows a piecewise-linear function that first applies overtime linearly to crash the activity one day at a time, from 22 to 17 (circled in in Figure 3.5). This linear crashing is associated with extra cost per crashed day (i.e., cost slope) of $961, and a linear overtime period per crashed day (i.e., overtime segment of 3 days). Thus, the cost slope and the overtime segment clearly define the linear crashing characteristics between days 22 and 17 (as used later in the optimization formulation). Continuing the crashing on Figure 3.5a from 17 days to 15 days, the 4 overtime hours for all workers were not sufficient, thus, 2 additional workers were used. This combination of overtime and overmanning is shown on Figure 3.5a as another piecewise portion of the spectrum. The right side of the spectrum in Figure 3.5b visually shows the piecewise-linear portion of the TCR spectrum where overmanning is used first. Because only 2 extra workers are available, overmanning was sufficient only to crash the activity to 20 days (bottom calculation portion of Figure 3.5), afterwards, additional linear overtime hours were needed for further crashing. For demonstration purposes, the detailed calculation of branch (a) and (b) will be explained in depth as illustrated later in Figure 3.7, and Figure 3.8, respectively. In terms of cost, the choice of options depends on the overtime rate as opposed to the cost of overmanning, as well on the productivity loss expected in case of applying overtime hours due to workers' fatigue and in case of overmanning due to less flexibility of workers in limited space, as well as need for more supervision and coordination (Hazini et al. 2013, 2014).

All the crashing strategies are displayed visually on the detailed spectrum of Figure 3.5, which can be extended with multiple shifts and working weekends, for all execution modes. Using the TCR spectrum, crashing is easily associated with a specific resource utilization plan, not only time and cost. TCR spectrum extends the typical activity time-cost function in the form of a piecewise-linear spectrum that defines crashing options in terms of time, cost, resource plan (e.g., overtime, and overmanning), and the "*crashing time-segment"*.

Crashing time-segment: crashing segment is the minimum time period of the activity duration that is required to crash the activity for one day. For instance, in the above example, crashing the sample activity for one day (21 days instead of 22 days) requires 123 overtime hours. These overtime hours can be applied in different scenarios (e.g. overtime the 10 workers 2 hrs on for 6 days; i.e, activity segment = 6 days), while, the minimum crashing segment is 3 days/crashing-day (i.e. overtime 4 hrs all workers for 3 days to crash the activity 1-day and for 6 days to crash the activity 2-days) as shown in Figure 3.6. Crashing segment has a linear relation per crashing day. The crashing segment has a special significance as it can be used to avoid implementing multiple schedule compression strategies (e.g. overtime and overlapping) in the same activity segment, as formulated later in section 4.2.2.8. .

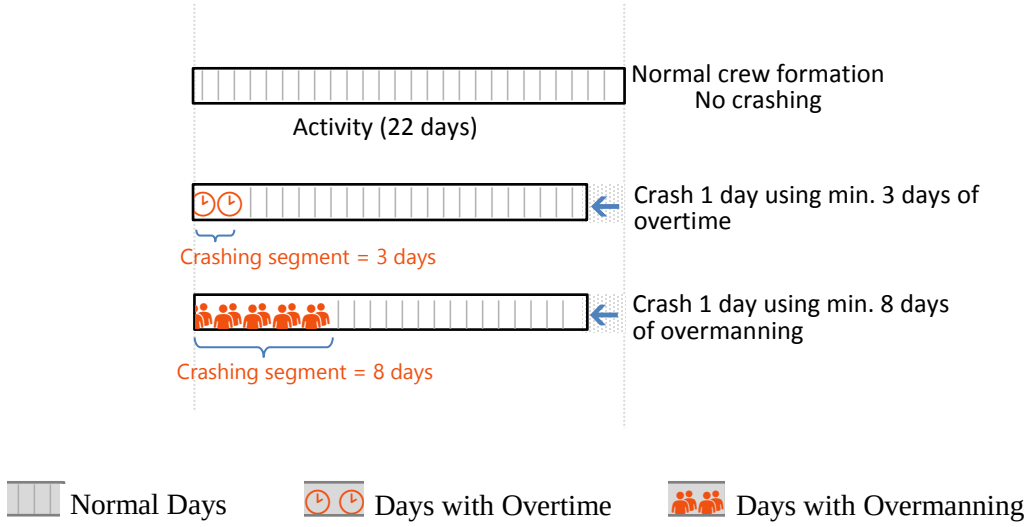


Figure 3.6 Activity Crashing Segment

Enhanced modeling of activity options by incorporating the detailed analysis of the crashing time-cost relationship and the specific implementation plan in terms of hours and amount of resources is

important for the two phases of the proposed framework. For Phase-I, this option supports the selection of the optimum activity mode before construction. For Phase-II, this option also supports the mode substitution, and linear crashing in terms of overmanning, overtime, and outsourcing (Abuwarda and Hegazy, 2018). Using TCR spectrum, the proposed CP optimization models can accurately consider the time, cost, and resource implications to determine a combined decision of the optimum mode and crew formation to consistently keep the projects within its borders.

Details calculation of Spectrum branches: the details calculation of crashing branch (a) of the activity spectrum shown in Figure 3.5 are shown in Figure 3.7. The normal activity duration is 22 days (right side point) while the crashed duration is 15 days (with productivity loss of 17%) and involves 10 workers. Crashing is done first by employing overtime only, without increasing the number of workers.

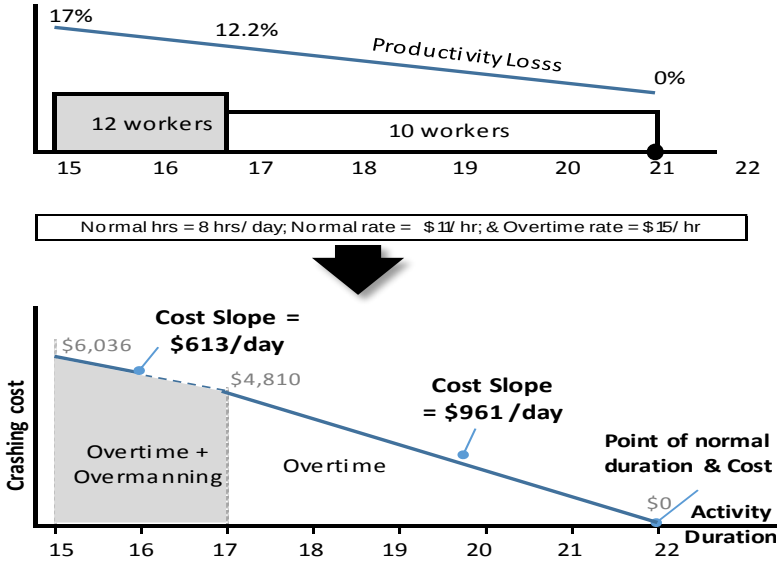


Figure 3.7 Details of branch (a) of the sample activity spectrum

As per Hanna (2003, 2005) overtime, although costly, is a preferred choice because it speeds progress without the coordination problems created by multiple shifts or overmanning. With different overtime

hours, the activity can be reduced from 22 days to 21, etc., until 17 days, beyond which overtime hours become excessive (i.e. more than 4 overtime hrs/day are needed). To reduce the activity duration even further, the next phase is to combine overtime with over-manning (left side of the curve, employing 2 additional workers) considering that overtime hours are more expensive ($15/hr as opposed to $11/hr), crashing costs (i.e., additional cost) are calculated, considering the loss in productivity when using overtime. To simplify the calculations, an assumption is adopted from (Hazini et al. 2013, 2014) that the productivity loss is linear, based on the amount of crashing bounded by the maximum crashed duration (e.g., crash for 1 day) and regardless the method of crashing (e. g. overtime or overmanning). As such, the full time-cost relationship for the example activity is generated at the bottom of Figure 3.7. In the figure, the normal duration of 22 days is associated with total normal hours of 22 days x 10 workers x 8 hrs/day = 1,760 hrs, with cost of (1760 hrs x $11/hr = $19,360). The first stage of crashing is to employ a maximum of 4 overtime hrs/day. When using 12-hour days over the whole activity duration, the total work hours needed considering a 12.2% productivity loss becomes (1,760 * 1.122) or 1,974. This results in a duration of (1,974 hrs /12 hrs/day/10 workers) =17 days, as shown in the Figure 3.7, with 1,360 normal hours and 614 overtime hours. The extra Crashing Cost in this case is $4,810 and the cost slope per day equals ($4,810/ (22-17)) or $961/ day. Also, the required overtime hours for each day of crashing equals (614 / (22-17)) = 123 hrs, which can be arranged in different manners (e.g., 3 overtime hours on 8 days for the 10 workers = 124 hours). The crashing segment which is the minimum overtime segment per crashing equals (123/10 workers /4 overtime hrs = 3 days). To crash the activity to less than 17 days, the second stage of crashing is starting by employing over-manning (12 workers instead of 10). To crash the activity to 15 days, total working hours needed, considering productivity loss of 17%, is 1,760 x 1.17 = 2,060 hrs. As such, 2 additional workers will be employed for only 11 days, with all workers employed 12-hour days. In this case also, the additional

crashing cost becomes $6,036, and the cost slope per day equals $613. Similar calculations are used in case of starting with applying overmanning strategy first and then combined with overtime (branch-b) of the activity spectrum shown in Figure 3.8.

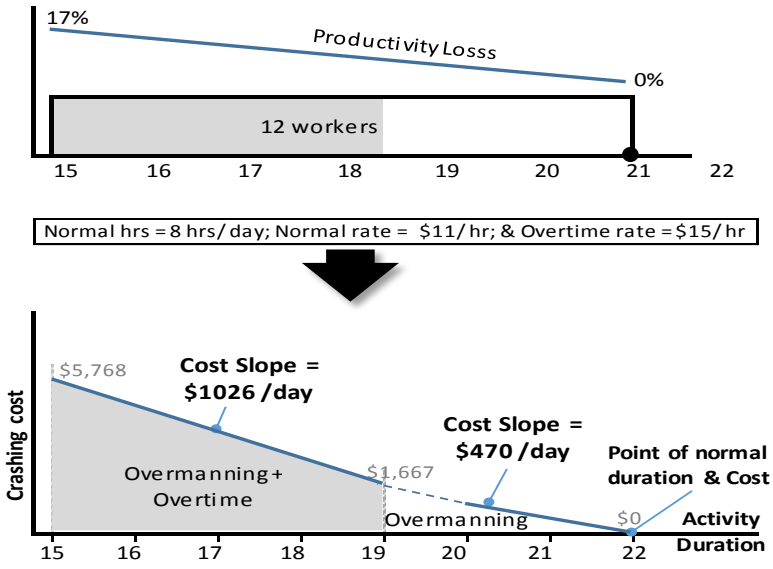


Figure 3.8 Details of branch (b) of the sample activity spectrum

3.4.3 Relation Level: Overlapping using Flexible Relations

In the standard project scheduling problems, a logical relationship between two activities is specified by the relationship type and a fixed lag-time. This rigid representation, however, does not consider the situation when two activities have a degree of flexibility in the relationship. Such flexibility, or soft relation, can be very beneficial as it provides a range of overlapping options that can be utilized in situations that require the schedule to be optimally accelerated. A new formalization of a generic logical relationship (hard or soft) of any type (finish-to-start, etc.), between two activities is used to allow overlaps associated with soft relations, and it is used in the schedule optimization before and during

construction to determine the optimal mix of activity mode, crew formation, and overlapping while accounting for resource constraints (Abuwarda and Hegazy, 2016a, and Abuwarda and Hegazy, 2018). "Flexible relation" or "Soft relation" is used in the proposed model formulation to indicate the permissible overlapping range between two activities. As opposed to the typical hard relations (e.g., "FS 3") that has two parameters (relationship type and hard lag time of 3 days), flexible relations are represented in this work with a third parameter to indicate the degree of flexibility. For example, the hard "FS 3" relationship is more generically represented as "FS 3,1", to indicate that the 3-day lag can be reduced, up to a minimum lag (ML) time of 1 day. As such, a flexible relation is represented in the form of "FS Lag, ML". Thus, the "FS 3, 1" relationship indicates an initial lag of 3 days, which can be reduced to 2, or to a minimum of 1 day (i.e., ML = 1), as shown in Figure 3.9, as permissible choices for the scheduler. A hard relation that permits no change in the lag value, thus, becomes a special case when the ML = Lag, e.g., "FS 3, 3" indicating no flexibility. Figure 3.9 shows one hard relation ("FS 3") and two soft relations ("FS 3, 1" and "FS 3, -1"). In the latter case, the lag can be either 3, 2, 1, 0 or -1 days. When the lag is decided to be -1 (as can be determined by optimization), overlapping occurs and entails some rework in the downstream activity caused by possible coordination challenges.

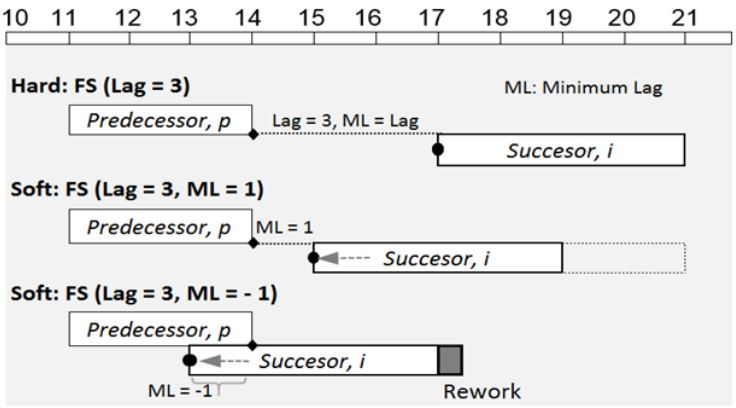


Figure 3.9 Representation of the flexible relationship that allows overlapping

As reported in most literature studies on overlapping, the calculation of rework time and cost related to overlapping can be challenging. Researchers (e.g., Cho and Epinnger, 2005; Gerk and Qassim, 2008, Hazini et al., 2013, 2014; and Dehghan and Ruwnapura, 2014) simplify this calculation by adopting a simple assumption that the rework cost of overlapping a pair of activities is directly proportional to overlapping time. Similarly, the relationship between overlap amount and rework time is linear and continuous. For simplicity, this work adopts this assumption, although the formulation accommodates any functional relation between overlapping time and rework time and cost. Based on this new definition of flexible (soft) relationship, revised equations have been established to compute the Start Time (ST_i) and Finish Time (FT_i) for any activity *i* (Figure 3.10), as a function of the predecessors' times and the ML of various flexible relations (Finish-to-Start, Start-to-Start, Start-to-Finish, and Finish-to-Finish).

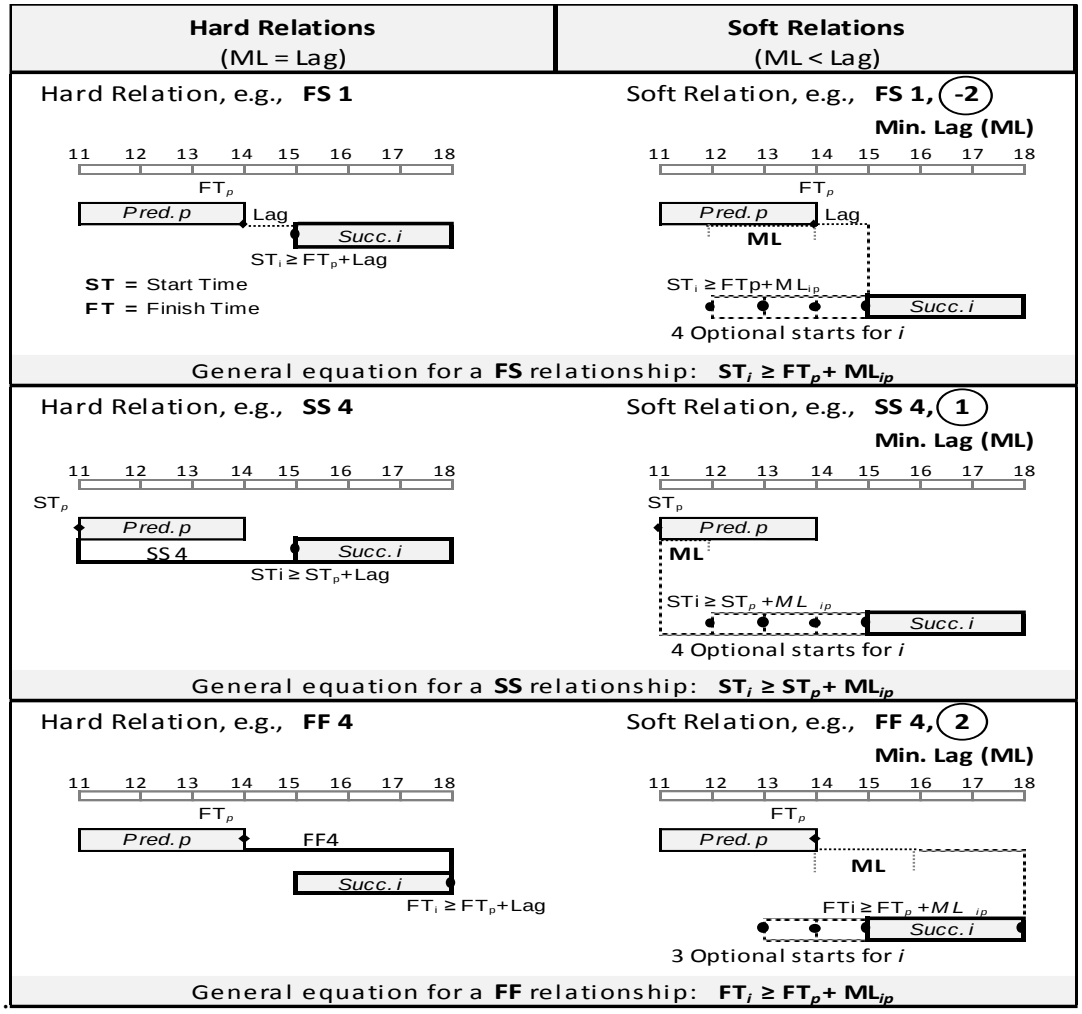


Figure 3.10 Activity start and finish times under different hard and soft relations

3.4.4 Enhanced Practical Constraints

To eliminate the barriers towards developing feasible and practical schedules, the model presents a number of enhanced constraints, as discussed in the following subsections.

3.4.4.1 Generalized Resource Constraints

The basic scheduling problem features only renewable resources where the availability of the resources has been assumed to be constant over time. However, this assumption may not be applicable in some cases, e.g., a change in the availability of workers due to holidays. The proposed enhanced model generalize the representation of resource constraints to embrace different types of resources and time-availability as shown in Figure 3.11 as follows:

1- Renewable resources are available on a period-by-period basis, i.e. the available amount is renewed from period to period and they are recoverable after serving an activity. The total availability of this resource at every time instant is constrained. This availability can be constant over the project period or might be time-dependent as shown in Figure 3.11. Typical examples include manpower, machines, tools, equipment, and space;

2- Nonrenewable or consumable resources are available on a total project basis, with a limited consumption availability for the entire project. Nonrenewable resources are not recoverable and constrained for the overall project (e.g., materials such as tiles, windows, energy); and

3- Doubly-constrained resources are a combination of the two previous categories and are constrained per period (e.g. per period cash flows) as well as for the overall project (e.g. total expenditures, overall pollution limits, and skilled worker).

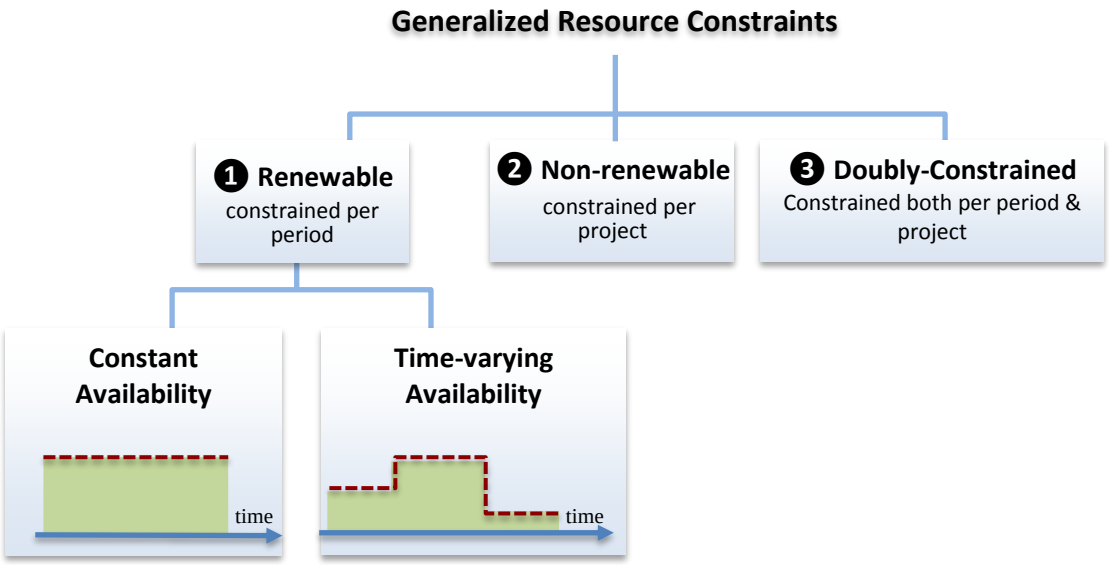


Figure 3.11 Generalized Resource Constraints

To show the importance of considering accurate renewable resource constraints while integrating activity options like activity crashing and overlapping in the scheduling optimization problem, an example is shown in Figure 3.12. The example is showing a schedule acceleration problem using crashing and overlapping strategies. Activity linear crashing increases resource consumption along the activity duration, such as manpower, whereas overlapping activities lead to more resource consumptions during the overlapped period of time. Thus, overlapping is less applicable with limited resource constraints. However, as discussed in the literature, many schedule compression models that integrate crashing and overlapping did not consider the resource limits.

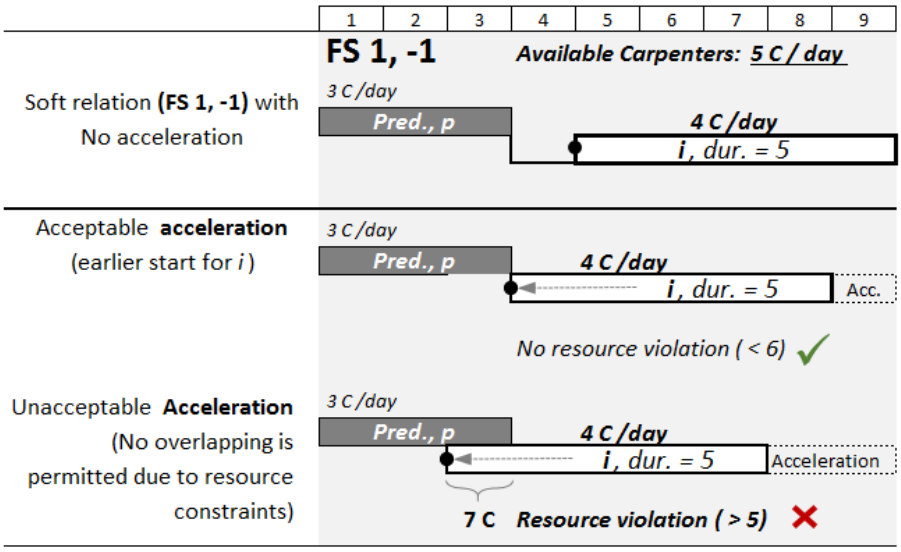


Figure 3.12 Effect of resource constraints on acceleration strategies integration

3.4.4.3 Activities' Mode Correlation

In construction projects, the planner needs to constrain their schedule to standardize methods or use the same subcontractor for different activities, to save cost. One of the important enhancements of the proposed model is enabling the planner to define a correlation between any pair of modes of two or more activities (Abuwarda and Hegazy, 2016b). The selection of one mode for the first activity necessitates that another mode for a second activity be selected too.

Correlation is mutual relation of two or more things based on co-occurrence. Generally, literature studies in multimode schedule optimization models have assumed that activity modes are mutually independent, and the selection of the activity modes does not take into account the interdependencies or correlation among different activities' modes in order to facilitate their models. To the author knowledge, very few efforts, like Hegazy and Togla (2001), have used contingent activity groups in their model.

3.4.4.4 Intermediate Milestones

One of the main scheduling constraints before and during construction is to meet intermediate target dates (milestones) for key project activities (Ashuri and Tavakolan, 2013; and Abuwarda and Hegazy, 2016b). Milestones are significant events in the course of a project that mark the completion of a project phase or the completion of a major de liverable (McCormick 2002). Most contracts of large-scale projects allow penalty and incentive payments in respect of intermediate milestones, particularly if these milestones give the owner an opportunity to start using parts of the project. To enhance the proposed model considering the intermediate milestones, the criterion used is to minimize the total project cost including the incentives associated with completing certain tasks earlier than a given date or tardiness costs associated with completing certain tasks later than a given date (Abuwarda and Hegazy, 2016b).

3.4.4.2 Rework Constraints

A constraint is added to the model to limit excessive rework that could be resulted from the cascade of overlapping between the activity and its multiple predecessors. The activity entails excessive rework due to either the high degree of overlapping between the activity and its predecessor or the activity is overlapping with two or more predecessors simultaneously. To avoid schedule complications and risks that could happen due to excessive overlapping, rework constraints are added to the downstream activities so the resulting rework does not exceed a ratio of the original activity duration. The ratio is assigned individually to each activity based on the size and complexity.

3.4.4.3 Integration Constraints

As the proposed model is enhanced to consider multiple strategies (e.g. linear crashing, overlapping, and substitution) to accelerate the schedule at any stage of the project to maintain the target date, it is important to represent the desire to integrate the use of these strategies (Abuwarda and Hegazy, 2018). In essence, overlapping, crashing, and substitution are not mutually exclusive, they can be used simultaneously. In some situations, therefore, it may be necessary to the planner to avoid having crashing, overlapping, and substitution occurring simultaneously because extensive using of these strategies can lead to space congestion and overstressed workers. The proposed schedule optimization uses optional constraints to prevent the simultaneous use of activity crashing, overlapping, and substitution at the same activity leading to more practical schedule.

3.4.5.5 Activity-Segment Constraints

Alternatively to the integration constraints, in a response to the need for highly compressed schedules in some cases, the proposed schedule optimization permits the use of crashing and overlapping strategies collectively for the same activity. However, the model uses optional constraints to prevent the simultaneous use of overlapping during the activity crashing time-segment, leading to a less stressful work environment.

Identifying the activity segments to perform crashing and overlapping was investigated in few literature efforts. Among them, Roemer and Ahmadi (2004) presented a study to integrate crashing and overlapping and they stated that crashing the early part of the predecessor activity will decrease the

impact of rework on the successor activity. Later, Hazini et al. (2013) suggested dividing the activity into three segments to integrate crashing, and overlapping. As shown in Figure 3.13, the overlapped segments are not recommended to apply crashing strategies. However, they did not use the three segments in their optimization model. In this work, the detailed specification of crashing strategies and the calculation of crashing segments adds a very critical potential solution to prevent the integration of multiple techniques, i.e. crashing and overlapping, simultaneously in the same activity segment as illustrated later in section 4.2.2.8.

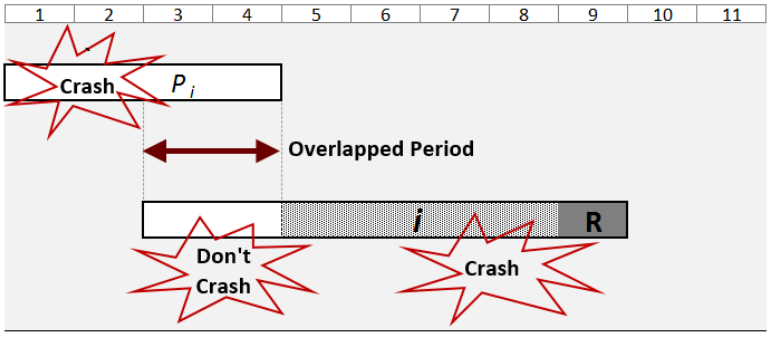


Figure 3.13 Functioning activity segments for crashing (adapted from Hazini, 2013)

3.4.5 Alternative Scheduling Objectives

The proposed model can handle a broad variety of objective functions for project scheduling models other than minimizing the project total cost or the project total duration, which are commonly used in the literature. The formulation of the objective function of Phase-1 framework considers time-based, cost-based, and resource-based objectives, separately or simultaneously using single and multiple objective functions. Accordingly, the project manager decides his desired objective function (e.g., minimizing total cost) while all other features of the schedule are controlled by a set of constraints (e.g.,

resource usages and deadline), as discussed later in section 4.2.3. The enhanced objective functions are either: cost-based objectives (e.g. minimizing the total cost incentive/ penalties with respect to important milestone due dates, and/or minimizing total corrective-action cost. The models can incorporate any cash flow objectives as well time-based objectives (e.g. minimizing total project duration). The models incorporate generic total project duration objective by minimizing the weighted tardiness with respect to important milestones due dates. To find a good compromise between the time, and cost, the proposed model applies multi-objective functions using alternative approaches to simultaneously consider several objectives.

3.5 Framework Extension for Schedule Optimization during Construction

This section presents the proposed extensions of the Phase-I framework to support corrective-action decisions throughout the project execution. The Phase-II framework for corrective-action planning at any reporting period during construction is schematically shown in Figure 3.14. The process starts at the end of any reporting period r, and follows four steps as follows (details in the next subsections):

- Step 1 - "Schedule Updating": the baseline (S_r) schedule at reporting period r (note: at the first reporting period, baseline (S_o) that resulted from Phase-I optimization) is updated with the current progress data, where the daily site reports provide the required progress information, work condition, evolved requirements, and/or new activities, constraints, and milestones. This step ends by asking the project manager to determine the need for corrective-action plan or not;

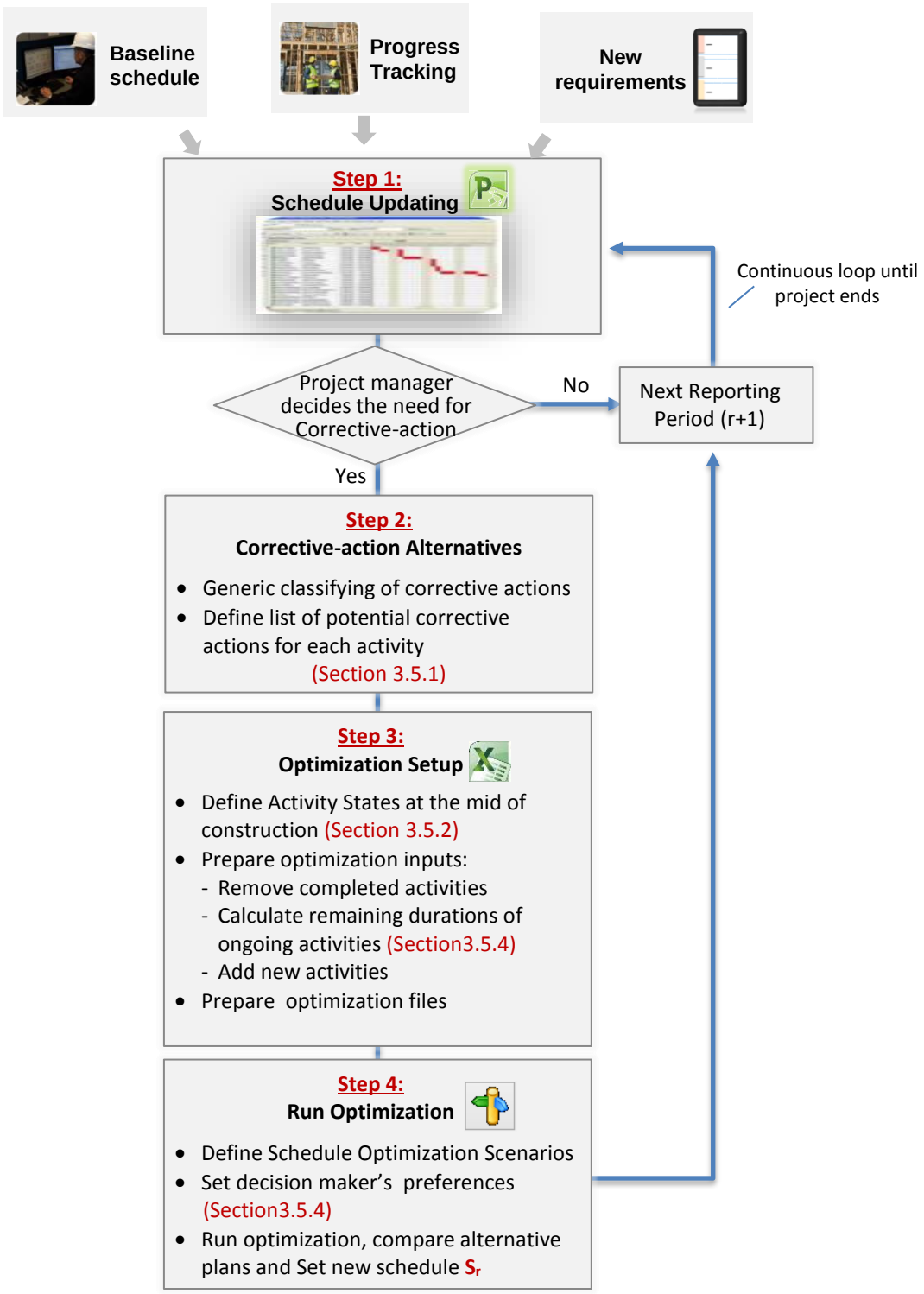


Figure 3.14 Phase-II framework for corrective-action planning at any reporting period r

- Step 2 - “Corrective-action Alternatives”: defines the possible list of corrective actions for each project activities and classify them according to a generic classification (as discussed later in section 3.5.1) according to how they will be dealt mathematically in the optimization model;

- Step 3 - “Optimization Setup”: which separates the project activities into four categories, according to their status: completed; ongoing; new; and upcoming. The model then determines how these different statuses of activities will be handled in the optimization. Accordingly, the model automatically prepares the required inputs for the corrective-action optimization; and

- Step 4 - “Run Optimization”: asks the project manager to set preference scenarios regarding the period(s) of schedule changes. The optimization result chosen by the manager then defines the revised schedule, including activities’ modes and start times. This plan will be the inputs to next reporting period (r+1).

3.5.1 Classification of Corrective Actions

In practice, project managers have various strategies to speed up the project: *Activity Crashing* by maneuvering the activity resources through working overtime hours, working multiple shifts, weekends, and/or adding more workers; *Activity Mode Substitution* by selecting alternative execution option that has different types of resources and that range from a cheap (slow) mode to an expensive (fast) mode; *Path-Substitution* by replacing of a set of activities in series (i.e., part of a path) by an alternative one; *Activity Overlapping* by changing the work sequence from being in-series to being partially parallel, to save time. The proposed framework classifies wide variety of corrective actions

into six categories according to how they are mathematically handled in the optimization model as alternative options as shown in Figure 3.14

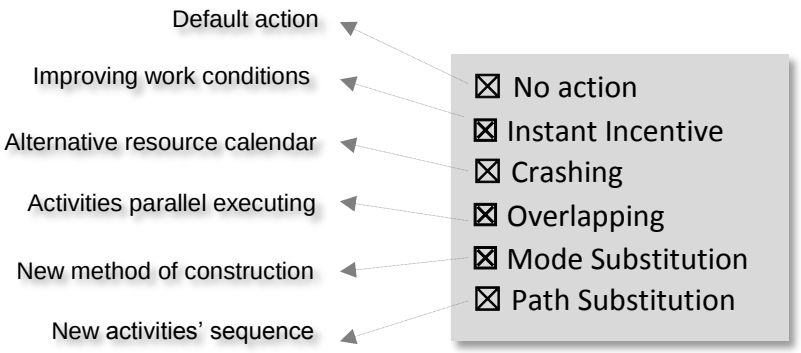


Figure 3.14 Generic classification of corrective actions

This classification is intended to encompass the most common actions that are chosen by project managers in practice and place them under these six categories. The detailed categorized list of possible corrective actions for each project which are often very project specific can be customized by the user. Table 3.1 introduces an example of a detailed corrective-action list for an activity *i* indicating the required considerations of each options in the optimization model.

Table 3.1 Example of Corrective-action List

Action Category	Action example	Optimization considerations
1. No action	(The default)	
2. Improve worker Incentive 2.1 Financial action 2.2 Non-financial action	Financial action - Monetary Incentive programs - Technical Training and career development Non-financial action - Non-monetary rewards - Social activities outside the work environment - New company retention plan	- Not considered as a schedule variation - Decreases activity duration with or without cost increase -Applies only to ongoing activities
3. Activity crashing 3.1 Overtime 3.2 overmanning 3.3 shiftwork	(keep the same construction mode with new resources' arrangement) - Use of scheduled overtime - Additional resources equipment/ labor - Employing staggered shifts for interfering trades - Use of shift work	- Considered as a schedule variation - Decreases activity duration and linearly increases cost (Linear TCT) -Applies only to ongoing and short-term activities
4. Overlapping	(Parallel execution of precedence related activities)	- Causes schedule variation based on start-time changes - May cause rework time and cost - applies only to ongoing and short-term activities
5. Activity Mode Substitution	- New method of construction - Use alternative subcontractor	- Causes schedule variation - Decreases the activity duration and increases cost (Discrete TCT) - Applies to any activity, respecting contractual commitments
6. Path Substitution	(affect more than one activity) - Use prefabricated elements instead of cast-in-situ	- Causes schedule variation - Applies to long-term activities

3.5.2 Activity States during Construction

During construction, the running activities change in accordance with the conditions of the work at the construction site. Activities' quantities can change, and due to change orders, activities can be added and/or omitted, the owner adds new requirements, and risks evolve. Accordingly, after the "schedule updating" step which identifies the project schedule changes due to actual progress of project activities,

the "Optimization Setup" prepares the required inputs for the optimization of corrective actions. At the current time (T_r), which is at the end of reporting period r, the setup process configures the activities, which are classified into four types: completed, ongoing, upcoming, or new activities, as shown in Figure 3.15. The figure highlights these activity types and the list of corrective actions that suit each type. The definition of these activity types and how they are handled in the optimization are as follows:

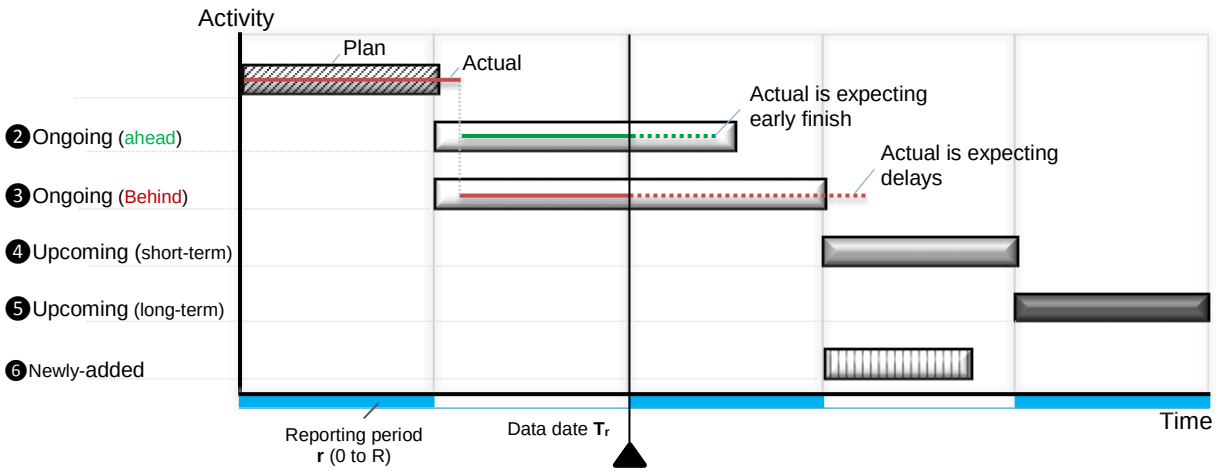
1- Completed Activity:

- *Definition:* actual progress (P_i^a) $= 100\%$, where P_i^a is the actual (a) progress (P) of activity (i)
- *In optimization:* This activity will be deleted during the corrective-action optimization
- *Possible corrective actions:* Not applicable for completed activities

2- Ongoing Activity:

- *Definition:* Has Actual Start Time (ST_i^a) before the current time, and Actual Progress (P_i^a) amount less than 100%, i.e, ($P_i^a < 100\ \%\quad and\quad ST_i^a \leq T_r$)
- *In optimization:* completed portion will be removed and only its remaining part will be considered in the optimization as an activity with Start Time equals T_r and duration equals its remaining duration. The calculation of the ongoing activity's remaining duration is illustrated in the next subsection
- *Possible corrective actions:* the possible corrective-action list of ongoing activities might include Instant incentives, crashing, mode substitution, overlapping, and path substitution. However, mode substituting for some ongoing activities can greatly disturb the work

and could be undesired. So, the user is required to indicate the possibility of each task to consider switching to a faster mode, if needed.



	❶	❷	❸	❹	❺
Activity State (Progress)	Completed	Ongoing (ahead)	Ongoing (behind)	Upcoming Short-term	Upcoming long-term
How to handle In Optimization	Totally removed	Completed portion removed. Activity has start time equals T_r and the remaining duration as a function of the possible corrective-action (Figure 3.16)		Add New activities	
Possible List of Corrective Actions	☐ No action ☐ Instant Incentive ☐ Crashing ☐ Overlapping ☐ Mode Substitution ☐ Path Substitution	☒ No action ☐ Instant Incentive ☐ Crashing ☐ Overlapping ☐ Mode Substitution ☐ Path Substitution	☒ No action ☒ Instant Incentive ☒ Crashing ☒ Overlapping ☒ Mode Substitution ☐ Path Substitution	☒ No action ☐ Instant Incentive ☒ Crashing ☒ Overlapping ☒ Mode Substitution ☒ Path Substitution	☒ No action ☐ Instant Incentive ☐ Crashing ☐ Overlapping ☒ Mode Substitution ☒ Path Substitution

Figure 3.15 Four activitiy states during construction and possible corrective actions

3- Upcoming Activity:

- *Definition:* Activity not started yet, i.e., $(P_i^a < 100\ \%\quad and\quad ST_i^a \geq T_r)$

- *In optimization:* These activities are two types: upcoming in the short-term, and upcoming in the long-term, according to activity's time, from its planned start to the T_r date and the user preference about how many reporting periods are considered short-term.

- *Possible corrective actions:* For short-term activities, the list of corrective actions include crashing, mode substitution, overlapping, and path substitution. For long-term activities, same options except linear crashing and overlapping as these are not reasonable to use.

4- New Activity:

- *Definition:* Newly introduced activity, i.e., considered as an upcoming activity (type 3)

- *In optimization:* This activity and its cost and relations are inserted into the schedule

- *Possible corrective actions:* Same as upcoming activities

3.5.3 Calculation of remaining duration of ongoing activities'

As discussed above regarding ongoing activities, each activity *i* will be replaced by a new activity *i`* with a start time ($ST_{i`}^{Sr} = T_r$) to force activity continuity. For ongoing activities, depending on their progress data, worker's morale, and the potential corrective-action list, the calculation of the remaining duration is made. This study adapts and refines equations proposed by Hegazy and Petzold (2003) to calculate the remaining duration, as follows:

Case (a) - ongoing activity i is ahead of schedule (i.e. $P_i^a \geq P_i^{So}$): While current progress is ahead of the plan, a conservative estimate of remaining duration assumes the remaining work will follow the plan. Hence, the remaining duration of the activity is calculated as in Eq. 3.1. This

activity will continue with the same method of construction without need for corrective actions. As such, no optimization variables are used for these activities although the activity data are used to accumulate project costs and resources.

$$D_i^{S_r} = \left(1 - P_i^a\right).D_i^{S_o} \quad (3.1)$$

Case (b) - ongoing activity i is behind schedule (i.e. $P_i^a < P_i^{S_o}$): Since the activity is slower than planned, remaining duration depends on the type of corrective-action decision, as shown in Eqs. (3.2 to 3.5):

- No corrective action: remaining duration is based on actual progress, as follows:

$$D_i^{S_r} = \left(T_r - ST_i^a\right).\left(1 - P_i^a\right)/P_i^a \quad (3.2)$$

- Instant incentives to improve workers' morale, remaining duration follows planned progress, as follows:

$$D_i^{S_r} = \left(1 - P_i^a\right).D_i^{S_{r-1}} \quad (3.3)$$

- Appling overtime and / or overmanning strategy, remaining duration follows weighted sum of planned and actual progress w_1, w_2 as set by user respectively:

$$D_i^{S_r} = w_1.\left[\left(T_r - ST_i^a\right).\left(1 - P_i^a\right)/P_i^a\right] + w_2.\left[\left(1 - P_i^a\right).D_i^{S_{r-1}}\right] \quad (3.4)$$

- Switching to faster mode of construction k,

$$D_{ik}^{S_r} = \left(1 - P_i^a\right).D_{ik} \quad (3.5)$$

It is important to mention that an assumption of the linear relationship between activity duration and cost is made to calculate the associated cost of each duration as shown in Eq. 3.6.

$$C_{ik}^{S_r} = D_{ik}^{S_r}.(C_{ik}/D_{ik}) \quad (3.6)$$

An example of ongoing activity is shown in the top part of Figure 3.16 (Activity F in case study-4 discussed later in chapter 5), showing a 190-day planned duration (mode 1). As shown in the bottom part of the figure, the activity started on day-50 as planned. By day-120 (reporting period date), only 25% progress was completed instead of the planned 37%. In this case, the list of potential corrective actions for this activity varies from "No Action" to "Mode Substitution" (activity has three other optional modes that are faster and more expensive than the planned mode). The table on the bottom part of Figure 3.16 calculates the remaining duration for each possible corrective action, in addition to the extra associated cost. The data in this table is readily used in the corrective-action optimization.

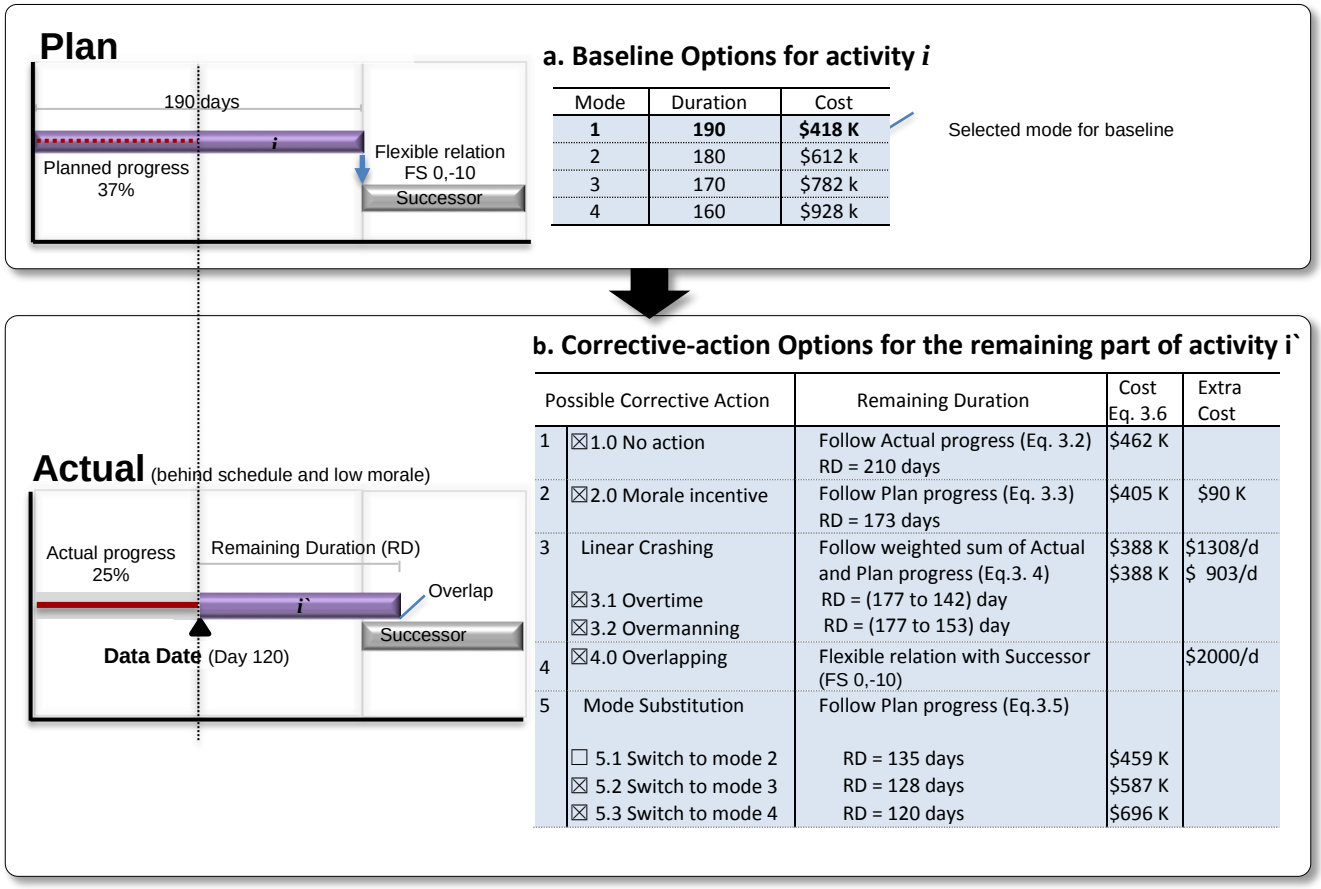


Figure 3.16 Example ongoing activity with planned data versus data during construction

3.5.4 Defining Scenarios with Different Corrective-Action Preferences

In construction projects, the baseline schedule upon which the contract has been signed should be followed. In reality, following the baseline schedule is difficult, let alone maintaining it after time and cost deviations that occur during construction. As discussed previously in section 3.3, the main unresolved challenge of making scheduling decisions during construction is to maintain the initial schedule and control the schedule changes that are required to get the project back on track.

Typically, the classical dynamic scheduling models target the minimization of the Total Project Cost as a principal goal of construction companies. However, adopting total cost minimization for rescheduling problems means schedule changes can occur at any period within the planning horizon. Recognizing the time periods of deviating from the initial schedule is critical and important for contractors who are typically responsible for contract execution and dealing with suppliers, subcontracting and outsourcing during construction. They are always have specific preferences regarding having schedule changes in the immediate period or in future periods. These preferences vary based on which periods of the schedule project managers consider more risky to change. Figure 3.17 shows two scenarios of project recovery plans that either focuses all schedule changes on the short-term (Figure 3.17a), or another situation where the preference is to exercise all corrective actions at the long-term (Figure 3.17b). In the former case, the project manager is looking for quick recovery from delays, while retaining the schedule flexibility for further corrective actions in the future. Whereas, in the latter case, the project manager is avoiding to change the commitments made at the short-term.

Added to the project manager's preference in the corrective-action period (short versus long-term), the project manager may need to consider the individual activity desirability to be included in the

corrective-action plan. This depends on the availability of resources, activity complexity, and the logistics related to the activity work. In general, despite of their importance, these preferences of project managers are not typically included in the literature efforts related to schedule optimization, either before or during construction

.

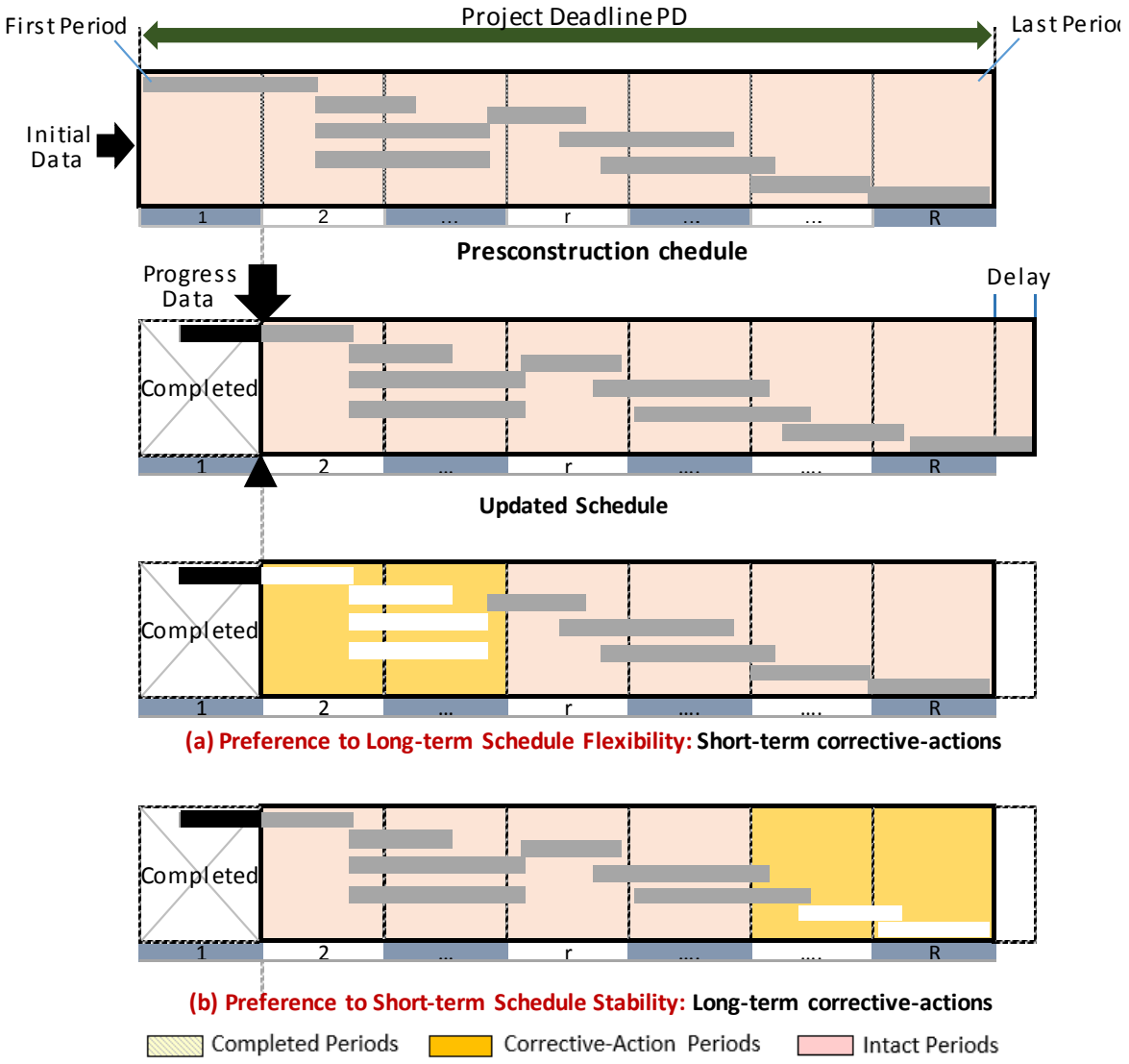


Figure 3.17 Short-term vs. Long-term corrective-action scenarios

3.5.5 Representation of project manager's preferences in the schedule optimization

To develop a satisfaction-driven optimization, the proposed framework introduces two-dimensions of project manager's preferences as discussed below:

a. Preference regarding the unwanted schedule deviations: The proposed framework utilizes preference functions as an extension to Phase-I schedule optimization model to represent the project manager's preference regarding the periods of unwanted deviations between the baseline and the new schedule associated to increasing schedule long-term flexibility, or schedule short-term stability as discussed. Examples of preference functions that are used commonly research related to modeling of decision makers' preferences (e.g. Martel and Aouni, 1990, and Allouche, M. 2014) are shown in Figure 3.18. These preference functions are normalized in the interval [0, 1] in such a way that an upper bound for the function is 1 means "unwanted deviation" and the lower bound for satisfaction is 0 means "possible deviation". As shown in Figure 3.18, the preference functions (usually referred to as satisfaction functions in the literature) can be either a step function as present in Figure 3.18a, and Figure 3.18c, linear as in Figure 3.18b, or mixed as Figure 3.18d, where all shown functions are descending but can also be ascending.

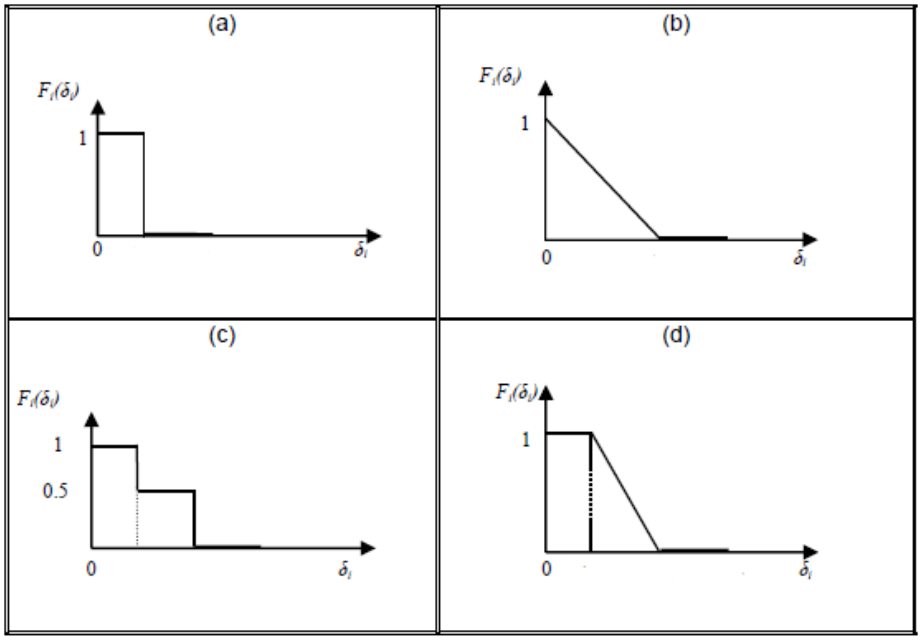


Figure 3.18 Multiple shapes of preference functions (adopted from Allouche. 2014)

The manager can choose any of these shapes to explicitly choose one of three scenarios, as follows:

Scenario-1: "No Preferences", where the corrective actions and schedule changes can happen in any period along project duration. This scenario represents the traditional TCT problem where the objective function is to minimize total project cost.

Scenario-2: "Short-term Corrective-actions" The project manager prefer having long-term schedule flexibility and hence, unwanted schedule changes during the long-term reporting periods. This scenario assigns step ascending, or linear ascending function along the project duration as shown in the top part of Figure 3.19a.

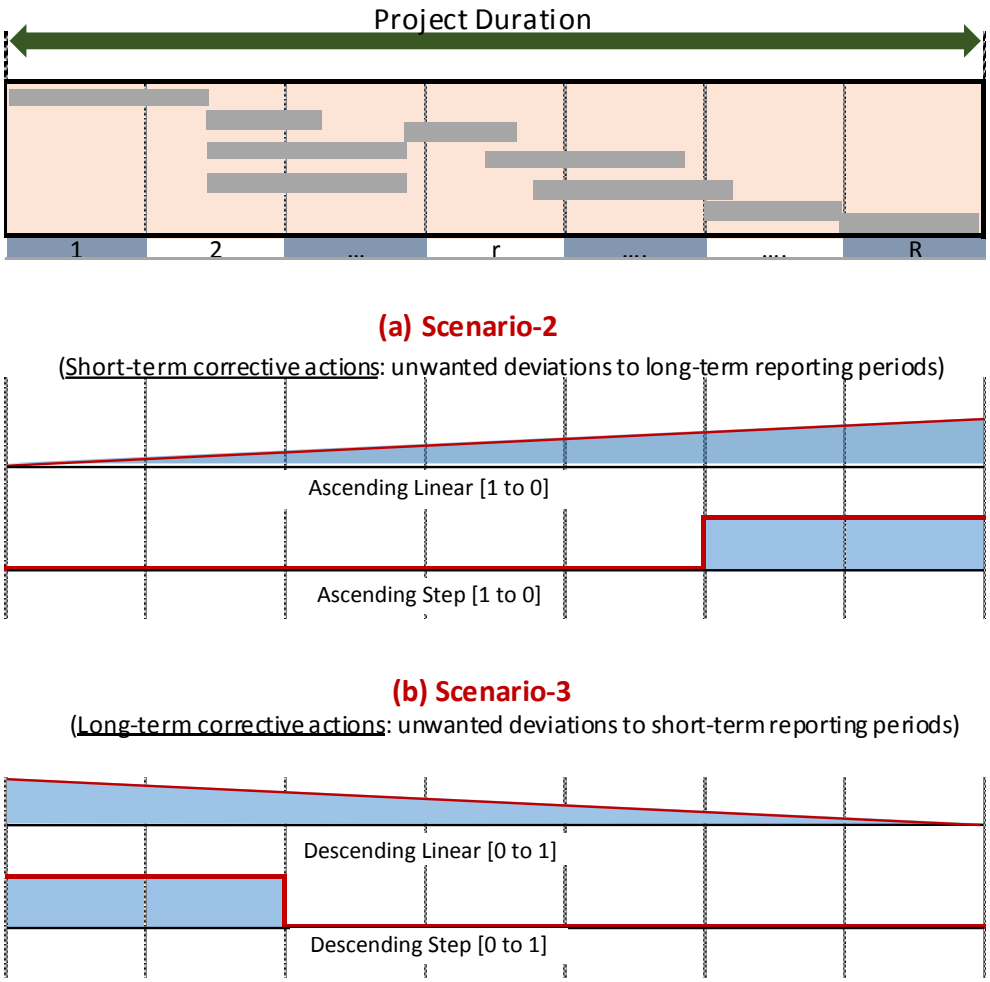


Figure 3.19 Adopted preference functions and corrective-action scenarios

Scenario-3: "Long-term Corrective-actions" The project manager prefer having short-term schedule stability and hence, unwanted schedule changes during the short-term reporting periods. This scenario assigns step descending, or linear descending along the project duration as shown in the bottom part of Figure 3.19b.

The proposed framework utilizes the selected scenario to formulate a *New Objective Function.* To maximize the project manager's satisfaction, the new objective is minimizing the total schedule variations weighted by the project manager's preference function. The new objective function replaces the objective functions of the Phase-I model of minimum Total Cost, minimum Project Duration, or minimum Resource Fluctuations (Section 3.4.5). These objectives can be used as secondary objectives or in a multi-objective decision environment depending on the user preferences as well. The schedule variations is defined by the summation of two parameter (1) activities' start times deviations between the new schedule and the original schedule, and (2) activities' mode variations other than their modes in the original schedule as illustrated later in the next chapter in Eq. 4.39.

b. Preference regarding the activity flexibility to mode change during construction: The proposed framework utilizes a second preference dimension regarding the flexibility in changing the construction methods of each activity during construction. The activities will be grouped into either: flexible; possible but not desirable; or inflexible. The developed framework of Phase-II utilizes these groups during the optimization to determine the selected activities in the pool of candidates for optimization. Accordingly, during the optimization, it is possible to consider the first group of activities (flexible) not the others. To exclude the others, constraints are added to fix their activity modes to their baseline. If the solution is not satisfactory or cannot be obtained, then another round of optimization can be tried with the first two groups of activities considered as variables. If still no feasible solution is obtained, then a third optimization experiment can be tried with all the three activity groups as variables.

3.6 Summary

This chapter investigated the real-life parameters, decisions, and constraints on project scheduling, before and during construction. Experts in the field of construction were consulted. Based on the investigation results, the structure of the proposed framework was designed to combine two distinctive phases of schedule optimization: "Phase I: preconstruction", and "Phase II: during construction".

The schedule optimization model Phase-I introduces a new representation of the decision options at the network level, activity level, relation level, and project level. This model was then extended to develop Phase II schedule optimization model that improve the decision making of corrective actions and recovery plans during construction. This generic model considers the baseline schedule, evolving constraints, a generic list of possible corrective-actions that suit different activity types, and two dimensions of project manager's preferences about corrective-action implementation scenarios. The detailed mathematical formulation of the framework and its implementation will be presented in chapter 4.

Chapter 4

Mathematical Formulation and Implementation

4.1 Introduction

This chapter first presents the mathematical formulation of the developed optimization model starting with "Phase I: Pre-construction", which will be extended in a successive step to model "Phase II: During construction". The mathematical formulation of the decision variables, constraints and objective functions are presented in detail, followed by a description of the working procedure to implement the mathematical formulation in the CP optimization.

4.2 Mathematical Formulation of Schedule optimization Before Construction

To accommodate the optimization options on network, activity and relation levels that are identified in chapter 3, a detailed mathematical formulation has been developed. Similar to any optimization model, the model has three main components: 1) a set of decision variables, which control the value of the objective function, 2) sets of constraints and data, which control the variables to take on certain values but exclude others and 3) an objective function, expressing the main objective of the model, to be either minimized or maximized. The entire model builds a relationship between the objective function, constraints, decision variables, and known data to find values of the variables that minimize or maximize the objective function while satisfying the constraints. Following are the decision variables, constraints and objective functions of the proposed model, respectively.

4.2.1 Decision Variables

The model five key decision variables for each activity i are represented in Equations 4.1 to 4.7 and Figure 4.1, Figure 4.2, and Figure 4.3.

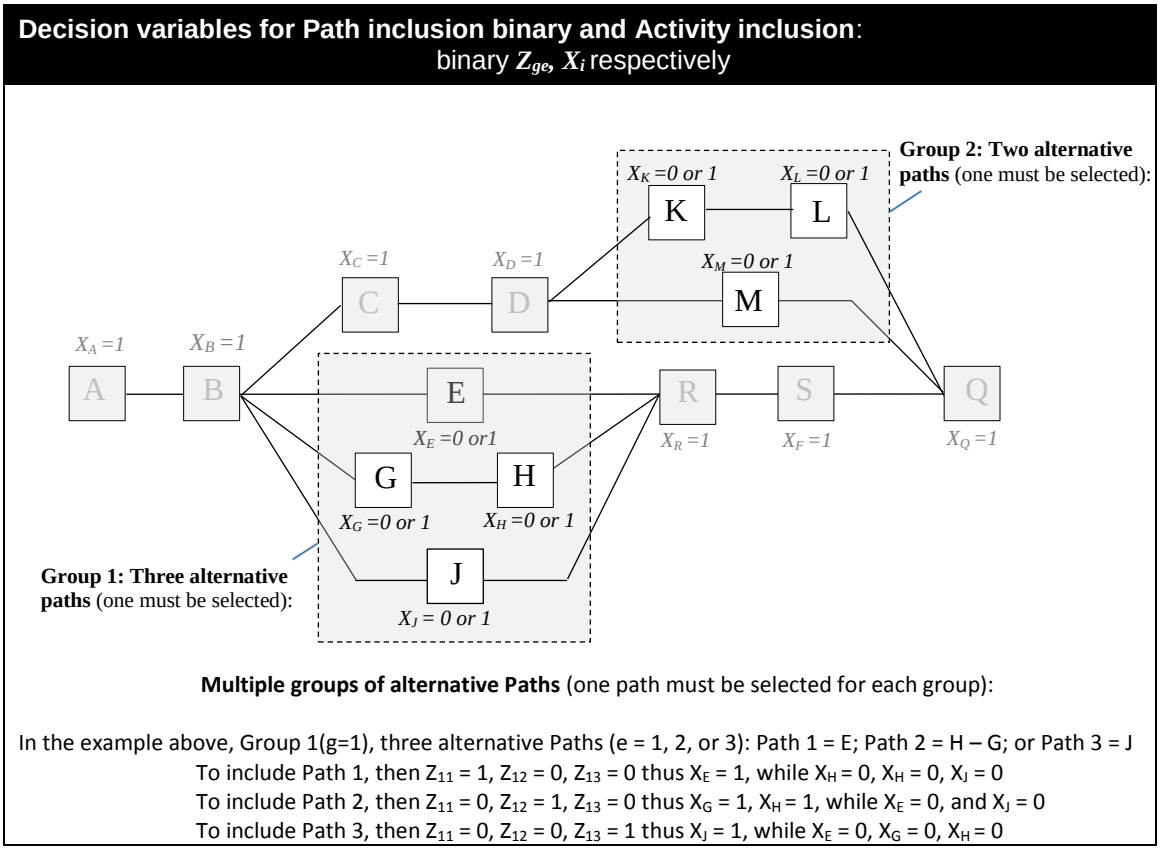


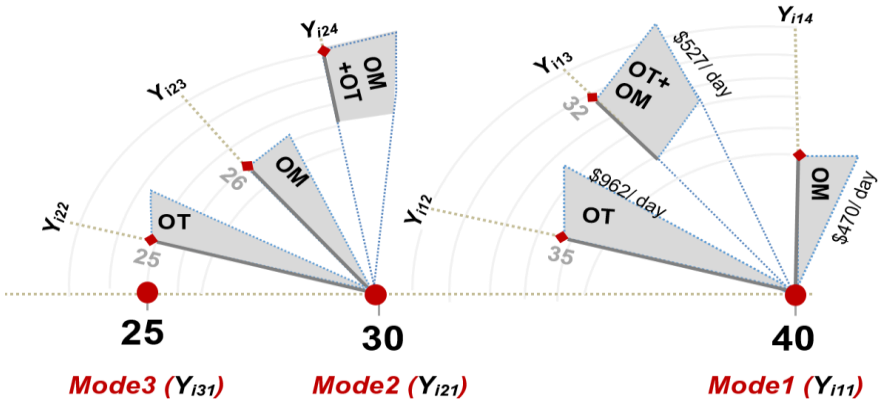
Figure 4.1 Representation of the Z and X variables

To finalize the set of project activities and recipe, the model used the first two variables Z and X for path and activity inclusion respectively. The Z_{ge} binary decision variable to include or exclude each alternative path e (1 to E) for each group of alternative paths g (1 to G) in the project, and accordingly the X_i binary decision variable is used to include or exclude each activity i (1 to N), depending on the selection of the path:

Path inclusion: $Z_{ge} \in \{0, 1\}$ $\forall$ g = 1, …, G; e = 1, …, E (4.1)

Activity inclusion: $X_i \in \{0, 1\}$ $\forall$ i = 1, …, N (4.2)

Decision Variable for Activity Mode and Crashing Option: two dimensional binary Y_{ikq}



Example activity time-cost spectrum with three modes

For each activity *i,* set of **alternative modes** (Mode 1 is the cheapest and slowest mode). Each mode set of **alternative crashing strategies** (strategy 1 is the Normal without crashing); One mode and crashing strategy must be selected:

Mode *K*	Crashing Strategy *q*	Duration	Cost	Cost Slope CS_{ikq}	**Binary Variable Y_{ikq}**
	1 Normal	40 d	$65,400	---	$Y_{i11} = 0$
K = 1	***2 Overtime***	***(40-35) d***	***$65,400***	***$962 / d***	***$Y_{i12} = 1$***
	3 OT+ OM	(34-32) d	$70,900	$527 / d	$Y_{i13} = 0$
	4 Overmanning	(40-35) d	$65,400	$470 / d	$Y_{i14} = 0$
K = 2	1 Normal	30 d	$75,700	---	$Y_{i21} = 0$
	2 Overtime	*(30-25) d*	$75,700	*$962 / d*	***$Y_{i22} = 0$***
	3 Overmanning	(30-24) d	$75,700	$527 / d	$Y_{i23} = 0$
	4 OM+ OT	(23-20) d	$80,700	$470 / d	$Y_{i24} = 0$
K = 3	1 Normal	12 d	$103,300	---	$Y_{i31} = 0$

Example data for activity *i* with three optional modes. Example decision is to use mode 1 and crashing option: Overtime (shaded row)

Figure 4.2 Representation of Y variable

As such, all activities in the network will have their $X_i = 1$, except for the activities on an excluded path. Figure 4.1 represent Z and X decision variables. The third variable (as shown in Figure 4.2) decides activity mode and crashing decisions as follows:

$$Y_{ikq} \in \{0, 1\} \qquad \forall\, i = 1, \ldots, N;\ k = 1, \ldots, K_i;\ q = 1, \ldots, Q_{ik} \qquad (4.3)$$

Where, Y_{ikq} is a binary decision variable of the activity mode and crashing option; N is the number of activities; K_i is the number of modes for activity i; Q_{ik} is the number of available crashing options (overtime, overmanning, etc.) for mode k of activity i.

The example in Figure 4.2 shows sample activity data and a decision $Y_{ikq} = 1$ to indicate using mode 1, with crashing strategy 2 (overtime). Accordingly, the activity duration and cost are determined, then modified according to the overlapping strategy. Thus, activity start and accelerated duration become:

Activity Scheduled Start: SS_i positive integer value $\forall\ i = 1, \ldots, N$ (4.4)

Activity accelerated duration: D_i positive integer value $\forall\ i = 1, \ldots, N$ (4.5)

In order to have the flexibility to implement different strategies individually or combined (as needed by the user), two more variables are defined as follows:

Binary variable of (not allowing/allowing) activity crashing: $U_i \in \{0, 1\}\ \forall\ i = 1, \ldots, N$ (4.6)

Binary variable of (not allowing/allowing) activity overlapping: $W_i \in \{0, 1\}\ \forall\ i = 1, \ldots, N$ (4.7)

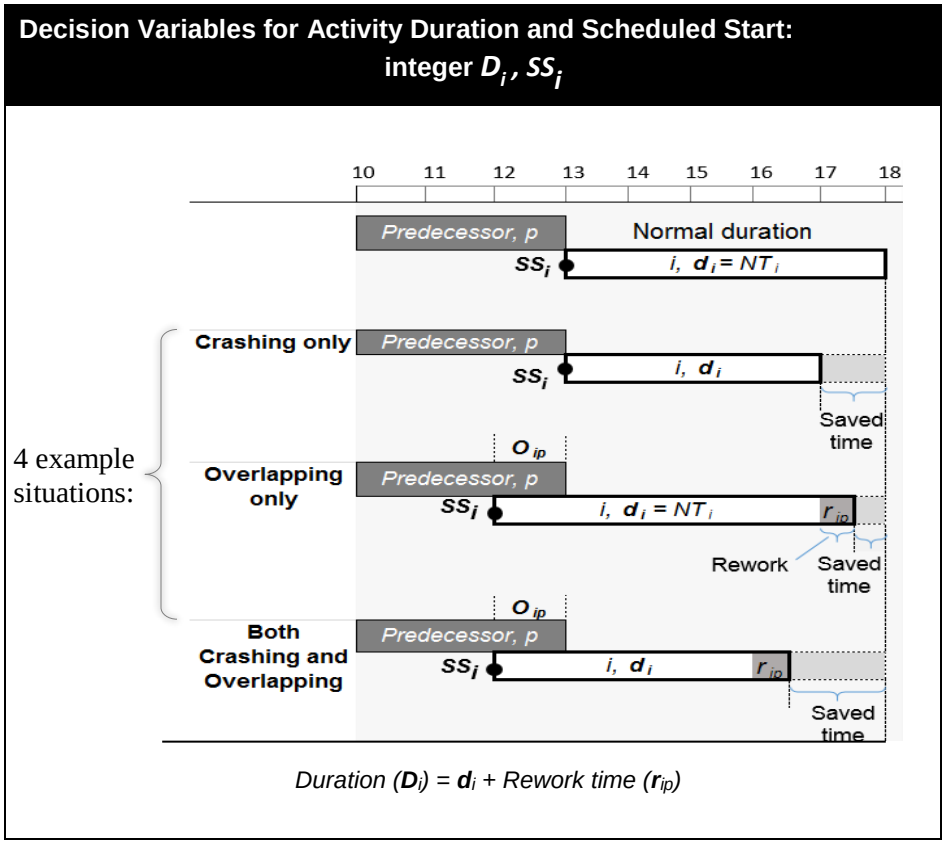


Figure 4.3 Representation of Activity Duration and Start Time Variables

4.2.2 Constraints

The set of constraints in the model are as follows:

4.2.2.1 Alternative Paths Constraints

Since only one of the alternative paths must be selected, the following constraint becomes necessary:

$$\sum_{g=1}^{G} \sum_{e=1}^{E} Z_{ge} = 1 \qquad (4.8)$$

Constraint on excluded activities: To ensure that all the activities on an excluded path e (its $Z_{ge} = 0$)

are excluded, the value of the decision variable X_i of the excluded activities on this path must be set to zero through the following constraint:

$$Z_{ge} - X_i = 0 \qquad \forall\, i \in \text{Activities of path e of group g ; e=1,…,E; g=1,…,G} \tag{4.9}$$

The same constraint in Eq. 4.8 applies to all paths. If a path is included (its $Z_{ge} = 1$), then the decision variables X_i of its included activities becomes 1's, thus the relationship in Eq. 4.9 still holds. Therefore, meeting this constraint ensures that included activities have their $X_i = 1$, while excluded activities have their $X_i = 0$. For all other activities that are compulsory, i.e. not on alternative paths, (its $X_i = 1$).

4.2.2.2 Activity Mode and Crashing Constraints

To consider activity mode selection, the model defines multiple modes *k* (from 1 to K_i modes) for each activity. Each activity mode has its linear time-cost spectrum that defines variety of crashing choices (q), from 1 to Q_{ik} (as shown in the example of Figure 4). Each portion of the spectrum has its cost slope (CS_{ikq}) that facilitate cost calculations. As such, activity data is defined using a tuple with parameters ($NT_{ikq}, NC_{ikq}, CT_{ikq}, CS_{ikq}, S_{ikq}, (r_{ik1}, r_{ik2}, \ldots\ldots, r_{ikL})$) which represent the normal time, normal cost, crash time, cost slope, overtime segment (per crashing day), and the amount of *l* resources (1 to L), respectively. This representation is powerful enough to enable the activity to have multiple modes and a piecewise-linear time-cost spectrum of each mode. To select a mode and a crashing strategy, the binary decision variable Y_{ikq} (0, 1) indicates which activity mode is selected and which crashing strategy is selected (e.g., example in Figure 4.2). To make sure that only one mode of construction is used for each activity, one constraint is needed for each activity to define that the sum of the binary variables for all modes, i.e., $\sum_{q=1}^{Q_{ik}} Y_{ikq}$must be equal to 1. For the excluded activities (their $X_i = 0$), on the other hand, no mode should be selected, thus the sum $\sum_{q=1}^{Q_{ik}} Y_{ikq}$ must be equal to

zero. Thus, a general activity constraint, which applies to any activity (included or excluded) can be defined as follows:

$$X_i \; - \sum_{k=1}^{K_i} \sum_{q=1}^{Q_{ik}} Y_{ikq} = 0 \qquad (4.10)$$

Based on the value of the X_i , and Y_{ikq} decision variables, the normal duration of activity i is calculated as follows:

$$NT_i = X_i \; . \sum_{k=1}^{K_i} \sum_{q=1}^{Q_{ik}} NT_{ikq} . Y_{ikq} \qquad \forall i = 1, \dots, N; \qquad (4.11)$$

Similar expressions in Eq. 4.11 are also used to determine the normal cost NC_i, crash duration CT_i, and cost slope CS_i.

4.2.2.3 Constraints on Correlated Modes

For practical decisions, the present model allows a pairwise correlation between the modes of two different activities$(i, i`)$, where the selection of a specific mode $k$ for activity $i$ necessitates the selection of mode $k`$ for activity i'. This has been defined through the following constraint:

$$\sum_{q=1}^{Q_{ik`}} Y_{i`k`} = \sum_{q=1}^{Q_{ik}} Y_{ik} \quad \text{or simply} \quad \sum_{q=1}^{Q_{ik`}} Y_{i`k`} - \sum_{q=1}^{Q_{ik}} Y_{ik} = 0 \qquad (4.12)$$

4.2.2.4 Flexible-Relation Constraints

To accommodate overlapping within the traditional CPM calculations, each flexible relationship between an activity i and its predecessor P_i must be defined using three inputs: Relationship type (e.g.,

FS); Lag time (Lag_{ip}); and the Minimum Lag (ML_{ip}). For the activities with hard lag relations, ML_{ip} is equal to the Lag_{ip}. For the flexible relations, the model incorporates modified CPM equations to calculate the activities' Scheduled Start times (SS_i) and Scheduled Finish times (SF_i). Having any type of relationship between activity i and its predecessor$\overline{P_i}$. Then, the following constraints become necessary:

Finish-to-Start: $$SS_i \geq X_i(SF_p + ML_{ip}.W_i + lag_{ip}.(1 - W_i)) \quad (4.13)$$
$$\forall p \in \overline{P_i} \quad i = 1, \dots, N;$$

Start-to-Start: $$SS_i \geq X_i(SS_p + ML_{ip}.W + lag_{ip}.(1 - W_i)) \quad (4.14)$$
$$\forall p \in \overline{P_i} \quad i = 1, \dots, N;$$

Finish-to-Finish: $$SF_i \geq X_i(SF_p + ML_{ip}.W_i + lag_{ip}.(1 - W_i)) \quad (4.16)$$
$$\forall\, p \in \overline{P_i} \quad i = 1, \dots, N;$$

Start-to-Finish $$SF_i \geq X_i(SS_p + ML_{ip}.W_i + lag_{ip}.(1 - W_i)) \quad (4.17)$$
$$\forall\, p \in \overline{P_i} \quad i = 1, \dots, N;$$

Where, the Scheduled Finish Time (SF_i) of an activity i is the sum of the scheduled start time (SS_i) plus modified activity duration D_i (defined later in Eq. 4.21). These equations are generic and can schedule the activities with multiple predecessors of various types. An example of CPM-based schedule calculations in case of multiple predecessor is shown in Figure 4.4.

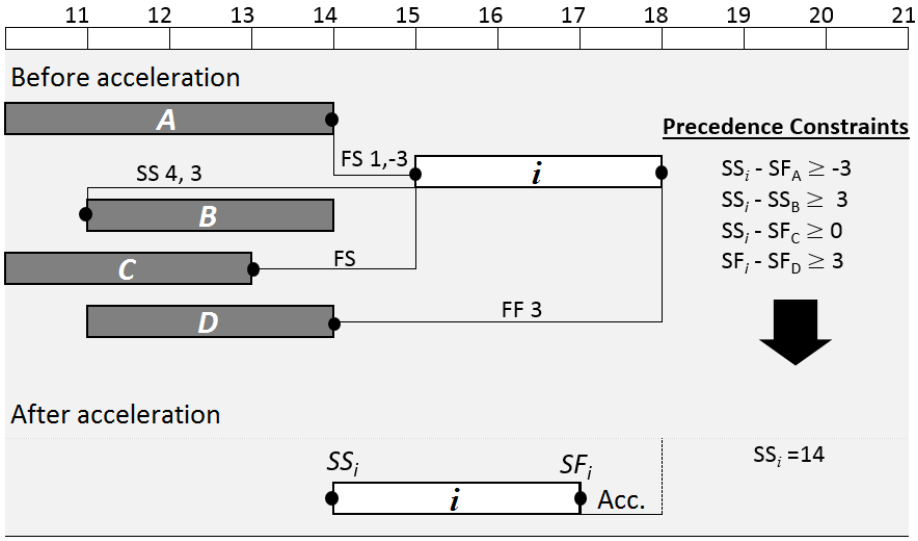


Figure 4.4 Precedence constraints in case of multiple precedence relationships

To determine the overlapping implications (rework time and cost) on the schedule, a detailed evaluation of the "additional" overlapping that a schedule exhibits, is required, as compared to the initial schedule. Thus, the scheduling process first evaluates an initial schedule using the same equations above but with all the U_i s and W_i s being zeroes. Accordingly, the Initial Start time (IS_i) and Initial Finish time (IF_i) for each activity are determined. These initial start and finish times are used to calculate the initial overlapping (IO_{ip}) between each activity i and its immediate predecessor p, as follows:

Initial Overlap $$IO_{ip} = max\ (0,\quad IF_p - IS_i\ ) \tag{4.18}$$

Then, during the schedule optimization process, the scheduled overlap is determined, as in Eq. 4.19, as follows:

Scheduled Overlap $$SO_{ip} = max\ (0, SF_p - SS_i\ ) \tag{4.19}$$

Accordingly, the additional overlap (O_{ip}) due to schedule acceleration is computed, as in Eq. 4.20 and an example is shown in Figure 4.5:

Additional Overlap $$O_{ip} = SO_{ip} - IO_{ip} \quad (4.20)$$

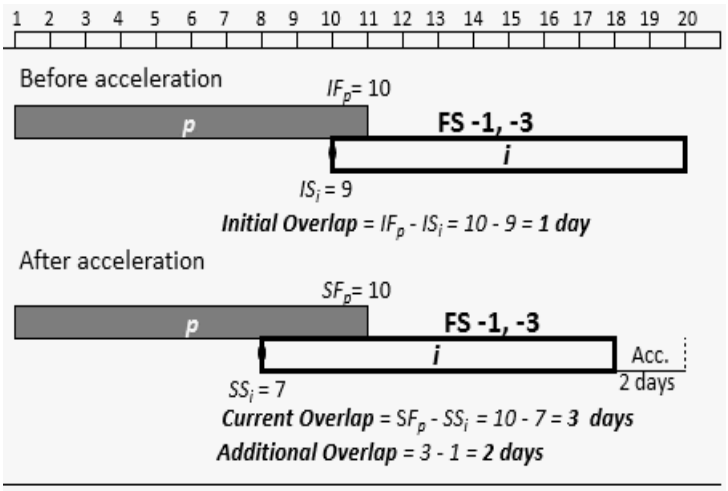


Figure 4.5 Calculation of additional overlaps

4.2.2.5 Activity Duration constraints

In the model, each activity *i* can be crashed so that its duration d_i can take any integer value, from its normal duration to its crash duration. Each duration value relates to a certain crashing strategy of either overtime, overmanning, or combination of both (as shown in Figure 4.2). Thus, constraints are needed to provide upper and lower bounds for activity duration, as follows:

$$NT_i \geq d_i \geq CT_i \qquad \forall i = 1, \dots, N; \quad (4.21)$$

Where, NT_i and CT_i are the normal and crash durations of the activity, respectively. Thus, when the activity exhibits no crashing (i.e., $U_i = 0$), the activity duration become ($d_i = NT_i$).

4.2.2.6 Rework constraints

As discussed earlier in the literature, overlapping causes the duration of the successor activity to be extended by the expected rework time (rw_{ip}), which is proportional to the additional overlap (O_{ip}) time between activity *i* and its predecessor *p***.** This direct relation multiplies the additional overlap time by a constant rate R_{ip}. As reported by Hazini et al. 2013, a reliable value of R_{ip} can be set through focus groups with experienced practitioners. Any other function, however, can be easily incorporated into the present model. To manage the cascade of reworks when an activity is overlapped with multiple predecessors, the model uses the sum of individual rework as the resultant amount of rework allotted to this activity (as in Cho and Eppinger 2005; Berthaut et al. 2011, Hazini et al. 2013, 2014), as follows:

$$rw_i = \sum_{p \in \overline{P_i}} rw_{ip} = \sum_{p \in \overline{P_i}} R_{ip} \, . O_{ip} \quad \forall\, p \in \overline{P_i};\ \forall\ i = 1, \dots, N; \tag{4.22}$$

Using this rework duration, the final activity duration (D_i) becomes as follows:

$$D_i = d_i + rw_i\,; \qquad SF_i = SS_i + D_i \qquad \forall\, i = 1, \dots, N; \tag{4.23}$$

4.2.2.7 Resource Constraints

As discuss in the literature section 2.5.2, the basic categories of resources in constructions are renewable, nonrenewable, and doubly constrained resources. Constraints are required to define the limits of these categories of resources for all activities. As typically done, the model assumes that activity resource demand is constant all over the activity duration and the rework extension. Accordingly, *for renewable resources*, constraints (one per resource) are expressed in the model such that the sum of the demands on a resource *l* by all eligible activities in each day (*t*, from day 1 to the

end of all tasks) must be within the resource availability limit. Assume Q_{lt} is the daily resource demand of renewable resource *l* and the resource constraint is:

$$Q_{lt} = \sum_{i} \sum_{k=1}^{K_i} \sum_{q=1}^{Q_{ik}} (r_{ikql} . Y_{ikq}) \leq [R_{lt}]$$

$$\forall\, i \in \textit{eligible activities in day } t;\ t = 1, \ldots, \textit{max SFi};\ l = 1, \ldots, L \quad (4.24)$$

Where R_{lt} is a two dimensional matrix $\begin{matrix} 1 & 2 & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & T \end{matrix}$

$$\begin{bmatrix} R_{11} & R_{12} & \ldots & \ldots & \ldots & \ldots & R_{1T} \\ R_{l1} & R_{l2} & \ldots & \ldots & \ldots & \ldots & R_{lT} \\ R_{L1} & R_{L2} & \ldots & \ldots & \ldots & \ldots & R_{LT} \end{bmatrix} \begin{matrix} 1 \\ \ldots \\ \ldots \\ L \end{matrix}$$

Where, r_{ikl} is amount of resource (*l*) required by combined decision of mode k and crashing strategy *q* of activity *i*. R_{lt} is two-dimensional array is used to specify resource availability of resource *l* each day , and *L* is number of renewable resources. In the case of *Nonrenewable resources*, the constraint is:

$$\sum_{i=1}^{N} \sum_{k}^{K_i} \sum_{q=1}^{Q_{ik}} (r_{ikql`} . Y_{ikq}) \leq R_{l`};$$

$$\forall i=1,\ldots,N; \quad k=1,\ldots,K_i; \quad l`=1,\ldots,L` \quad (4.25)$$

Where, $r_{ikl`}$ is amount of resource *l`* required by construction mode *k* of activity *i*, $R_{l`}$ is the amount of resource *l`* available, *L`* is number of nonrenewable resources. For *Doubly constrained resource*, as the total usage at every moment and total usage over the period of project duration are constrained, both Eqs 4.24 and 4.25 and must be met.

4.2.2.8 Optional integration constraints

Several optional constraints may be needed for practical considerations. For example, constraints have been added to represent the desire to avoid crashing and/or overlapping when a decision is made to change from the normal activity mode (k=1 and q=1) to any other mode (from 2 to K_i), as follows:

$$U_i + \sum_{k=2}^{K_i} Y_{ik1} \leq 1 \quad \text{and} \quad W_i + \sum_{k=2}^{K_i} Y_{ik1} \leq 1 \quad \forall\ i = 1, \dots, N; \tag{4.26}$$

From another perspective, as overlapping and crashing are not mutually exclusive, they can be used simultaneously, but this intensive compression strategy may decrease the activity flexibility during construction. In some situations, therefore, it may be necessary to avoid having both crashing and overlapping occurring simultaneously in one activity. Thus, a constraint may be used, as follows:

$$U_i + W_i \leq 1 \qquad \forall\ i = 1, \dots, N; \tag{4.27}$$

It is also possible to add a third optional constraint; that if both crashing and overlapping occur simultaneously in one activity, they occur in different activity segments, as shown in Figure 4.6 and suggested in the literature by Hazini (2013). In the figure, the second segment of the activity is crashed while the first and third segments overlap with other activities.

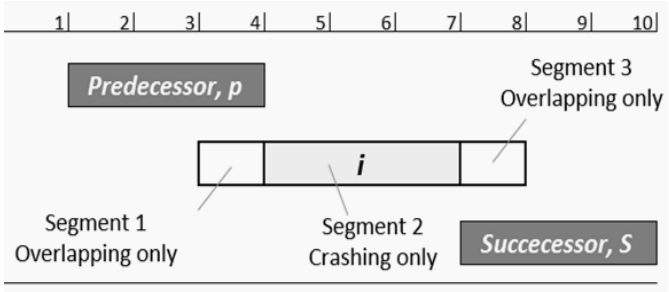


Figure 4.6 Avoiding multiple acceleration strategies in one activity segment

To consider for this case, a constraint is added to ensure that the activity duration is greater than or equal to the maximum overlap period between the activity and its predecessors (segment 1), plus the maximum overlap period between the activity and its successors (segment 3), plus the required time period to apply crashing (segment 2).

$$D_i \geq \left[\max_{p \in \overline{P_i}} O_{ip} + \sum_{k=1}^{K_i} \sum_{q=1}^{Q_{ik}} Y_{ikq} . \min(D_i, S_{ikq} . (NT - d_i)) + \max_{s \in \overline{S_i}} O_{iS} \right] \; ; \quad \forall \, i = 1, \dots, N; \qquad (4.28)$$

Where S_{ikq} is the overtime segment (per crashing day) required to apply crashing strategy q. For example, activity i (e.g., activity 5 in the case stud-3 discussed later) uses 3 overtime days to crash the activity for 1 day (i.e. $S_{i1q} = 3$ days). The use of activity segments in Figure 4.6 can possibly reduce rework and the stress on the work team by avoiding the excessive congestion due to overlapping and overmanning crashing at the same activity segment.

4.2.3 Objective functions and additional constraints

Based on the decision variables and basic constraints, three main expressions have been represented in the model (f_1 = Total Project Duration in Eq. 4.29; f_2 = Total Project Cost in Eqs. 4.30 to 4.34; and f_4 = Total Resource Variation in Eq. 4.36) to be used individually or combined as an objective function. One additional expression (f_3 = Acceleration Cost in Eq. 4.35) has been used in the model as alternative objective function in some cases. The function f_1, total project duration, is represented as the maximum schedule finish time SF_i among all activities as expressed as follows:

$$Project\ Duration\ (f_1) = \ max \ \ SF_i \qquad (4.29)$$

To incorporate the different optimization options when calculating the total project cost (f_2) during the optimization process, the variation in cost of alternative path and/or alternative activity mode in addition to the extra cost due to activity crashing and mode substitution is considered in the calculation of direct cost whereas the extra costs of overlapping is treated as part of the project penalties as follows:

$$Total\ \ Cost\ (f_2) = Direct\ Cost + Indirect\ Cost\ +\ Penalties - Incentives \qquad (4.30)$$

The first term in Eq. 4.30 is the summation of the activities' direct costs (i.e., sum of all normal costs $(NC_i)s$ plus all crashing costs) which is a function of activity duration and cost slope CC_i as in Eq. 4.31.

$$Direct\ costs\ =\ \sum_{i=1}^{N} NC_i +\ CS_i\ .\ (\ NT_i -\ d_i\ ) \qquad (4.31)$$

The second part in Eq. 4.30 is the total project indirect cost$(i.e., f_1\ *IC)$, which is a multiplication of project-duration (f_1, Eq. 4.29) by the daily indirect cost (IC, input to the model as a constant).

$$Indirect\ costs\ =\ f_1 * IC \qquad (4.1)$$

The third term relates to the summation of all penalties associated with missing the project deadline and/or any of predefined target dates for M milestones, plus the cost of activity overlapping as stated in Eq. 4.33 as follows:

$$\begin{aligned} Penalties &= Overlapping\ Penalties + Milestone\ penalities +\ Delay\ penalitie \qquad (4.33)\\ &= \sum_{i=1}^{N}\sum_{p\in\bar{P}_i} C_{ip}\,.\,R_{ip}\ .\,O_{ip} + \sum_{m=1}^{M} max\ (\,SF_m - T_m\ ,0) *\ P_m \\ &+\ max\ \ (f_1 - Deadline, 0).\,P \end{aligned}$$

Unlike crashing, there is no obvious extra cost of overlapping. Thus, the model assumes that it has a direct relationship (C_{ip}, the overlapping cost function) to the rework extension which in turn has a relationship R_{ip} with additional overlapping time O_{ip} between i and its immediate predecessor p. Furthermore, is noted that in Eq. 4.33, any scheduled delay (PD - Deadline) beyond the deadline is multiplied by a daily project penalty (P). The "max" function in the equation ensures that when the (PD - Deadline) delay is negative, then a zero is used to assign no penalty. Similarly, any delay in completing any milestone m beyond its target date T_m is evaluated as (SF_m - T_m) and is multiplied by a milestone penalty per day (P_m). In a similar manner, Eq. 4.33 calculates any project time saving (Deadline - PD) and multiplies it by a daily project incentive amount (I). Additionally, any time saving in completing any milestone m is evaluated as (T_m - SF_m) and is multiplied by a milestone incentive per day (I_m). The last term of Eq. 4.30 also relates to the summation of all the incentives for meeting the target project duration and the target dates for M milestones, as calculated in Eq. 4.34.

$$Incentives = max\ (Deadline\ - f_1, 0)\ .I + \sum_{m=1}^{M} max(T_m - SF_m, 0) * I_m \qquad (4.34)$$

Where, (I) is the daily incentive (bonus) amount for every day saved by completing the project before a preset deadline; and (P) is the daily penalty (or liquidated damage) amount for every day beyond the deadline. Based on these equations, the total cost of schedule acceleration f_3 is expressed, in terms of crashing, overlapping, mode, and path substitution as:

$$Acceleration\ Cost\ (f_3) = Crashing\ Costs + Overlapping\ Penalties + Substitution\ Cost$$

$$= \sum_{i=1}^{N} CS_i\ .(\ NT_i - d_i\ ) + \sum_{i=1}^{N} \sum_{p \in \overline{p_i}} C_{ip}.R_{ip}\ .O_{ip} + \sum_{i=1}^{N} (NC_i - NC_{i11}) \qquad (4.35)$$

4.2.3.1 Multi-objective functions

As discussed earlier, expressions of (f_1 = Total Project Duration, f_2 = Total Project Cost, and f_4 = Total Resource Variation) are to be used individually or combined as an objective function. In case of minimizing Total Project Cost, an additional constraint is needed to restrict the Project Duration to the deadline. The various options in formulating the objective function are presented in Table 4.1 with additional constraint to suit each option.

Table 4.1 Objective function options

Option	Objective Function	Additional Constraint(s)
1	Minimize Total Project Cost f_2	Project Duration $f_1 \leq Deadline$
2	Minimize Project Duration f_1	Total Cost $f_2 \leq Given\ Buget$
4	Multi-Objective (f_1 , f_2) a - Single-Weighted objective function b - staticLex Function* c - Two-step optimization	Project Duration $f_1 \leq Deadline$ Total cost $f_2 \leq Given\ Buget$

* Internal multi-objective function in the IBM ILOG CPLEX-CP optimization software

The developed model proposes three different ways to deal with the multi-objective optimization problems that involves a set of objective functions ($f_1, f_2, \ldots\ldots, f_n$):

a. **Single-weighted function:** The global objective function is the sum of the different objective functions multiplied by their respective weight value of objective weights depending on the project managers' preferences (e. g. objective function is minimize $w_1.f_1 + w_2.f_2$);

b. **staticLex function:** Defines a multi-criteria optimization used by CP optimizer, to order the different objectives. The last objective is the least important one. The CP optimizer will improve both objectives at the same time but will improve f_2 only if it does not degrade f_1 and in turn any improvement of f_1 is worth any loss on $f_2$and so forth (e. g. objective function is minimize staticLex (f_1, f_2));

c. **Two-Step optimization:** The optimization problem can be solved in *n* successive steps, where n is the number of objectives. First minimize objective f_1 to produce a solution, then add a constraint to avoid deteriorating f_1 and solve the problem with objective $f_2$using the first solution as a starting point to produce the second solution (e. g. objective function first is minimize f_1, then objective function first is minimize f_2)

4.3 Extended Formulation for Dynamic Schedule Optimization

This section extends the preconstruction schedule optimization model presented in the previous section to a comprehensive dynamic schedule optimization model to optimize the scheduling decisions at any reporting period r. These extensions generate an adaptable scheduling model for scheduling the remaining tasks, with the ability to tradeoff between the cost of corrective actions and the user/project manager's preferences. These extensions also mandate adding more inputs and constraints, and reformulating some of the objective functions. In the model formulation, indices $(\,)^{S_{r-1}}$ and $(\,)^{S_r}$ are used to denote the data of the baseline of previous reporting period(S_{r-1}), and new schedule(S_r), outputs respectively.

4.3.1 Additional inputs

As such, activity data tuple will be extended to include the baseline schedule (S_{r-1}) parameters. The new tuple($(NT_{ikq}, NC_{ikq}, CT_{ikq}, CS_{ikq}, S_{ikq}, (r_{ik1}, r_{ik2}, \ldots\ldots, r_{ikL}),\ ST_i^{S_{r-1}}$). The new added parameters to the tuple are the activity's Start Time of the baseline schedule(S_{r-1}). The other required parameters express the preferences of the user are the length of each reporting period (R), and the selected scenario of corrective-action period: "No preference", "Short-term corrective-action scenario", and "Long-term corrective-action scenario" as explained in Section 3.5.2. In addition to the list of the activities that belongs to the: "Flexible", "Possible but not desirable", and "Inflexible" as explained in Section 3.5.2.

4.3.2 Additional Constraints

As there is no interruption in activity execution, the Start Time of all ongoing Activities equals the reporting date (T_r).

$$SS_i^{S_r} = T_r \qquad \forall\ i\ \in Set\ of\ ongoing\ activities \qquad (4.37)$$

Another constraint is required to ensure the corrective-action planning cycle is starting from the end of reporting period r by the date of updating T_r

$$\min SS_i^{S_r} = T_r \qquad \forall\ i = 1, \ldots, N \qquad (4.38)$$

Being at the mid of construction also mandates other important constraints that are activity-specific such as activities that can start/end at the date given in the constraint, before, or later. The purpose is to enable the execution of activities within their preferred time-slots, resulting in a minimal loss of quality due to schedule acceleration.

4.3.3 Project Managers' Preferences

As discussed in Section 3.5.4, the developed optimization formulation will consider two dimensions of project manager's preferences:

- Their basic preference regarding the unwanted periods of deviations between the project baseline and the recovery plan, which will be considered through the *new objective function* to minimize the weighted schedule variations; and
- Their preference regarding the flexibility to change the activities' mode during construction, which will be considered through imposing *additional constraints to fix/release* the activities modes during construction.

New objective function: To maximize the satisfaction of the project manager, the objective function is formulated to minimize schedule changes and variations weighted by the selected preference function according to the selected scenario. The schedule variations is defined as the summation of the activities' start-times deviations between the new schedule and the baseline schedule, plus the value "1" in case of the activities' changes their modes in the baseline schedule as illustrated in Eq. 4.39.

$f_5 =$ Schedule deviations weighted by the preference function of the selected scenario

$$f_5 = \min.\left(\sum_{i=1}^{N} \delta_i . X_i .\ abs\,(SS_i^{S_r} - SS_i^{S_{r-1}}) + \sum_{i=1}^{N} \delta_i\ . X_i .\ (1 - Y_{i11}^{S_{r-1}} - Y_{i12}^{S_{r-1}})\right) \quad (4.39)$$

Where

$$\delta_i = \begin{cases} 1 & Scenario\ (1)\ No\ Preferences \\ {}^{r_i}/_{R} & Scenario\ (2)\ Short\ term\ Actions \\ 1 - {}^{r_i}/_{R} & Scenario\ (3)\ Long\ term\ Actions \end{cases} \quad \forall\ i = 1, \dots, N$$

δ_i is the preference weight of each activity according to which reporting period (r_i) the activity *i* starts and R is the number of reporting periods. The weighting procedure applies weights depending on the selected scenario. As an illustrative example, for Scenario-2, increasing linear weights (from 0 to 1) is used in periods of increasing distance in the future over the entire project span. A small value (${}^{r_i}/_{R}$) places decreasing weight on the schedule deviations in the near future, while a large value of (${}^{r_i}/_{R}$) close to 1 provides more preference for unwanted schedule changes in the far future flexibility. It is important to mention that the model is able also to incorporate step and nonlinear satisfaction functions. Additional experiments could be run to improve the results by considering multi-objective function of Minimizing Total Schedule Deviations(f_5), and Total Acceleration Cost(f_3).

Additional constraints: To consider the project managers' preferences in changing the methods of activities after starting the real construction, additional constraints are added to the model. The list of activities of each group of "G_1: Flexible"; "G_2: Possible but not desirable"; and" G_3: Inflexible" are inputs to the model.

A parameter γ_i will be assigned to each activity *i* either to force the mode of the activity to the initial mode (if $\gamma_i = 0$) or being released during the optimization (if $\gamma_i = 1$) as shown in Eq. 4.40 based on in which group is the activity and in which round to find a feasible solution.

Fixing Activity mode:

$$\gamma_i^G \cdot \left(\sum_{k=1}^{K_i} \sum_{q=1}^{Q_{ik}} Y_{ikq} \right) + (1 - \gamma_i \ ) . Y_{i11} = 1 \qquad \forall \ G = 1 \text{ to } 3; \ \forall \ i = 1, \dots, N \qquad (4.40)$$

$$\text{Where} \quad \gamma_i^G = \begin{cases} 0 & Activity\ i\ constrained\ group \\ 1 & activity\ i\ released\ group \end{cases}$$

Three rounds of optimization experiments are performed as schematically shown in Figure 4.7 until finding feasible solution.

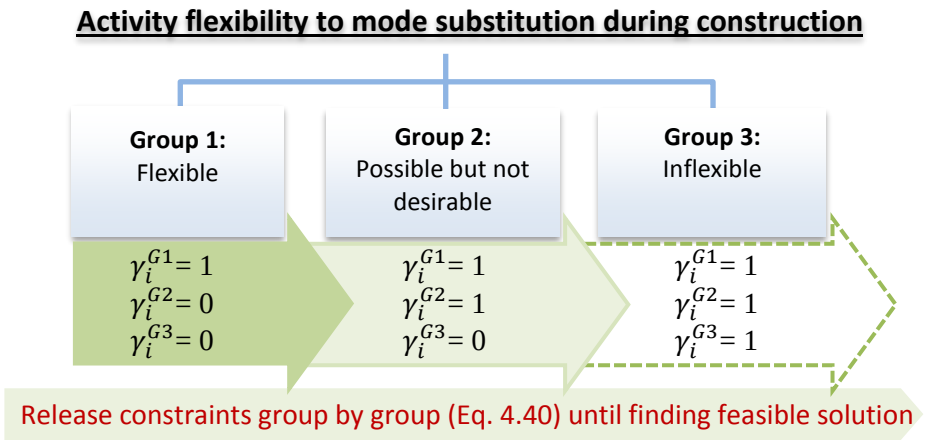


Figure 4.7 Three activity groups considering flexibility to mode substitution

The incorporation of the additional inputs, constraints, and the new objective functions an alternative approach to the rescheduling problem to generate a new schedule that matches up the original schedule depending on the project manager's preferences.

4.4 Model Implementation

In this work, IBM ILOG Optimization Studio is used to model and solve the proposed comprehensive scheduling optimization model using constraint programming technique. The latest version of IBM ILOG Optimization Studio (version 12.7) and OPL language are used in this research to model and solve the proposed scheduling optimization models using constraint programming techniques. The mathematical formulation has been coded in the Model-Editor of the IBM ILOG CPLEX Optimization Studio and ready for execution by CPLEX-CP solver. To facilitate inputs and outputs of the model, the author established links to pass data automatically among Microsoft Project, Microsoft Excel, and the Model Editor of IBM ILOG CPLEX-CP optimizer as schematically shown in Figure 4.8.

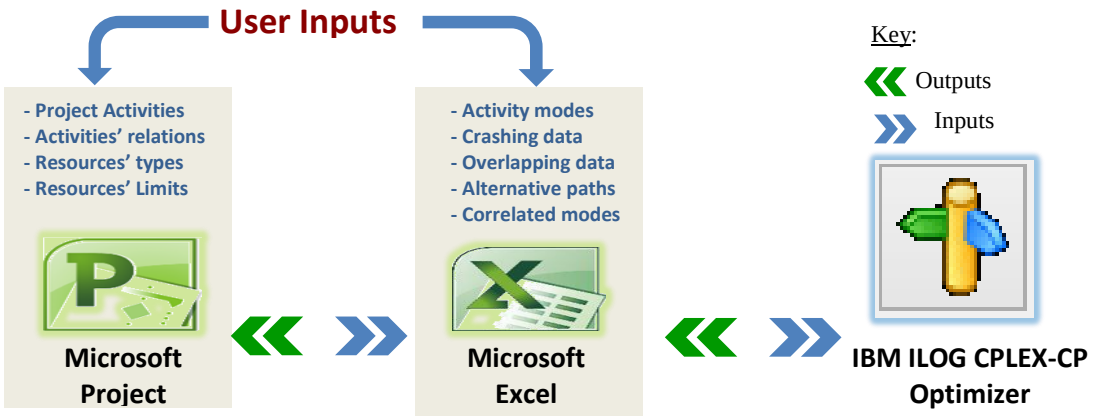


Figure 4.8 Automated CP environment for model implementing

As data preparation for various projects and their corresponding activities and modes of construction within each activity is much more convenient in a spreadsheet, the author took the advantage of fact that CPLEX-CP optimizer has the option to define a connection to retrieve the data to/from a spreadsheet. Likewise, Microsoft Project, which is widely used by practitioners to perform the schedule network calculations and determining the overall project duration, is easily linked with

Microsoft Excel to pass data between the two programs. Therefore, an Excel spreadsheet is programmed as shown in Figure 4.9 to automatically receive the required inputs from Microsoft Project about the project network and resources' data, or it can be entered directly in the spreadsheet by the user. The platform of all computational experiments is personal computer "Intel(R) Core i5-4210U CPU: Dell with a 1.70 GHz processor".

The user is then required to fill the spreadsheet with the remaining inputs of activity modes, crashing data, overlapping data, alternative paths, correlated modes as well as the constraints of deadline, budget and intermediate milestones as shown in Figure 4.9 and Figure 4.10; and for the extended model, the required data regarding progress as shown in Figure 4.11. All mathematical calculations are completed and the results are stored in the spreadsheet to form the inputs of the CPLEX-CP. Once the coded project of the proposed model in the CPLEX-CP Studio environment (shown in Figure 4.12) is executed, the solution results are automatically written back to the same spreadsheet (Figure 4.13) to produce the optimized project schedule in Microsoft Project (Figure 4.14). The automation of the model implementation process has enabled the author to perform various experiments on different case studies as shown in Chapter 5.

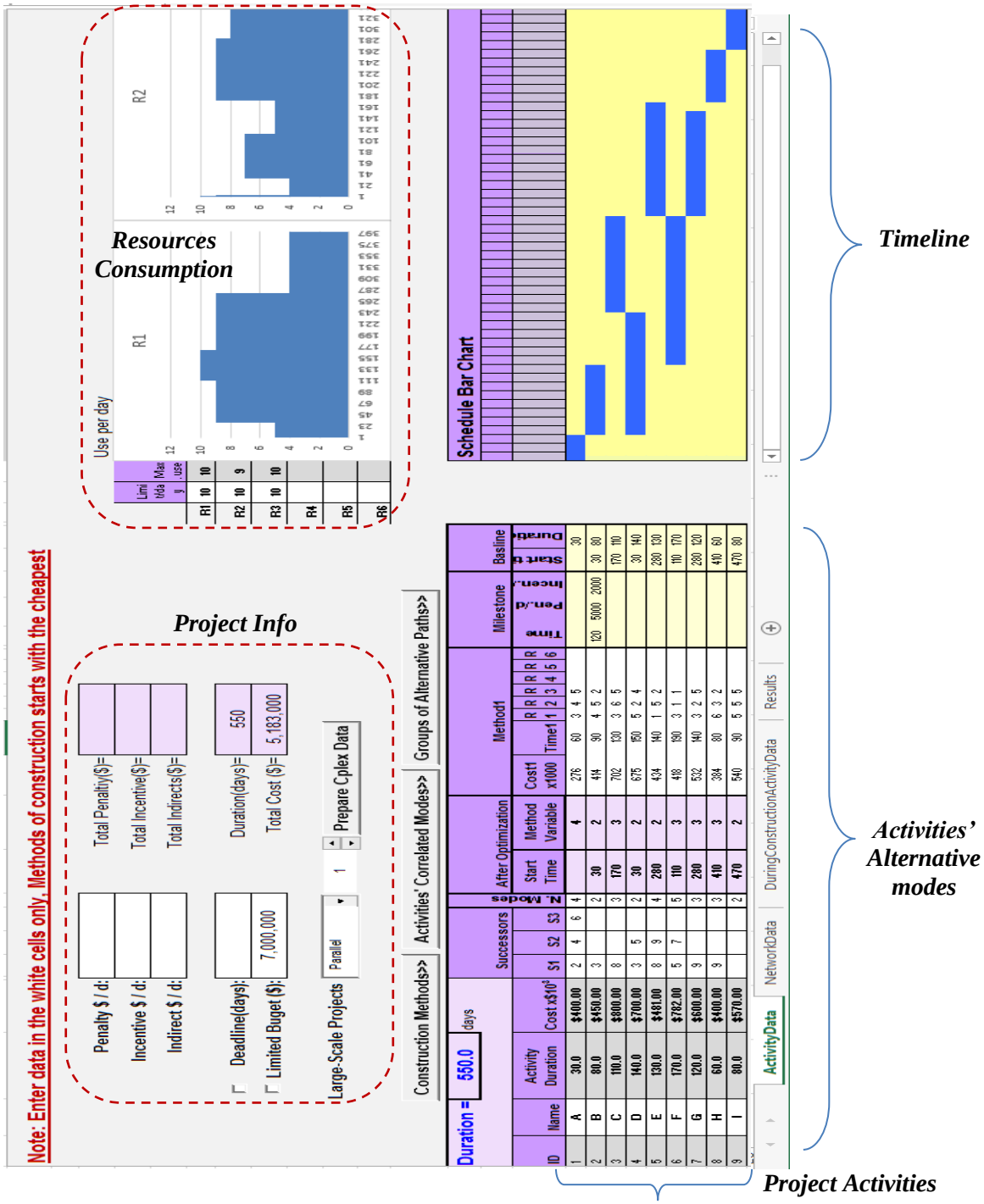


Figure 4.9 Snapshot of the programmed Microsoft Excel Spreadsheet: Activity Data

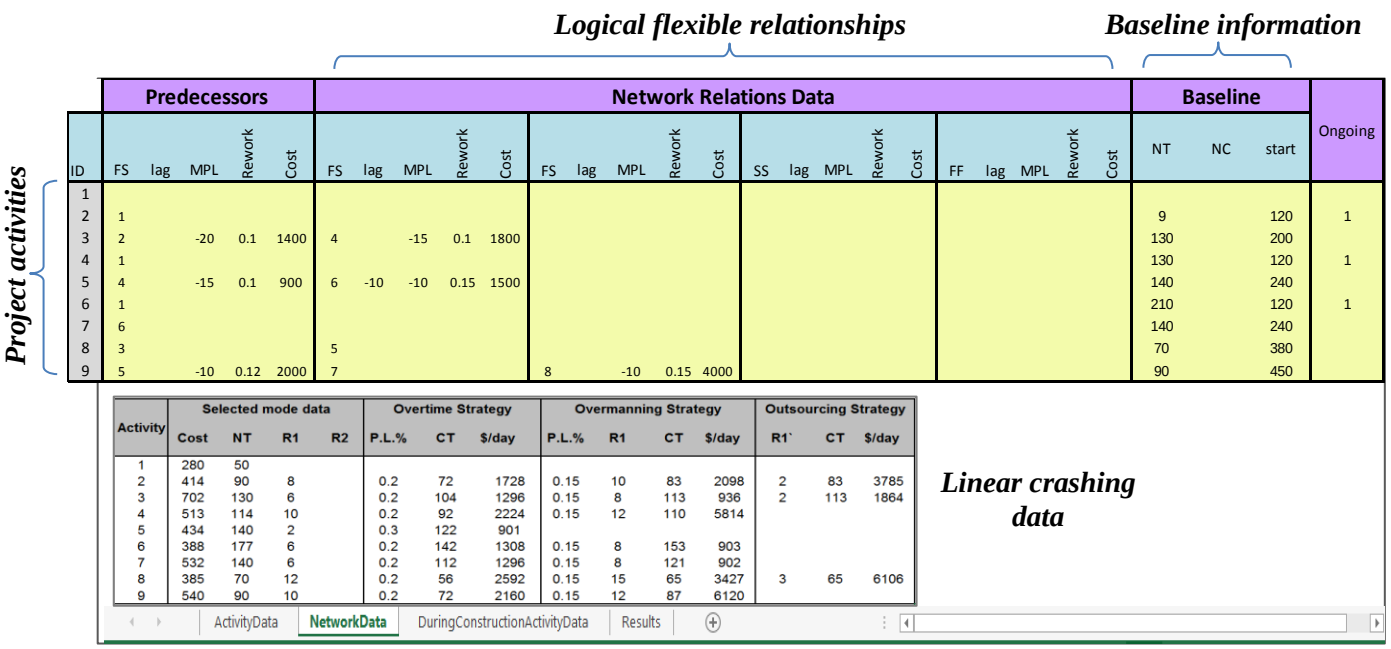


Figure 4.10 Snapshot of the programmed Microsoft Excel Spreadsheet: Network Data

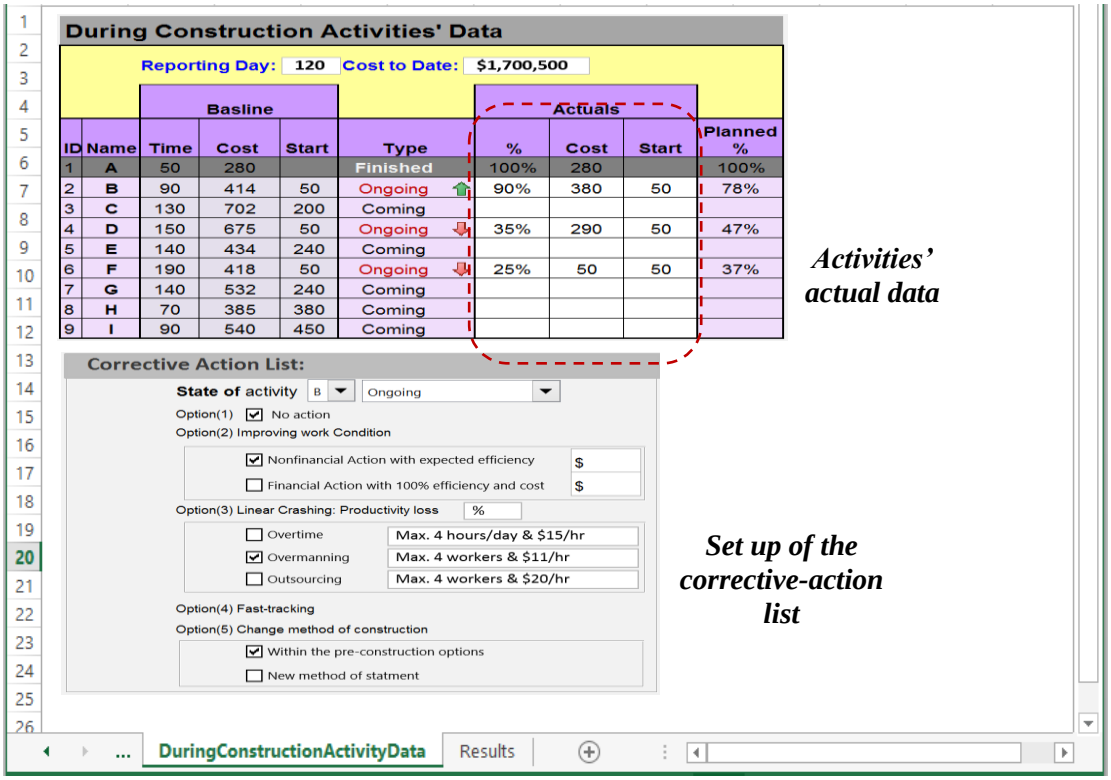


Figure 4.11 Snapshot of the programmed Spreadsheet: Data during Construction

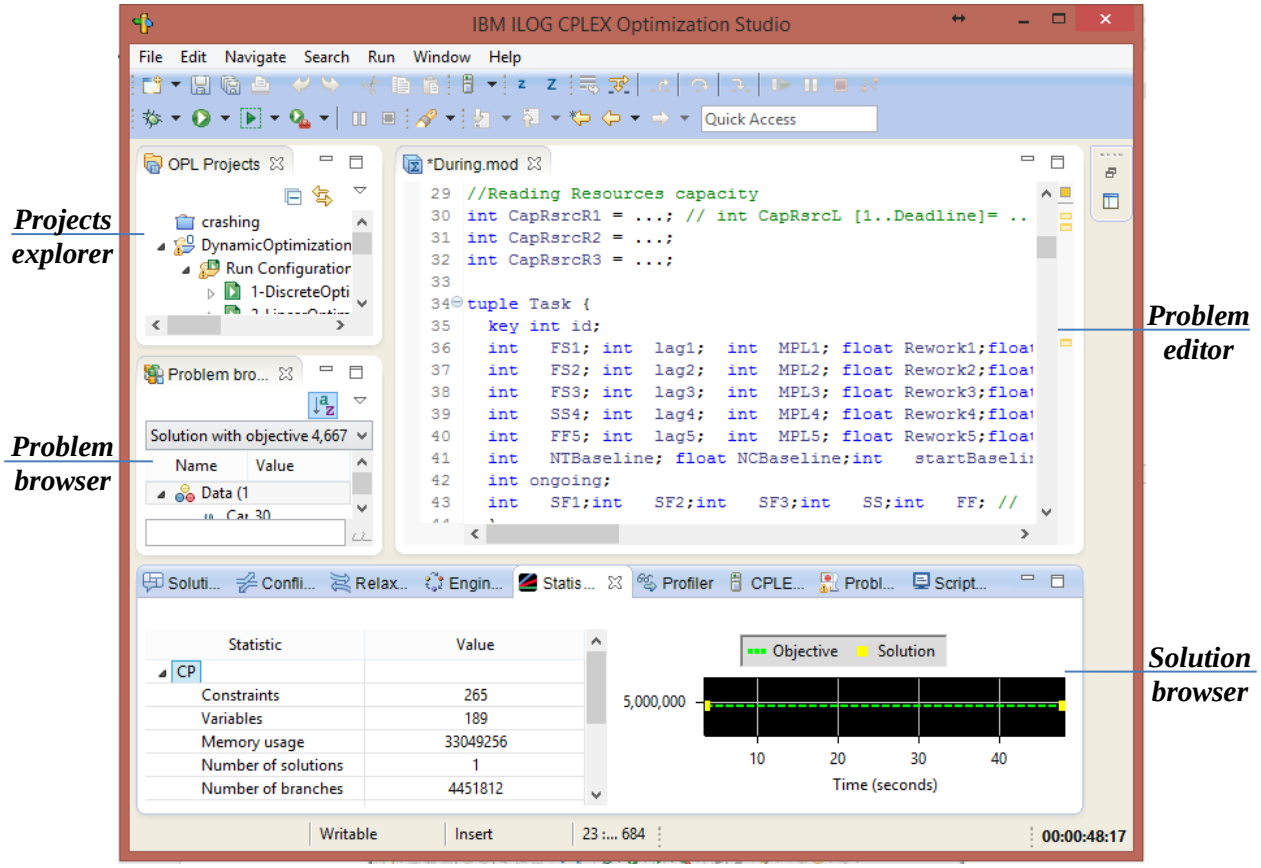
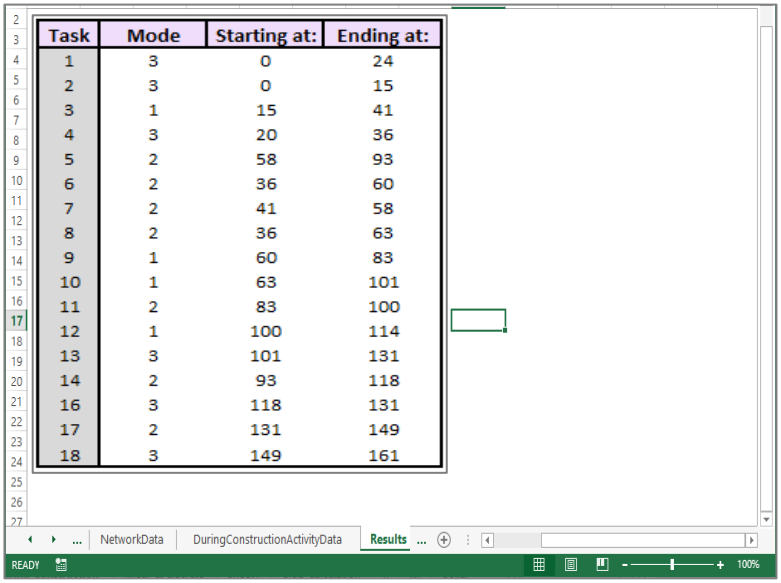


Figure 4.12 IBM ILOG Optimization Studio IDE



Task	Mode	Starting at:	Ending at:
1	3	0	24
2	3	0	15
3	1	15	41
4	3	20	36
5	2	58	93
6	2	36	60
7	2	41	58
8	2	36	63
9	1	60	83
10	1	63	101
11	2	83	100
12	1	100	114
13	3	101	131
14	2	93	118
16	3	118	131
17	2	131	149
18	3	149	161

NetworkData DuringConstructionActivityData Results
READY 100%

Figure 4.13 Snapshot of the programmed Microsoft Excel Spreadsheet: Results

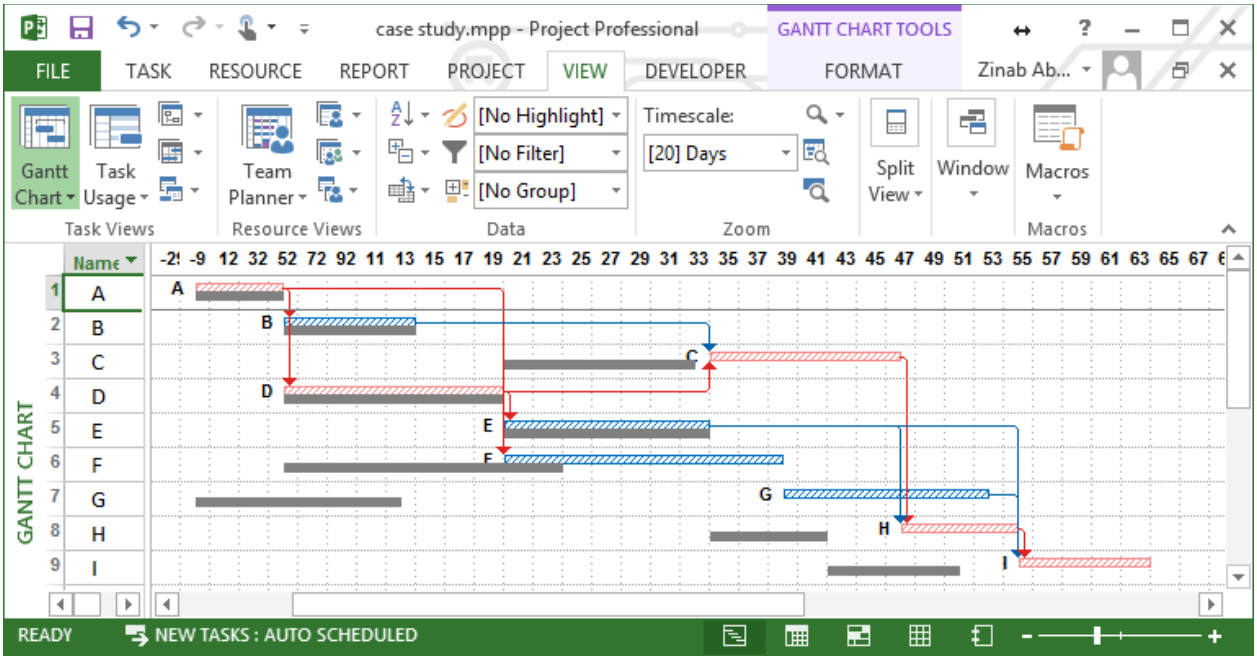


Figure 4.14 Snapshot of the Microsoft Project: Optimized Schedule

4.5 Summary

This chapter has presented the mathematical formulation of the preconstruction optimization model and the extended formulation to overcome the challenges of schedule optimization during construction through generic detailed equations that can be formulated using any optimization method. The presented formulation determines the optimum procedures for project execution options or corrective-action plans that combine optional paths, crashing, mode substitution, and limited resource allocation to satisfy objective functions of single, and multiple objectives, and meet practical constraints of mode-correlation; and strategies integration. The model is extended through the use of satisfaction functions to integrate project manager's preferences. The fully automated CP environment for model implemention is a step toward the adoption of proposed schedule optimization features within commercial scheduling systems.

Chapter 5

Validation Case Studies

5.1 Introduction

This chapter presents four case studies to demonstrate the effectiveness of the proposed framework along the various project lifecycles. The case studies have been chosen to illustrate the essential features of the model along the distinctive stages of the project: work-packaging at the early planning, schedule compression before construction, and corrective-action planning during construction. For simplicity, each case study test individual options and decisions that are presented in the framework to clarify the consistency of the model with the project targets and constraints. To validate the model and evaluate the framework performance, two of the case studies, from the literature, are used to compare their results with the current framework results.

5.2 Case study-1: Phase I- Preconstruction Planning

To demonstrate and test the model at the early planning stage, a small hypothetical project of 18 activities (Figure 5.1 and Table 5.1) was used for demonstration purposes, however, the model is expected to be able to handle realistic size projects (up to 2000 activities). The example project has a variety of activity relationships and a part of the project network has two alternative paths. The project has two renewable resources (R1, R2) and one non-renewable resource (R3). Each activity has three modes that vary from cheap-and- slow to fast-and-expensive, as shown in the table below Figure 5.1.

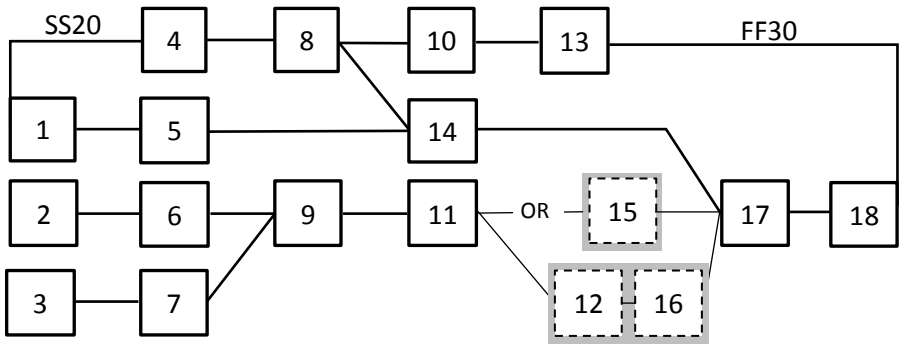


Figure 5.1 Network of case study-1

Table 5.1 case study-1 (activities' data)

Note: Costs are in $1,000s.

	Mode 1: Cheap & slow					Mode2					Mode3: Fast & expensive				
Activity	R1	R2	R3	Dur.	$Cost	R1	R2	R3	Dur.	$Cost	R1	R2	R3	Dur.	$Cost
1	3	3	0	34	27	5	4	0	28	31	9	8	0	24	35
2	2	5	3	24	20	0	6	4	20	26	2	8	3	15	32
3	3	4	4	26	35	5	6	3	22	40	5	8	3	18	44
4	12	3	2	23	60	18	3	3	19	65	20	3	4	16	62
5	10	4	5	38	80	12	5	6	35	85	15	6	4	30	92
6	16	6	0	27	100	18	7	0	24	105	20	10	0	21	110
7	14	4	4	19	8	14	6	3	17	11	14	8	3	15	14
8	7	0	3	30	35	2	2	3	27	40	5	2	4	22	45
9	9	5	3	23	55	15	0	4	19	60	15	3	2	17	64
10	12	2	6	38	80	18	4	4	34	85	20	6	5	32	89
11	0	8	3	20	40	4	6	4	17	45	6	6	2	15	48
12	6	0	6	14	35	0	8	4	12	38	4	6	5	11	41
13	2	5	0	38	120	0	8	0	33	130	2	8	0	30	136
14	10	2	3	30	60	14	4	3	25	68	14	4	4	20	72
15	15	0	3	34	115	10	5	4	24	130	8	3	3	20	160
16	12	5	4	20	95	15	6	2	16	90	20	5	3	13	87
17	17	3	0	22	200	20	5	0	18	232	22	6	0	14	240
18	0	3	5	20	42	8	2	6	15	45	10	5	5	12	48

5.2.1 Scheduling Experiment at early planning stage

At this stage, the planner's target is to determine the optimum mix of activities, their modes of construction, and their schedule to meet a 165- day deadline in addition to the resource limits, intermediate milestones and correlated modes specified in Table 5.2. The project duration, before

optimization, using the default (i.e. cheapest) modes for all activities mode is 179 days, top case in Figure 5.2, with a total cost of $1.463 million and the maximum usage of the R1, R2 and R3 being 52, 17, and 44 respectively. To examine the flexibility and results of the model, three successive sub-experiments were run and the results are shown in Figure 5. The successive experiments shows that each step improved the schedule in a manner that is consistent with its objective and satisfying the problem constraints.

Table 5.2 Constraints of Case study-1

Constraints
• Deadline= 165 days • Alternative paths: Path1 includes activity 15 & Path2 includes activities 12, 16 • Resource limits: (R1 = 34/ day, R2 = 18/ day and 54 of R3) • Intermediate Milestone: finish activity 14 on day 110 with bonus $1.000 per day saved and penalty $2,000 per day. • Correlated modes (Activity, mode): (6, 1) & (7, 1); (6, 2) & (7, 2); and (6, 3) & (7, 3).

In case-1 experiment of Figure 5.2, optimization was run with constraints of time and cost only. A new constraint is imposed in each of the successive experiments to show how the results are changed to meet the new constraints. For the first experiment, the optimum path of the project is selected (path 2 that includes activities 12 and 16). The model resolves the resource over-allocation and meets the project deadline with minimum cost of $ 1.467 million. In case-2 experiment, the total project cost increases to $1.477 million in order to meet the intermediate milestone of finishing activity 14 before day 110. The results of Case -3 experiment, shown in Figure 5.2, represents *the optimum decisions* that satisfy all constraints and considers correlated modes of activities 6 and 7 with the same total cost of $ 1.477 million but in less project duration (164 days). The results Case-3 experiment shows that

correlation did not necessitate additional cost or duration (The total project cost is 1.477 million for case-2 and case-3).

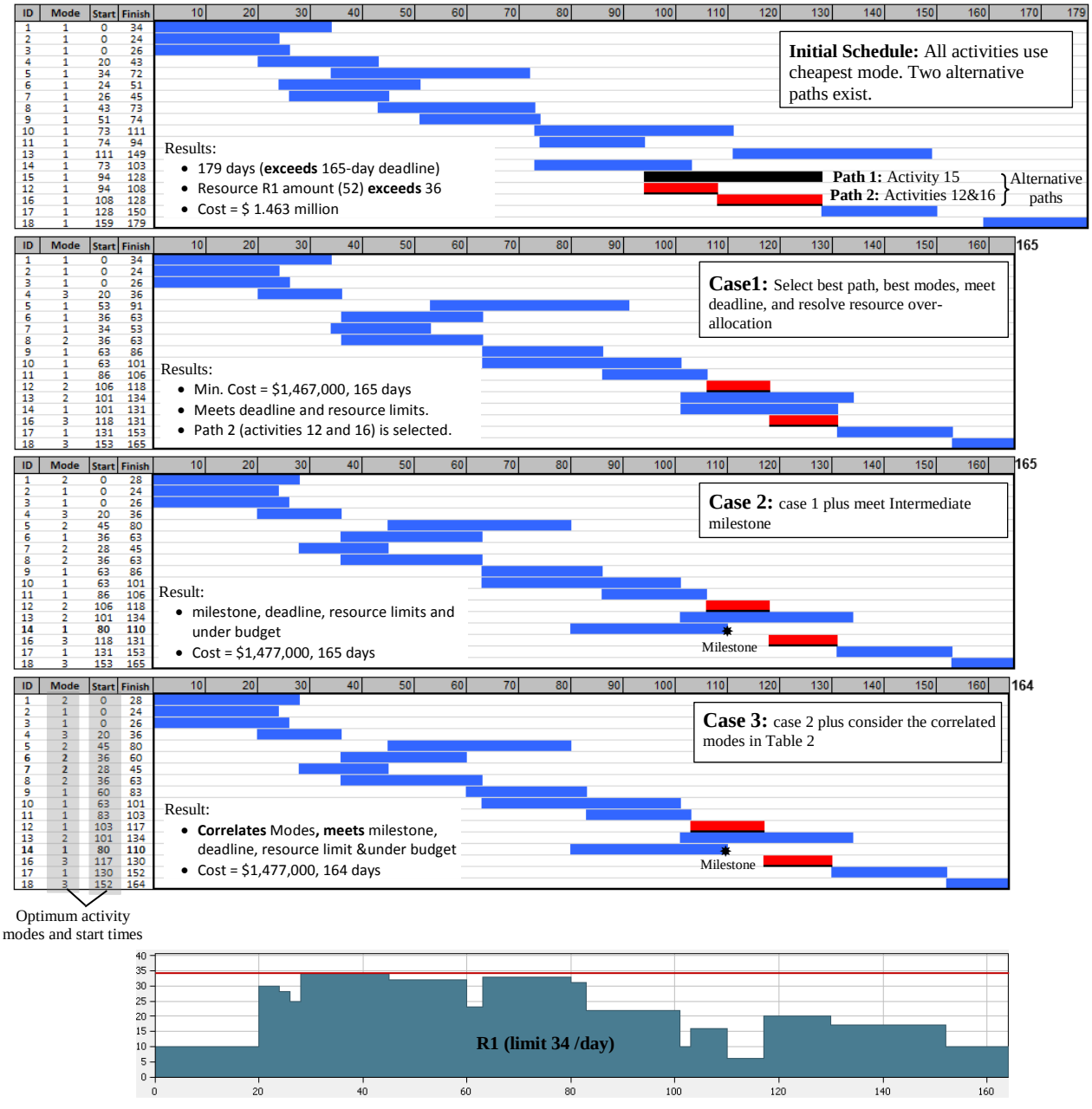


Figure 5.2 Results of Case study-1 experiments (early planning stage)

To further show the flexibility of model, an additional experiment (Figure 5.2) was run with the objective of minimizing the project duration. As shown in Figure 5.3, the optimum path is path 1 (activity 15, not the path with activities 12 and 16). The schedule meets all the milestone, correlation, and resource constraints, with a minimum duration of 155 days. The results of all these planning stage experiments prove that work packages and the final project network were optimally determined as a function of the objective function and constraints.

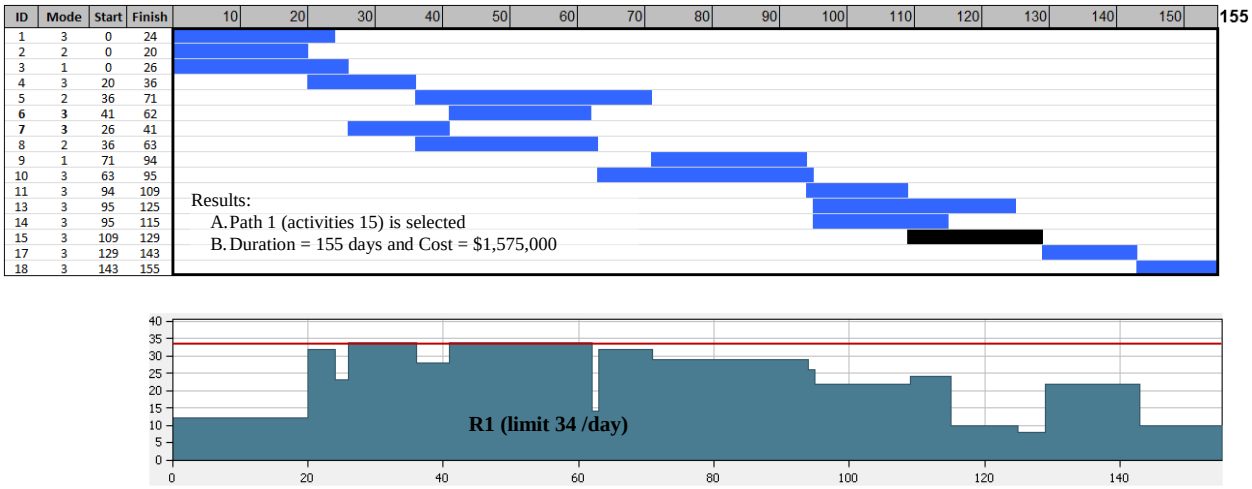


Figure 5.3 Case study-1 experiment with minimum duration

Multi-objective experiments: As discussed in section 4.2.3.1, three alternative ways to deal with the multi-objective optimization problems. To normalize the Single-weighted Function, minimum cost f_1 is divided by the limited budget and the least project duration f_2 is divided by project duration in addition to considering equal weights of time and cost The results of the multi-objective experiments performed are summarized in Table 5.3 and plotted in Figure 5.4 to illustrate the comparison of the results; which

shows that the two-step multi-objective optimization is able to get the minimum cost f_1 with the least project duration f_2 (point 2). This considers a generic trend that the N-step function surpasses the Single-weighted and the multi-criteria functions.

Table 5.3 Case study-1 results of multi-objective experiments

Experiment	Objective Function	Results	
		Cost	Duration
Focus on cost minimization			
1	Minimize total cost	$1.467	165 days
2	Two step (minimize cost, then duration)	$1.467	164 days
3	Min. staticLex**(total cost, project duration)	$1.472	164 days
4	Single weighted function***	$1.472	161 days
Focus on duration minimization			
5	staticLex (project duration, total cost)	$1.505	156 days
6	Two step (minimize duration, then cost)		
7	Minimum duration	$1.511	156 days

* All experiments are done only with basic constraints
** Internal function in the IBM ILOG CPLEX-CP optimization software
*** Normalized weights

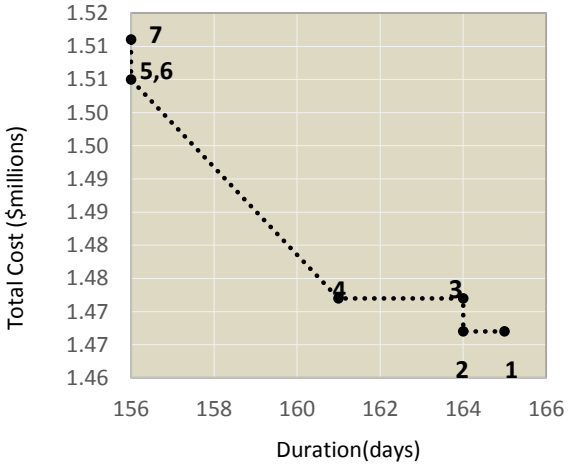


Figure 5.4 Case study-1 results of multi-objective experiments

5.2.2 Scheduling Experiment immediately before construction

In this case study, it is assumed that the experiment of case 3 (Figure 5.2) determined the work packages and the activity modes that are committed for the project before real construction. Later and immediately before construction, the deadline was revised to 150 days, under the same resource constraints. Table 5.4 shows the committed work packages, modes, and the optional crashing data related to the use of overtime, overmanning (with internal workers), or outsourcing (with external workers).

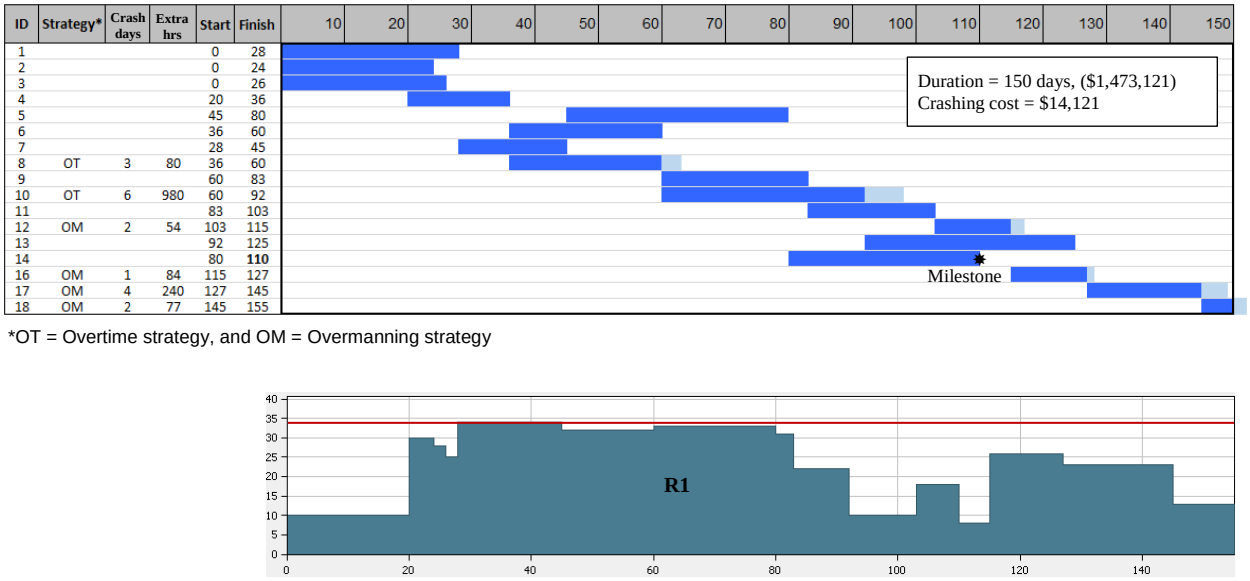
Table 5.4 Crashing data for Case study-1

No	Selected Mode			Overtime Strategy		Overmanning Strategy			Outsourcing Strategy		
	Normal cost	Normal duration	*r_1	Crash duration	Cost slope	r_1	Crash duration	Cost slope	r_1`	Crash duration	Cost slope
1	$31 K	28	5	22	580	7	22	164			
2	$20 K	24	2	19	237	3	18	56			
3	$35 K	26	3	20	330	4	22	137			
4	**$62 K**	**16**	**20**	**13**	**2560**	**26**	**14**	**1126**	**6**	**14**	**2963**
5	$85 K	35	12	27	1329	16	29	493	4	29	1478
6	$105 K	24	18	19	2131	23	21	1014	5	21	2667
7	$11 K	17	14	14	1876	18	15	838	4	15	2155
8	$40 K	27	2	21	226	3	20	54			
9	$55 K	23	9	18	1033	12	19	364			
10	$80 K	38	12	30	1410	16	31	459	4	31	1422
11	$40 K	20	0								
12	$35 K	14	6	11	696	8	12	296			
13	$130 K	33	0								
14	$60 K	30	10	23	1091	13	25	422	3	25	1251
16	$87 K	13	20	10	2200	26	11	915	6	11	2618
17	$225 K	22	17	17	1890	23	18	658	6	18	2029
18	$48 K	12	10	10	1400	13	10	422			

*r1 = required manpower and r_1` = outsourced r1 manpower

At this stage, the planner is required to meet the stricter deadline by determining the optimum crashing strategy considering all the other project constraints. Assuming that the normal working hours is 8 hr/day, maximum working hours is 12 hr/day, normal payment $11/hr, overtime payment $15/day, the

R1 is a manpower resource, overmanning will be provided either from the internally available workers (at the same rate) or outsourced at the rate of $18/hr. In addition, activities experience productivity loss equals 15% if an overtime is used over the whole activity duration. While the productivity loss in the case of overmanning (due to site congestion) is only 8%. An example of using Table 5.4 information is activity 4. First, for the overtime strategy, the maximum crashed duration (CD) = (1+ 0.15) x normal man-hours required / max. working hrs/day = (1+%15)*16*8/12= 13 days, and the cost slope (CS) is easily calculated considering the additional cost to save one day of activity duration. Similarly, the crash duration and cost slope of using overmanning or outsourcing are calculated. Upon entering the data of Table 5.4 and running the optimization, the results show that using a combination of crashing strategies, the revised deadline of 150 days was met, without changing the committed modes. Table 5.5 shows the crashed days of the affected activities and the required extra hours (overtime or overmanning) to perform activity crashing. For example, Activity 10 needs 980 overtime hours to reduce its duration 6 days. One possible way to perform this crashing strategy is to have all the 12 workers of the task spend 4 overtime hours for 2 days (12x4x21= 1,008 hrs). Likewise, activity 17, where the crashing of 4 days requires 240 overmanning hours, which can be achieved by acquiring 6 more workers for 5 days (6x8x5 = 240 hrs). These results show the flexibility of the model to incorporate the various strategies of crashing and consider the resource constraints. The proposed model was used also in performing various sensitivity analysis experiments to examine a combination of different crashing strategies (overtime, overmanning, and outsourcing) as summarized in Table 5.6. The results show that the minimum crashing cost ($14,121) is obtained when the model is permitted to combine the three strategies, while the crashing cost was $17,166 in case of using overtime only, and $31,899 in case of using outsourcing only. The model is unable to find a solution in the case of overmanning only.



*OT = Overtime strategy, and OM = Overmanning strategy

Figure 5.5 Results of Case study-1 experiment at the pre-construction stage

Table 5.5 Case study-1 experiment at the pre-construction stage
(Details of the optimum extended hours)

Activity	Crashed days	Optimum decision	Example strategy
8	3	80 overtime hours	
10	6	980 overtime hours	All workers (12) stay up extra 4 hrs for 21 days
12	2	54 overmanning hours	
16	1	84 overmanning hours	
17	4	240 overmanning hours	Acquiring 6 more workers for 5 days
18	2	77 overmanning hours	

Table 5.6 Case study-1 (crashing experiments)

Crashing Strategy	Crashing cost
All strategies	$14,121
Overtime only	$17,166
Outsourcing only	$31,899
Overmanning only	No possible solution

Comments on Case study-1 results:

- The model produced outstanding results with the case study of 18 activities under different types of constraints. The wide spectrum of options allowed in the model is not covered by any other work in the literature.
- The model has the flexibility to provide decision support at different project stages. The decision variables change from determining the optimum set of work packages at the bidding and early planning stage to determining the suitable crashing strategy and the detailed deployment of the crews and their overtime/overmanning hours needed.
- This model enables the planner to set the preferences among extending the hours for already available resources, versus utilizing any internally idle resources or outsourcing additional resources.
- The model enables the manager to meet the revised deadlines without disturbing the already committed modes. This feature is helpful especially in case the project requires early procurement of important subcontractors.
- The fast speed and the unique features of the model, particularly the alternative paths, mode correlation, and intermediate milestones, make the system practical and suitable for medium and large-size projects. These common decisions are difficult to meet using only intuitive guessing and experience. Projects can benefit from the presented decision support framework.

5.3 Case study-2: Phase I- Schedule Compression

Case study-2 focuses on experiments on schedule acceleration using a combination of activity crashing and overlapping. It is a small example project of 6 activities as shown in Figure 5.6 and Table 5.7. The figure shows the project network with both soft and hard relations, as well as overlapping and crashing

information. While the number of activities is small in this example, the various relationship types make it a comprehensive example to test the presented model. The rework columns define the rework time and cost associated with each additional overlapping day in the schedule. For example, if the schedule introduces a 10-day overlap between activity C and activity B, then the associated rework time and cost are 0.05 x 10 = 0.5 day and 0.05 x 10 x $1,200 = $600, respectively. The initial project duration is 140 days (all activities use normal durations). To test the proposed model, three experiments with different objective functions and constraints were carried out, as outlined in Table 5.8. In these experiments, no rework time was considered to void non-integer durations.

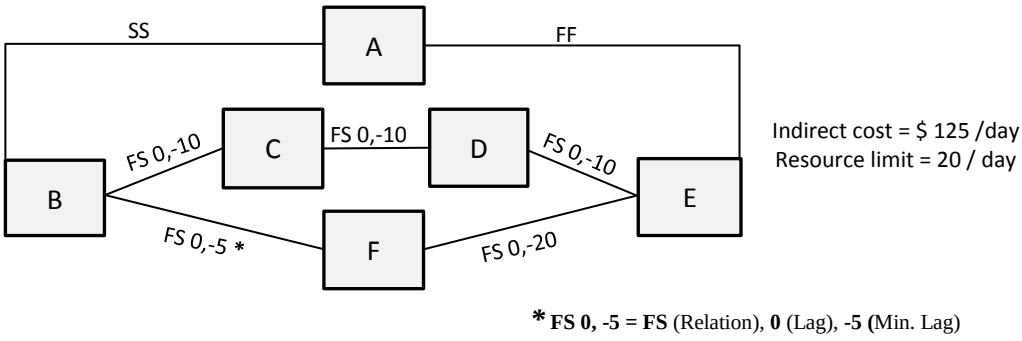


Figure 5.6 Network and activities' data for the Case study-2

Table 5.7 Activities' data for the Case study-2

Activity	Overlapping Data			Crashing Data				Resource
	Pred.	*Rework Time, R_{ip}*	*Rework Cost, C_{ip}*	Normal Duration	Normal Cost	Crash Duration	Cost slope ($/day)	Amount/ day
A	B	-	-	110	$12,000	100	$100	10
B	-	-	-	20	$1,800	15	$200	5
C	B	0.05	$1200	40	$16,000	30	$600	5
D	C	0.1	$1200	30	$1,400	20	$60	5
E	D	0.1	$1000	50	$3,600	40	$120	5
	F	0.1	$1,100					
	A	-	-					
F	B	0.05	$980	60	$13,500	45	$300	5

Comments on Case study-2 experiments:

The experiments' results are shown in Table 5.8 and Figure 5.7. Each experiment improved the schedule consistently with its objective. Comments on the results are as follows:

- In optimization experiment 1, the objective is to minimize the cost of accelerating the project from 140 days to 110 days. Four sub-experiments were carried out. Experiment 1.3 of allowing both crashing and overlapping resulted in the cheapest cost to reach 110 days, where the cost of acceleration reached only $3,195. This is compared to $7,300 in case of crashing only (experiment 1.1) or $4,695 in case of overlapping only (experiment 1.2). This shows that overlapping is cheaper than crashing and that a combination of crashing and overlapping can lead to the most cost-effective acceleration strategy. Remarkably, experiment 1.4 shows that at a slight extra acceleration cost $3,345, by allowing either overlapping or crashing, but not both, on any activity, which is a less stressful approach to acceleration.

- Experiment 2 targeted a dual objective to minimize both the project duration and the acceleration cost. Accordingly, the model was able to obtain a minimum duration of 100 days, with an acceleration cost of $6,245. Relaxing the schedule by allowing only one strategy for each activity resulted in 105 days at a little cheaper acceleration cost.

- Experiment 3, which targeted to minimize total project cost was also able to reach a minimum total cost of $65,020, which is the cheapest of all the experiments.

Table 5.8 Case study-2 results of optimization experiments

Exp.	Description	Project Duration	Project Cost	Acceleration Cost	Crash days	Overlap days
0	**Normal duration and cost before optimization**	140	$65,800	$0	0	0
1	**Minimum acceleration cost (i.e., min. f_3) and meet a deadline of 110 days, i.e., ($f_1 \leq$ 110 days)**					
1.1:	Only activity crashing is allowed.	110	$69,350	$7,300	35	0
	Only activity overlapping is allowed.	110	$66,745	$4,695	0	50
1.2:	Both crashing and overlapping are allowed.	110	$65,245	$3,195	20	20
1.3:	Both crashing and overlapping but not on the same activity.	110	$65,395	$3,345	25	10
1.4:						
2	**Minimum project duration and minimum acceleration cost (i.e., min (f_1, f_3)**					
2.1:	Both crashing and overlapping are allowed.	100	$67,045	$6,245	35	30
2.2:	Both crashing and overlapping but not on the same activity.	105	$67,645	$5,595	20	40
3	**Minimize total project cost (i.e., min. f_2)**	115	$65,020	$2,345	15	15

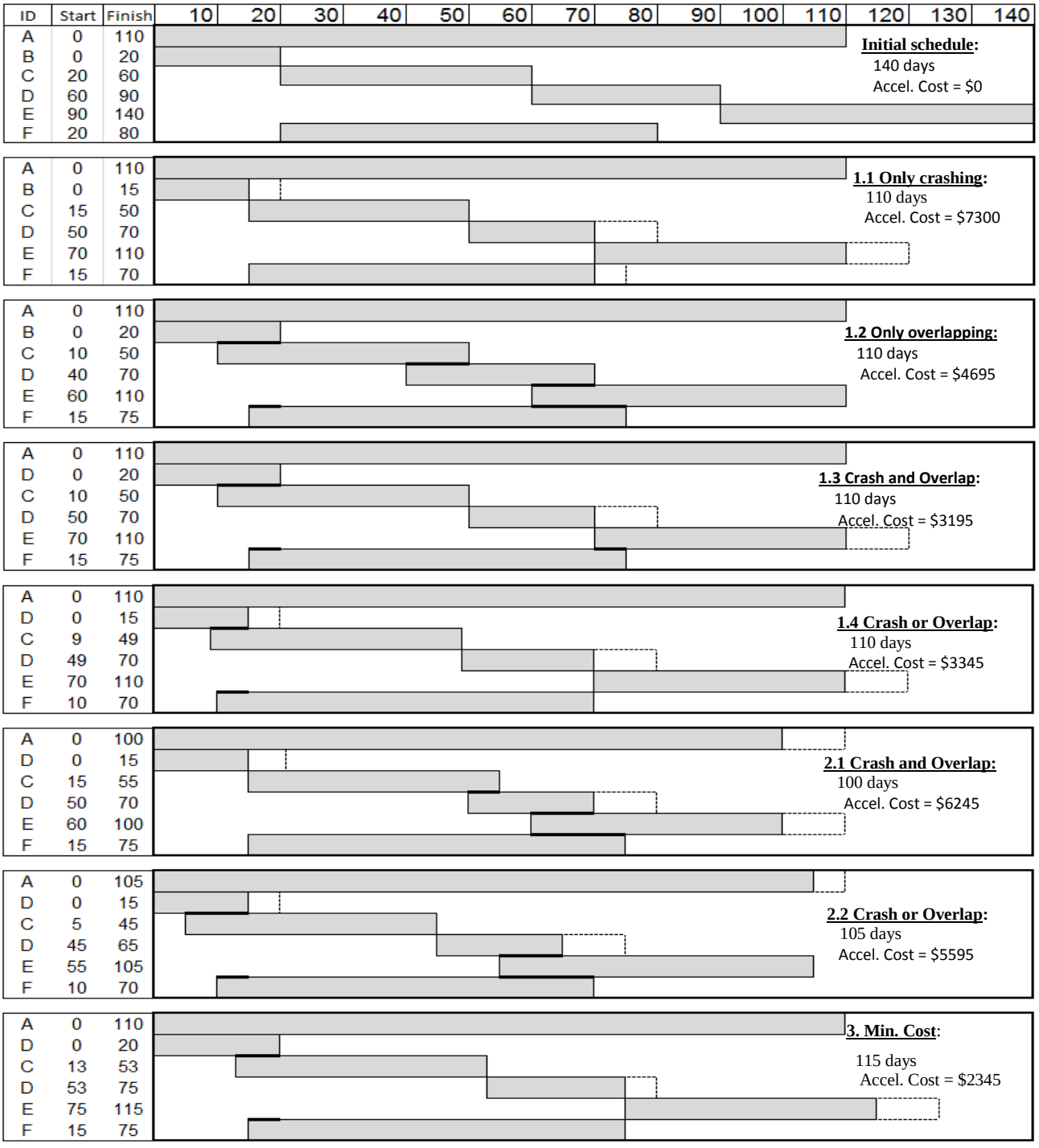


Figure 5.7 Case study-2 results of experiments

The results of the optimization experiments show that the CP model is able to efficiently optimize the schedule as desired by the user. The three experiments were re-executed with a resource limit of 20 workers/ day. For case study-2, activity crashing used an overtime strategy only. In this case, the optimization results show less usage of overlapping because it leads to more resource consumptions during the overlapped period and consequently an increasing in the Total Cost of each experiment, as shown in Figure 5.8.

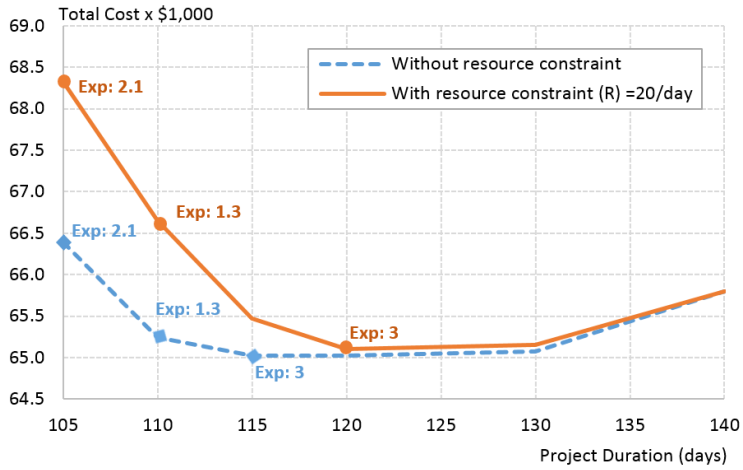
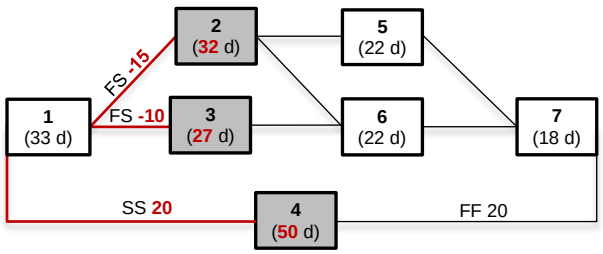


Figure 5.8 Effect of resource constraints on Case study-2 experiments results

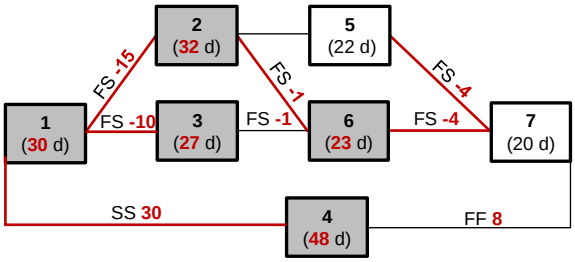
5.4 Case study-3: Phase I- Schedule Compression

To benchmark the proposed CP model, it was applied to a case study that involves both crashing and overlapping. The Case study-3 was used by Hazini et al (2013) to examine their heuristic schedule compression model. The same case study was also used by Hazini et al. (2014) to demonstrate a refined genetic algorithm model, however, both models did not consider resource constraints. To simplify the

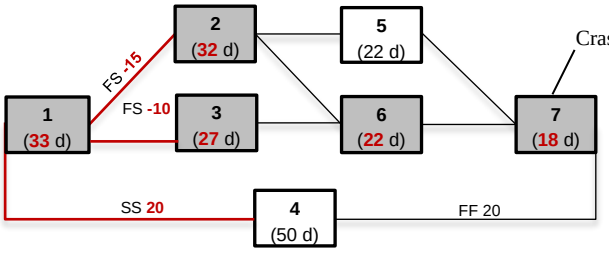
analysis of the heuristic model, Hazini et al. (2013) used an assumption that all the activities can be crashed or overlapped up to a maximum of 3 days only. For comparison purposes, a constraint was added to the CP model to provide the same condition (was not needed to compare with Hazini et al. 2014). Also, the CP model was modified so all durations are in hours, not days, to easily consider the daily fractions associated with the rework introduced by overlapping. It is noted that the formulation of this and that of Hazini (2014) use activity duration as one of the decision variables. Hazini et al (2014), however, does not use activity start time as another decision variable (as done in this paper). Rather, they use the relationship lead/lag days as a decision variable, which is a property of the relation, not the activity. However, such representation is suitable for schedule compression but is not general enough to handle resource constraints, which require some activities to be delayed (not overlapped) to avoid resource over-allocation. Figure 5.9 compares the CP results of the proposed model with the heuristic and genetic algorithm approaches of Hazini et al. (2013 and 2014). The figure shows that CP provides better solutions over the two other methods, both in terms of a shorter schedule and a higher cost benefit. Cases a, and b at the top of the figure represent the solutions of the heuristic model and the CP model, respectively. Cases c, and d at the middle of Figure 5.9 represent the solutions of the genetic algorithm and the CP model, respectively. It is important to mention that the CP model provided even better results (78.4 days, $10,600) if the solution is not restricted to the integer values of working days as shown in Case e at the bottom of Figure 5.9. The comparison results are sorted in Table 5.8.



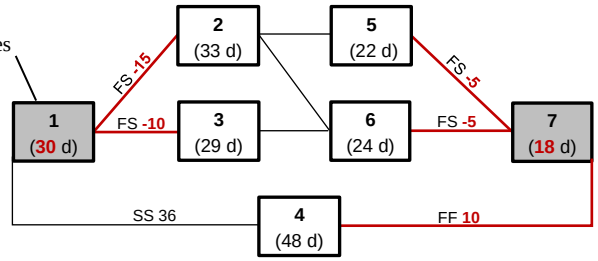
(a) Solution of Hazini et al. 2013(90 days, $8,154.8 benefits)



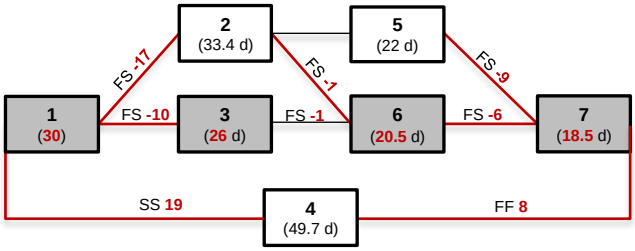
(b) CP solution (86 days, $9,328 benefits)



(c) Solution of Hazini et al. 2014 (90 days, $8,154.8 benefits)



(d) CP solution (86 days, $9,328 benefits)



(e) CP solution (78.4 days, $$10,599.9benefits)

Figure 5.9 Comparison of CP results of Case study-3 with Hazini (2013)

Table 5.9 Comparison of CP results of Case study-3 with Hazini (2013)

Study	Comments	Duration	Schedule Net Benefit $
a. Heuristic solution* (Hazini et al. 2013)	Real-number durations	98.0 d	$5,362.0
b. CP solution*	Real-number durations	96.0 d	$5,423.2
c. GA solution (Hazini et al. 2014)	Integer durations	90.0 d	$8,154.8
d. CP solution	Integer durations	86.0 d	$9,328.0
e. CP solution	Real-number durations	78.4 d	$10,599.9

* Extra constraint is added to limit activities crashing or overlapping up to a maximum of 3 days only

Comments on Case study-3 results:

Based on the model formulation and results, some comments on the developments made are as follows:

- The model formulation is able to combine crashing, overlapping, and substitution, in addition to resolving resource limits and improving resource profiles.
- The model is able to consider any type of the activity relationships (FS, SS, SF, and FF), schedules with multiple critical paths, multi-predecessor, and multi-successor.
- The ability to combine or to avoid having both crashing and overlapping on the same activity has a practical benefit and gives full flexibility to the project manager to adjust the schedule according to site restrictions, crew restrictions.
- The piecewise-linear crashing strategy is very detailed in the model and considers the detailed crew formation, and the type of changes to the crew daily work, including overtime,

overmanning, and multiple shifts. This feature is very practical as it suggests the specific method of applying crashing and the total needed overtime hours.

- One added benefit of a detailed crashing strategy (highlighted in the above point) is that it enables the project manager to allocate overtime to the specific days of the activity that has no overlapping. This avoids overstressing the activities, minimizes the chances of rework, and avoids communication/coordination problems.
- The CP tool used proved to be efficient. It produces optimal solutions within the range from (20 to 40 seconds).
- The model is flexible to adding new constraints that reflect practical situations. One possibility is to limit excessive rework, respect pre-defined milestones, and use dependent construction methods for some tasks.

5.4.1 Modified Case study-3

To demonstrate the generic capabilities of the presented models, same Case study-3 was used with various modifications to test the model (Figure 5.10). The figure shows the project network with both soft and hard relations. The data of the activities' modes as well as the possible crashing information are summarized in Table 5.10, where the optimization model decides the selected mode and crashing strategy simultaneously (data of activity 5 was used to draw the TCT spectrum of Figure 3.5) Table 5.11 also provides the rework time and cost associated with each additional overlapping day in the schedule. For example, if the schedule introduces a 10-day overlap between activity 1 and activity 2, then the associated rework time and cost are 0.2 x 10 = 2 days and 0.2 x 10 x $900 = $1800, respectively.

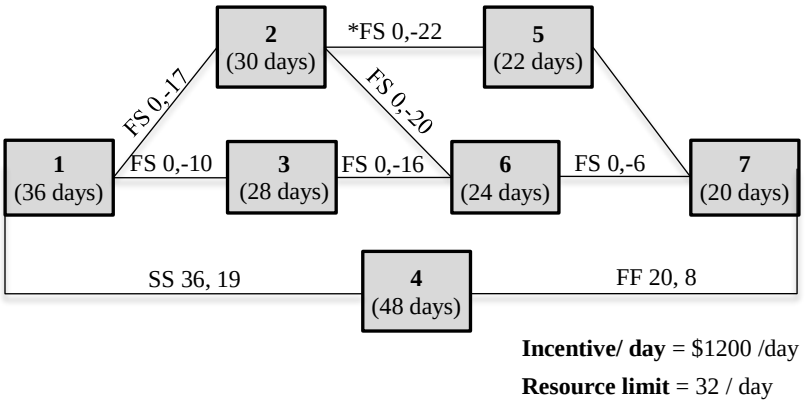


Figure 5.10 Network of the modified Case study-3

The initial project duration was prepared with all activities using their initial normal (slow and cheap) modes, and a resource limit of 32 resources per day, resulting in a project duration of 110 days and a total cost of 515,496 (top of Figure 5.11). After preparing case study-3 inputs to the optimization model, various optimization experiments were carried out. The objective of the first two experiments was to minimize total project cos considering multiple levels of resource constraints. The optimization results are shown in Figure 5.11 where the model was able to reach a minimum total cost of $491,750 in 92 days of project duration under a 32/day resource limit. The middle part of Figure 5.11 shows the resulting schedule with the specific strategy used for each task. Thee schedule has an optimum mix of crashing (activities 3 and 6), substitution (activity 4), and overlapping (activities 1, 2, and 3) in this experiment. The second experiment was then carried out with a stricter resource limit of only 22 resources per day. The resulting schedule is shown at the bottom of Figure 5.11 along with the strategy used for each task. As expected, the optimum schedule exhibits less use of overlapping (particularly among activities 4, 5, and 6) to avoid resource over-allocation. This, however, comes at the expense of

project duration which reached 110 days (same as the initial schedule but using 30% fewer resources), while total costs ($511,940) are still less than the cost of the initial schedule.

Table 5.10 Modes and crashing options of Case-4 (adapted from Hazini et al. 2014)

Activity	Mode	Crashing option	Cost $1,000	Duration (days)	Resource need (r)	Productivity Loss %	Cost Slope (CS) /day	OT Segment (S) /day
1	1	1. Normal	$ 76.4	36	10	-	-	-
		2. Overtime	$ 76.4	36 to 30	10	15%	$1,400	3.8 d
2	1	1. Normal	$ 72.0	30	10		-	-
		2. Overtime	$ 72.0	30 to 25	10	10%	$1,040	3.2 d
3	1	1. Normal	$ 70.2	28	10	-	-	-
		2. Overtime	$ 72.0	28 to 21	10	10%	$ 656	2.6 d
		3. OTM*	$ 75.4	20 to 18	12	10%	$ 276	18.0 d
4	1	1. Normal	$ 81.6	48	10	-	-	-
		2. Overtime	$ 81.6	48 to 40	10	20%	$1,760	4.4
	2	1. Normal	$ 82.4	40	10	-	-	-
5	1	1. Normal	$ 65.4	22	10	-	-	-
		2. Overtime	$ 65.4	22 to 17	10	17%	$ 961	3.0
		3. OTM	$ 70.9	16 to 15	12	17%	$ 527	15.0
	2	1. Normal	$ 97.7	16	15			
	3	1. Normal	$103.3	12	16			
6	1	1. Normal	$ 66.8	24	10	-	-	-
		2. Overtime	$ 66.8	24 to 19	10	15%	$1,184	3.5
	2	1. Normal	$ 69.0	20	12	-	-	-
	3	1. Normal	$72.0	16	13	-	-	-
7	1	1. Normal	$ 68.0	20	10	-	-	-
		2. Overtime	$ 68.0	20 to 16	10	20%	$ 1,120	2.6 d
		3. OTM	$ 73.4	15 to 12	12	20%	$ 658	14.0 d

*OTM= Overtime + Overmanning

Table 5.11 Rework data of Case study-4

Activity	Predecessor	Rework Time R_{ip}	Rework Cost C_{ip}
1	-	-	-
2	1	0.2	$900
3	1	0.1	$850
4	1	0.15	$1050
5	2	0.30	$800
6	2	0.25	$1000
	3	0.2.	$1000
7	4	0.15	$950
	5	0.05	$950
	6	0.25	$950

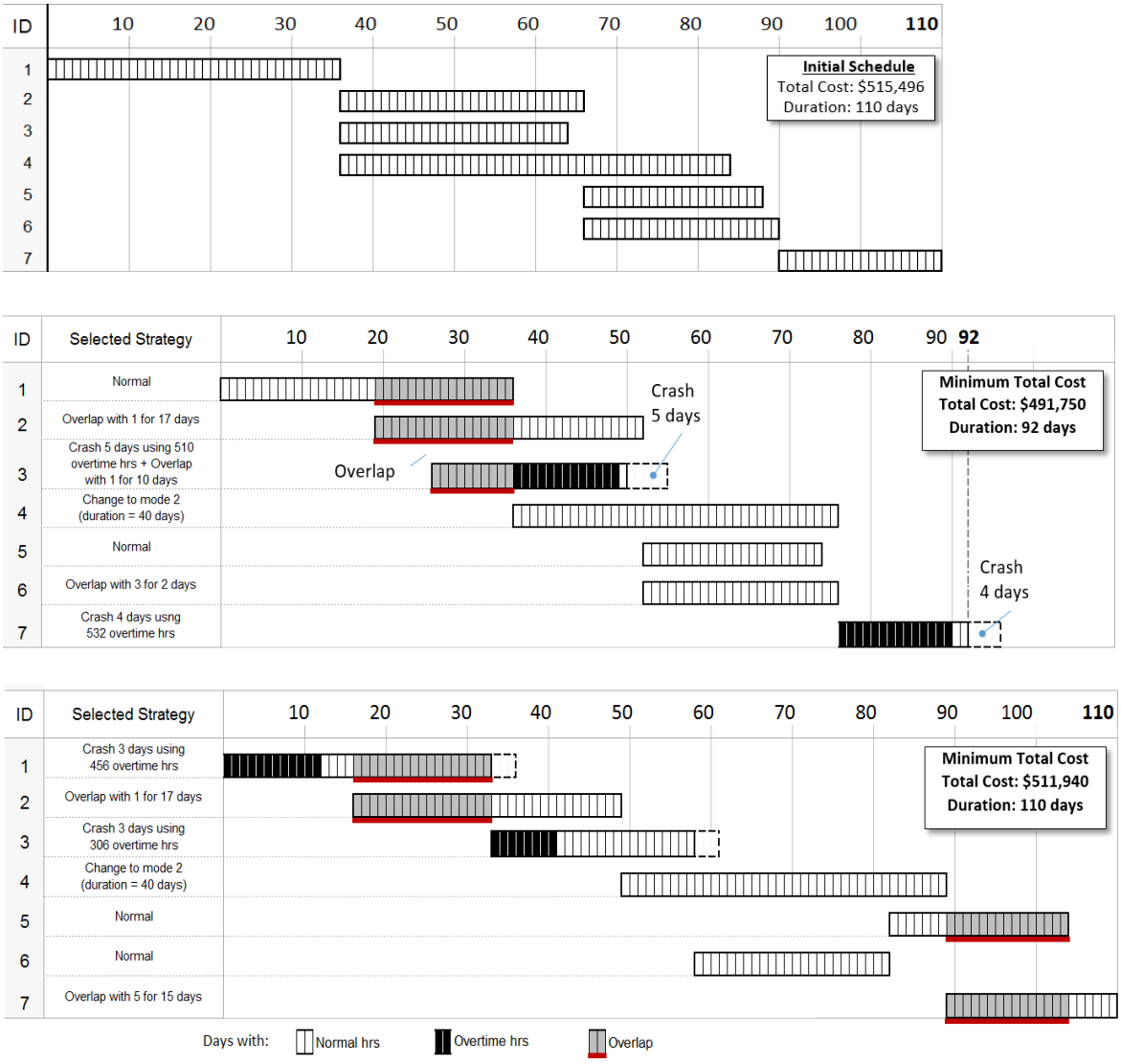
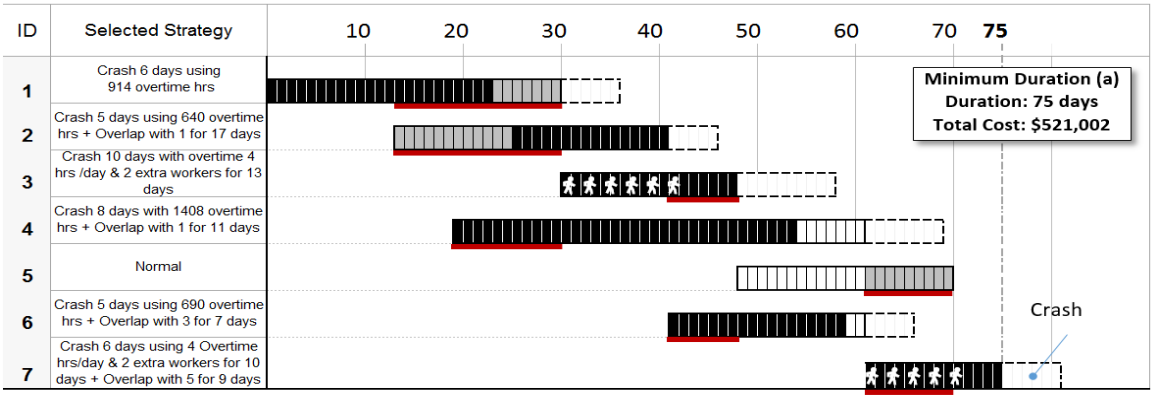


Figure 5.11 Optimum results of the cost minimization experiment modified Case study-3

Next optimization experiments targeted to minimize project duration (using a resource limit of 32 per day) and testing the advanced features of the model. Two experiments were performed as follows: (a) allowing simultaneous crashing and overlapping on same activity segment; and (b) preventing

simultaneous overlapping and crashing on the same activity segment (i.e., including the optional constraint in Eq. 4.4.). In case (a), the model was able to obtain a minimum duration of 75 days with a total cost of $521,002 (Figure 5.12a). To reach this aggressive duration, the schedule in Figure 5.12a show many activities using combinations of crashing and overlapping (e.g., activity 4 uses overtime, yet is overlapped with both 1 and 2). The results of case (b), on the other hand, are shown in Figure 5.12b and the model reached a minimum project duration of 82 days with a total cost of $518,842, which is a little longer than case (a) but is cheaper and exhibits no simultaneous use of two strategies, thus is better able to avoid overstressing the workforce.

(a) Combined crashing and overlapping allowed on same activity segment



(b) Crashing and overlapping in separate activity segments

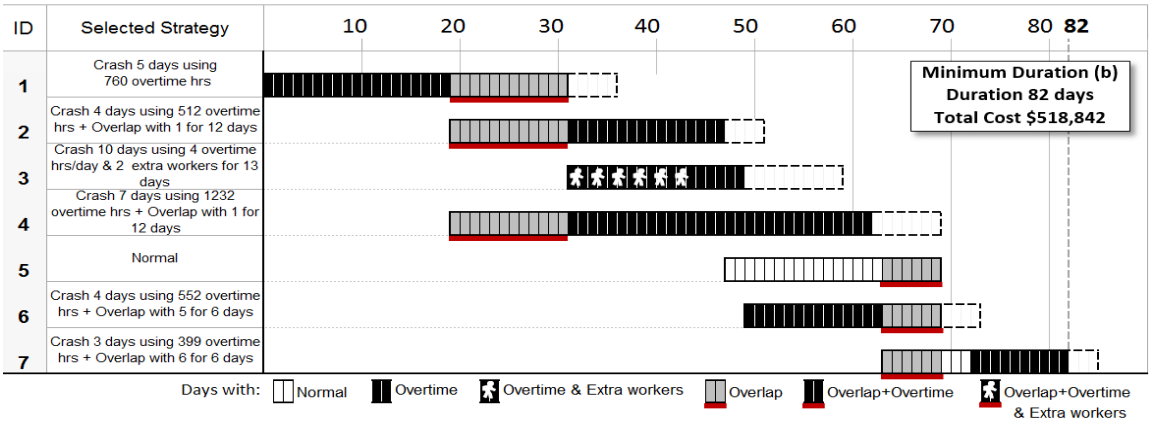


Figure 5.12 Optimum results of duration minimization experiments modified Case study-3

Based on the results shown in Figure 5.11 and Figure 5.12, the optimization framework performed consistently with the objective and constraints of each experiment, and showed to be able to use optimum combinations of a wide range of schedule compression strategies. Figure 5.13 also shows the project time-cost curve for case study-3, showing the optimum points under a resource limit of 22 and 32 workers/ day. It is observed that with a resource limit of 22 workers/day, the optimization results show less usage of overlapping because it leads to more resource use during the overlapped period and consequently an increase in project duration, as shown in Figure 5.13.

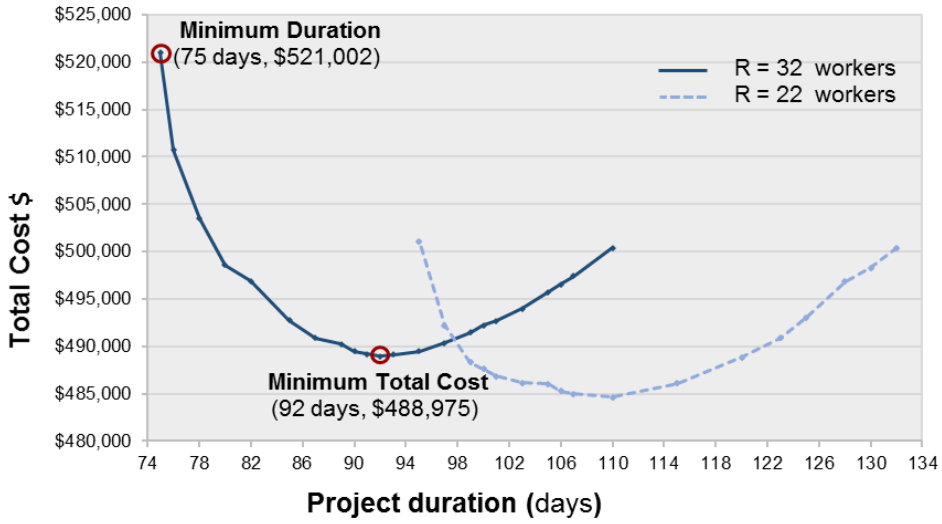


Figure 5.13 Time-cost curves with different resource limits of Case study-3

Comments on modified Case study-3 results:

- The model formulation determines the optimum recipe of project execution options that combine optional paths, crashing, mode substitution, and limited resource allocation;
- The model is able to consider all types of hard and soft (flexible) relationships (FS, SS, SF, and FF) and multiple-predecessors/successors in the overlapping calculations;
- The model visibly presents the full spectrum of activity time-cost-resource options. The piecewise-linear crashing spectrum clearly defines specific resource implementation strategy for any crashing option along with the applicable portion of activity duration; and
- The model can apply activity crashing and overlapping in different activity segments, thus avoids overstressing the workers and reduces the chances of rework.

5.5 Case study-4: Phase II - during Construction:

A small Case study-4 is presented, as shown in Figure 5.14, with activities optional modes, resource needs, relationships, as well as the project resource limits. Case study-4 is a modification of the one used by Liu and Shih (2009).

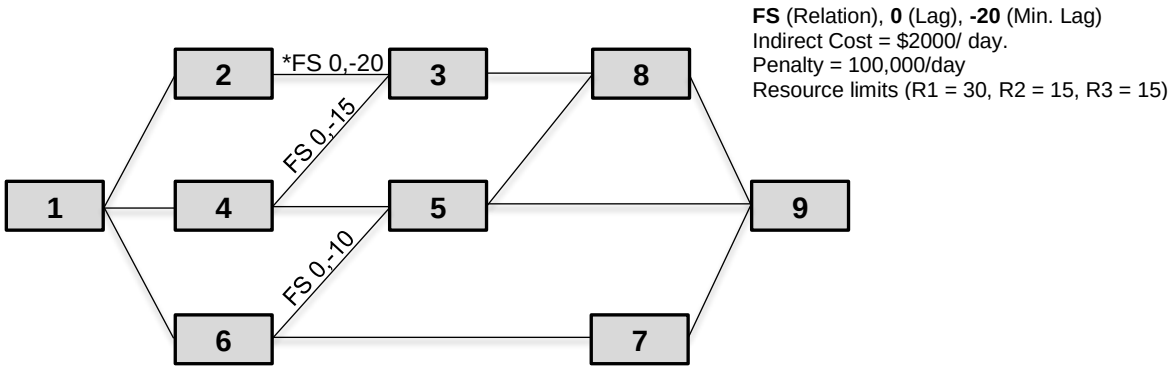


Figure 5.14 Case study-4 Network

Table 5.12 Data of the Case study-4

Activity	Dur.	R_1	R_2	R_3	Cost
A	***50***	***10***	***4***	***5***	***$280 K***
	60	6	4	5	$276 K
B	***90***	***8***	***5***	***2***	***$414 K***
C	120	8	6	6	$744 K
	130	***6***	***6***	***5***	***$702 K***
D	***150***	***10***	***2***	***4***	***$675 K***
E	120	2	5	6	$516 K
	130	2	5	4	$481 K
	140	2	5	2	$434 K
F	160	12	4	4	$928 K
	170	10	3	3	$782 K
	180	8	2	2	$612 K
	190	***6***	***1***	***1***	***$418 K***
G	130	6	3	6	$585 K
	140	***6***	***2***	***5***	***$532 K***
H	60	10	5	4	$404 K
	70	***12***	***4***	***3***	***$385 K***
	80	12	3	2	$384 K
I	75	10	5	6	$600 K
	85	10	5	5	$543 K
	90	***10***	***5***	***5***	***$540 K***

Table 5.13 Data of the initial mode and crashing options for Case study-4

No	Initial Mode (1)			Overtime Strategy			Overmanning Strategy				Outsourcing Strategy		
	Cost	NT	R_1	P.L. %	CT	$/ day	P.L. %	R_1	CT	$/ day	R_1`	CT	$/ day
A	$280 K	50											
B	$414 K	90	8	20%	72	1728	15%	10	83	2098	2	83	3785
C	$702 K	130	6	20%	104	1296	15%	8	113	936	2	113	1864
D	$513 K	114	10	20%	92	2224	15%	12	110	5814			
E	$434 K	140	2	30%	122	901							
F	$388 K	177	6	20%	142	1308	15%	8	153	903			
G	$532 K	140	6	20%	112	1296	15%	8	121	902			
H	$385 K	70	12	20%	56	2592	15%	15	65	3427	3	65	6106
I	$540 K	90	10	20%	72	2160	15%	12	87	6120			

*R_1 = Required Onsite workers, $R`_1$ = Outsourced workers

The baseline schedule uses the bold modes highlighted in Table 5.12, with baseline total cost being $5,640,000 and project duration being 540 days (top schedule in Figure 5.16). With a reporting period of 60 days, the project is currently in day 120 (2nd reporting period) with total cost-to-date being $1,100,000. Current progress shows that activity A is completed; while activities B, D, and E being in progress (90%, 35%, and 25%, respectively). As shown in Figure 5.15, the update schedule shows that the project expected to be 630 days with Total Cost $ 6,041,000. At that point, the project status is behind schedule with 90 days delay and cost overrun $ 581,000. Thus, a corrective action is being investigated for the project.

Once the Case study-4 inputs related to the linear crashing strategies are ready, various optimization experiments were carried out as shown in Table 5.13. The first three optimization experiments targeted comparing the results among the three proposed scenarios to optimize the corrective-action decisions during construction to prove the practicality to incorporate the project manager's preferences and minimize the changes of the baseline schedule.

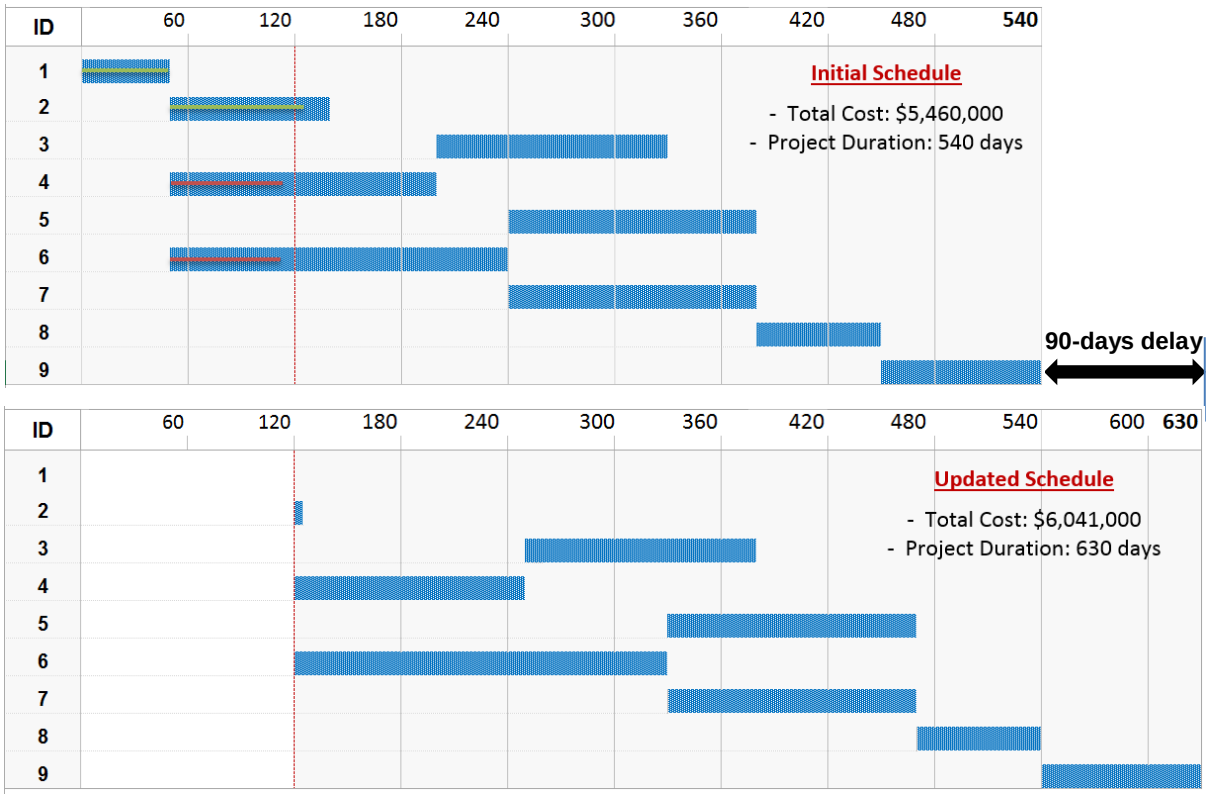


Figure 5.15 Baseline vs. updated Schedule for Case study-4

Scenario-1 "No Preferences": The experiment represents the traditional TCT problem where the objective function was minimizing total project cost, regardless the change that happens to the baseline schedule.

The optimization results are shown in Figure 5.16 with the specific strategy used for each task. The new schedule was able to recover the 90-days of delay at a minimum total cost of $571,335,9. The schedule has optimum mix of crashing (activities 5), substitution (activity 9), and overlapping (activities 5, and 6) in this experiment. The results shows that the corrective-actions are distributed along the schedule and the Total Schedule Variation equals 240.

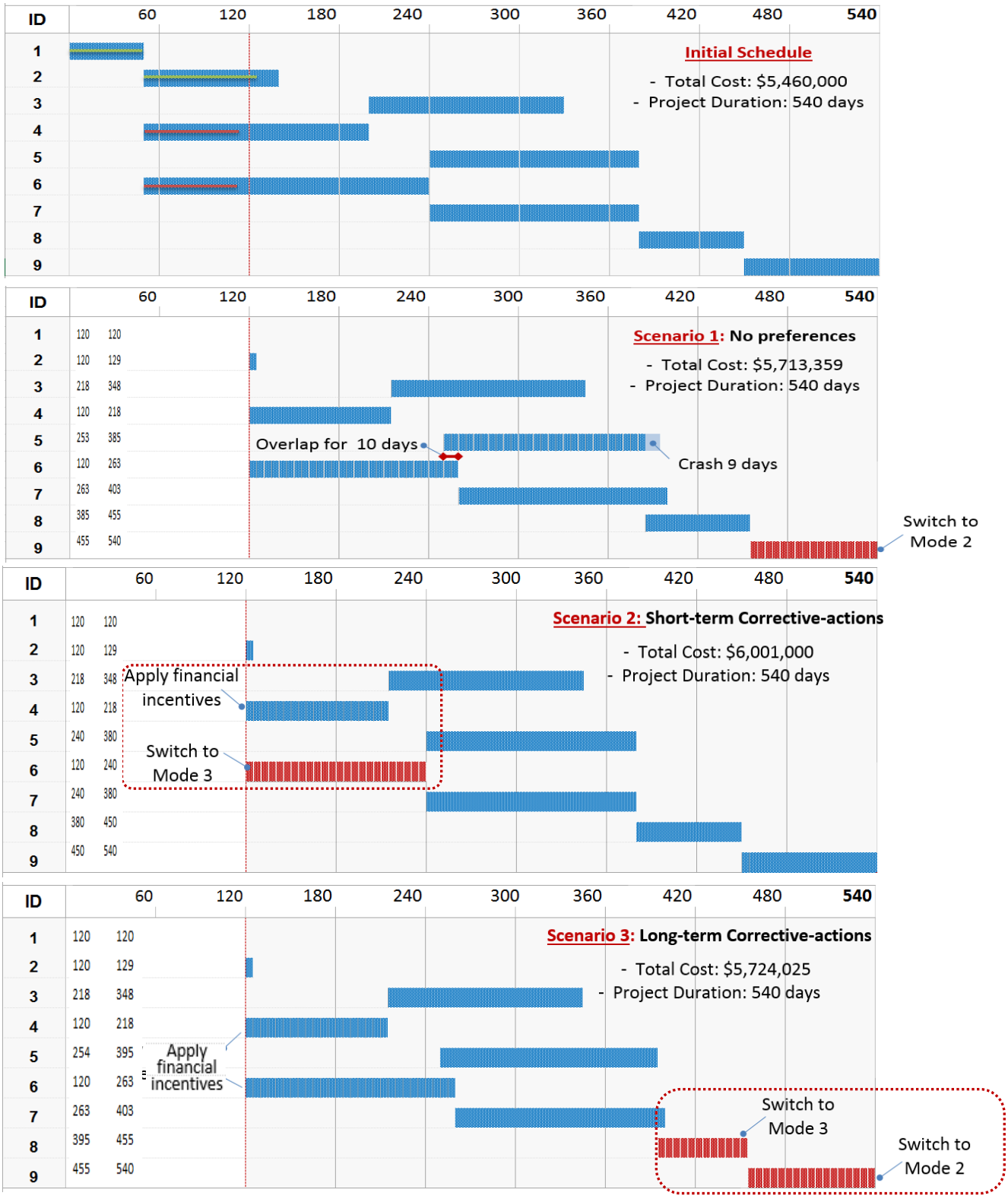


Figure 5.16 Baseline vs. alternative corrective-action schedules for Case study-4

Scenario-2 "Short-term corrective actions": the objective function was minimizing the total schedule variations weighted by linear descending preference function, and minimizing the total cost is a secondary objective function. The new schedule was able to recover the 90-days of delay at a minimum total cost of $6,001,000. The schedule has an optimum mix of substitution (activity 6), and the use of financial incentives (activities 4). The results show that the corrective-actions are present in first upcoming reporting period after the reporting date and the Total Schedule Variation equals 204.

Scenario-3 "Long-term corrective actions": the objective function was to minimize the Total Schedule Variations weighted by linear ascending preference function, in addition to the minimization of the total cost as a secondary objective function. The new schedule was able to recover the 90-days of delay at a minimum total cost of $572,402,5. The schedule has an optimum mix of substitution (activity 8, and activity 9), and applying financial incentives (activities 4, and 6). The results show that the corrective-actions are present in last reporting period and the Total Schedule Variation equals 215;

Table 5.14 Comparing Outputs of the proposed scenarios of Case study-4

Scenario	Total Cost ($)	Duration (days)	Corrective-action Cost $	Schedule Deviations
Scenario-1 (traditional TCT)	$5,713,359	540 d	$253,359	240
Scenario-2 Short-term corrective-action	$6,001,000	540 d	$541,000	204
Scenario-3 Long-term corrective-action	$5,724,025	540 d	$264,025	215

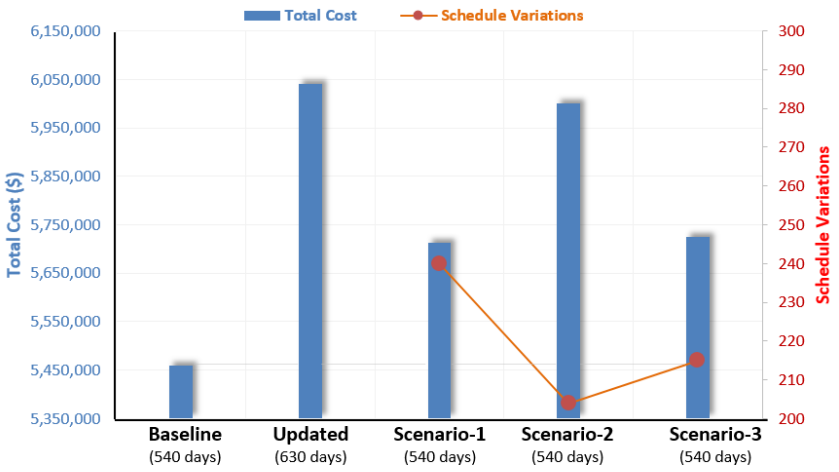


Figure 5.17 Comparison of results of different corrective-action plans for Case study-4

To test the model ability to consider the project manager's preferences regarding the flexibility of the project activities to change their modes during construction, one more experiment is performed. Assuming that the group of flexible activities includes activities 3, and 5; the group of possible activities includes activity 9; and inflexible activities are activities 6, 7, and 8.

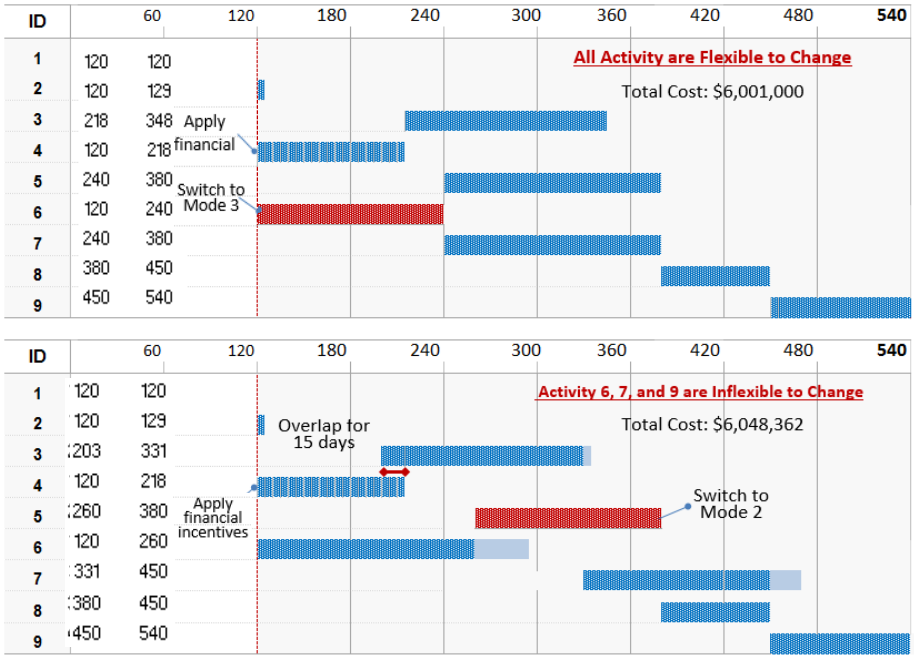


Figure 5.18 Case study-4 experiments of preferences to change activities' modes

A comparison of the last experiment results and the previous experiment of Scenario-2 (where all activities are allowed to change their modes during construction) is showing in Figure 5.18. The results of Scenario-2 previous experiment is shown in the top part of Figure 5.18 while the results of the new experiment is shown in the bottom par. Comparing the results of the two experiments show that the model complies with the preferences of the activity groups. Instead of switching the mode of activity 6 (inflexible activity) in the experiment of Scenario-2 as shown in the top part of Figure 5.18, activity 5 (flexible activity) switched its mode at the expense of increasing the cost from $6001000 to $6048362.

Comments on modified Case study-4 results:

- The formulation of the constraints and the objective functions of the extended model is able to consider the priority to accelerating short-term activities, or long-term activities according to the project manager's preferences. As shown in the results of Case study-4:
    - Applying "Scenario-1" resulted in schedule changes along the whole schedule. The resulted schedule is the minimum cost corrective-action plan, while the total schedule variation is the maximum of the two other experiments.
    - Applying "Scenario-2" resulted in schedule changes in the immediate period after the reporting date, while the mode of the long-term activities (i.e. 3, 5, 7, 8, and 9) is the baseline mode, which means they maintain their ability to be compressed. The presence of compressible long-term activities increases the retained capability of the schedule to compress toward absorbing future deviations and increase schedule future flexibility.

- Applying "Scenario-3" resulted in schedule changes in the last reporting period, where activities (8, 9) are substituted to faster modes while the ongoing activities (4, and 6) maintain their modes of construction. This corrective-action avoids disruption for the work crews and the committed procurements on the immediate periods after the reporting date which means more schedule stability for the immediate reporting periods.

- Considering different preferences of project manager during the optimization of corrective-actions yields to different project total cost, as expected, from that of considering only cost.

- The model formulation determines the optimum combination of the corrective actions that are required to recover from delays. The options of the corrective actions combine path substitution, linear crashing, mode substitution, and the resource limits, in addition to considering project manager's preferences.

- The CP tool proved to be efficient and produces optimal solutions within 20 to 40 seconds of processing-time.

- The model is able to consider all types of hard and soft (flexible) relationships (FS, SS, SF, and FF), multiple predecessors, and multiple successors.

- The fast speed and the unique features of the model, particularly incorporating the diverse preferences of project manager make the system practical and suitable especially for medium and large-size projects. In such projects, even small changes in the constraints, objectives, and preferences largely impact the scheduling decisions. These common decisions are difficult to meet using only the intuitive guessing and experience of the project managers or using the existing scheduling tools.

- Projects can largely benefit from the proposed decision support framework.

5.6 Experiment on large-scale Case-Study

To prove the suitability of the model to handle large-scale problems over 1000 activities, an optimization experiment was conducted with a total of 1,314 activities. The experiment used the same activities in case study-1 of 18 activities, and repeated the activities 73 times, in parallel, to generate one large project network of 1,314 activities. A significant advantage of developing larger models based on parallel copies of the base 18 activities case is that the optimum solution of this base network is already known and represents a baseline for comparison. The efficiency of the optimization model can thus be measured by comparing the deviation of the actual solution from the calculated optimum. For this experiment, the resource limit is multiplied by 73, but the deadline duration remained unchanged. Upon running this experiment, the CP optimizer showed superior results and achieved the exact optimum solution of the base case (165 days) without any deviation, within a processing time of about 20 minutes. Although these results are considered very efficient in terms of solution quality and processing time, full-scale experimentation with large-scale networks is beyond the scope of this research. This experiment just validated the suitability of CP models to complex and large-scale construction projects.

5.7 Summary

In this chapter, four case studies have been presented in order to demonstrate the advanced features and comprehensiveness of the model. The case studies have been chosen to illustrate the essential features of the model along the different stages of the project: work-packaging at the early planning, schedule

compression immediately before construction, and corrective-actions during construction. For comparison purposes, two of these cases are drawn from literature to prove the superior performance of the presented model.

For each case study, a number of experiments were considered taking into account different conditions. All the experiments' results are consistent with the objectives, constraints, and preference functions. The CP search algorithm is able to obtain optimal solutions for multi-objective combinatorial scheduling problems. The efficiency of the model flourishes when considering the processing time of the experiments. Considering the various experiments that are performed, the processing time was in the range between 10 seconds up to two minutes in few experiments. The CP shows superior results both in terms of the solution quality and the processing speed compared to the results of literature cases using the heuristic method and evolutionary-based algorithm. The model formulation and its CP engine are expected to provide efficient practical schedules and support the management of strictly constrained projects.

Chapter 6

Conclusions and Future Research

6.1 Summary

In large construction projects, studying the tradeoff between project duration and cost is critical for successful construction firms. Such tradeoff requires construction personnel to exercise the alternative options available to do the job, and to determine their optimum schedules. However, optimizing construction schedules is proven to be a combinatorial optimization problem, which is computationally difficult to be solved. A considerable research has been devoted to solve the schedule optimization problem over the last years using genetic algorithms, and meta-heuristic methods focusing on improving the solution quality and speed. Most of this research accommodates small size projects and uses limited options, and constraints such as activities crashing and resource levelling. However, real construction projects are large scale, and practitioners have more options to execute their projects, such as alternative crew formation, fast-tracking, and multimode activities; let alone incorporating all of these options collectively and adapting to the continuous changes during construction.

This work makes an effort to tighten the gap between the project scheduling literature, and the needs of project managers and schedulers to improve the project scheduling from a practitioner prospective. It develops an enhanced project schedule optimization framework that combines two phases of schedule optimization to suit their distinctive decisions and requirements: "Phase I: preconstruction", and "Phase II: during construction phase". The schedule optimization framework of Phase-I improves the decision making before construction by introducing a new representation of the scheduling options at the network level, activity level, relation level, and project level. The framework also integrates practical

constraints within a multi-objective decision making environment. Then, Phase-II framework extends the Phase-I framework to improve the decision making of corrective actions and recovery plans during construction; considering the baseline schedule decisions, and evolving constraints as a comprehensive dynamic scheduling framework. The comprehensive framework combines mechanisms to analyze the collected progress data and the workers' status during construction, prepares list of the potential corrective actions for each activity, and incorporates the project manager's preferences about the corrective-action decisions' implementation.

The extensions and enhancements were formulated in a generic mathematical formulation to optimize schedule decisions at any stage. This formulation integrates a wide range of scheduling options (e.g. linear crashing, activity multimode, fast-tracking, and multipath network), and a set of practical constraints (e.g. variable resources availability, correlation, and intermediate milestones) to optimize the tradeoff among the project time, cost, resource, and baseline schedule adherence. The mathematical formulation of the comprehensive framework is then coded using the advance constraint programming tool "IBM ILOG CPLEX Optimization Studio". To validate the model, multiple experiments on four different case studies were used to prove the functionality, practicality, and its better representation of real-life construction challenges. Two of these case studies are from the literature to prove the ability of the comprehensive model to achieve better solutions. Construction experts were also consulted at multiple stages of this work, and they confirmed the practicality and relevance of the model.

6.2 Conclusions

The development of the proposed framework and the multiple optimization experiments conducted in this research revealed important conclusions, as follows:

- Optimizing the schedules using a combination of various options (i.e., alternative crew formations, alternative activity modes, alternative degrees of overlapping, and alternative network paths) can meet strict constraints of deadline and limited resources with a considerable cost saving, either before or during construction.
- Practical constraints of correlation between activities' modes is important to represent the reality of construction. However, it is not necessary that considering correlation will have a negative impact on project time or cost.
- Satisfying persistent milestones without violating resource limits or project deadline can achieve cost-benefits to the project.
- Small changes in the optimization objectives priority or relative importance considerably impact the activities' modes and sequence.
- Using the two-step multi-objective function results in better solutions, compared to the single weighted objective function and multi-criteria objective function. However, using the multi-criteria objective-function simplifies the idea of improving the schedule after satisfying the main (first) objective function in a single experiment. For example, multi-criteria function of minimizing the total project cost and minimizing resource fluctuation achieves the minimum cost and improves the resource profile simultaneously. Another example, multi-criteria

function of minimizing total project cost and minimizing the start of scheduled activities achieves the minimum cost and reduces the criticality of the schedule simultaneously.

- The optimum project network is a function of the objective function and constraints, at the planning stage and during construction. The project network that achieves minimum cost is not the same network that achieves the minimum duration.
- Considering several project manager's preferences during corrective-action optimization yields to different project total costs.
- Selection among extending the hours for already available resources, and utilizing any internally idle resources or outsourcing additional resources is important for highly constrained resources availabilities. It contributes to considerable cost savings and increase the efficiency of resource utilization.
- The ability to combine or avoid having both crashing and overlapping on the same activity has a practical benefit and gives full flexibility to the project manager to adjust the schedule according to the site and crew restrictions.
- The detailed crashing strategy specifies the detailed crew formation, and the type of changes to the crew's daily work; including overtime, overmanning, and multiple shifts. Detailed crashing strategy also enables the project manager to allocate overtime to the specific days of the activity that has no overlapping. This avoids overstressing the activities, minimizes the chances of rework, and avoids communication/coordination problems.
- Experimentations of the model on schedule compression scheduling using linear crashing, overlapping, and substitution shows that the use of overlapping is highly correlated to the

resource limits. Hence it is not applicable to study the integration of overlapping and crashing without considering the resource constraints as occurred in many schedule compression models.

- "Activity order in the project timeline" is an added important criteria when prioritizing the activities for acceleration despite the fact that the "Activity cost slope" is the only criteria used in most of the published work.

6.3 Contributions

This work enhances the schedule optimization research by efficient modeling of real-life decisions and constraints, and develops a framework to optimize the planning and corrective-action decisions dynamically before and during construction. It provides a decision support mechanism to achieve real cost savings, and better management of projects within the constraints. The scientific contributions of this work, for both theory literature and practice, are as follows:

Understanding of real-life decisions at different construction phases: This research develops a detailed understanding of the real-life parameters, decisions, and constraints that are essential to optimize construction project schedules during planning and execution phases. In addition to the extensive literature review, and author's practical experience, meetings with experts in the construction field (from local and international construction companies) were very helpful to demonstrate the project managers best practices and preferences;

Comprehensive dynamic schedule optimization framework: This research has resulted in the development of a new scheduling optimization framework (as a detailed Decision Support System

DSS) that combines two distinctive phases to improve the decision making before and during construction. This framework primarily introduced a new representation of the option at the network, activity, relations, as well as practical constraints. The framework also considers the project manager's preferences about the changes that could happen to the baseline schedule due to the corrective-action decisions;

Satisfaction-driven corrective-action optimization: The concept of satisfaction functions is successfully utilized to customize the corrective-action plan according to the manager's preferences to minimize the baseline changes of on the short-term, or long-term. Experiments show that considering project manager's preferences during the optimization of corrective-actions yield different project total cost, as expected, from that of considering only cost. This feature enables the integration of contractors' judgment and experience in the acceleration unlike the traditional TCT.

Better representation of project network options: The framework supports optional network decisions using mathematical representation of the project network with multiple groups of alternative branches (paths). The planner will greatly benefit from selecting the optimum project recipe that satisfies the preconstruction phase; as well as, making path-substitution decisions to expedite late project delivery during construction. Experimentations on the model proved the ability of the model to determine the final optimum project network as a function of the objective function and constraints at the planning and during construction.

Rich representation of activity execution options: This work introduces "*Activity TCR Spectrum*" as a visual representation of linear change of crew formation, and/or resource calendar within each activity discrete. The spectrum combines the tradeoff of time-cost-resource to show the wide spectrum of options available to execute the activity. Experimentations show that this model enables the planner to optimize their combined decision of activity mode selection, and crew formation options (e, g. overtime, and overmanning) in a resource constrained environment. The planner is able to optimize among options of extending the hours for already available resources, versus utilizing any internally idle resources or outsourcing additional resources.

Efficient integration of schedule compression strategies: The model makes an efficient comprise between applying multiple schedule compression strategies and avoiding site congestion and overstressed workers, leading to more practical schedules. The model uses optional constraints to apply activity crashing, and overlapping in separate activity segments. Thus, it avoids overstressing the workers and reduces the chances of rework. These optional constraints are based on the ability of the spectrum to clearly define the applicable portion of activity duration "*Activity crashing segment*" for any crashing option. Experiments show that the model is performing consistently with the imposed optional constraints;

Fast-tracking using soft relations: The framework accommodates fast-tracking strategy within the traditional CPM calculations. A new flexible (soft) relation replaced the rigid representation of the logical relationship between two activities. The formalization of flexible relation is a generic logical relationship (hard or soft) of any type (finish-to-start, etc.) between any two activities, where it indicates the permissible overlapping range between the activities. Experimentations show that

the model enables the planner to exercise the alternative intermediate fast-tracking scenarios versus the other scheduling optimization options while accounting for resource constraints.

Fine-tuning of Construction Schedules: The framework supports practical features that enables the project manager to highly fine-tune and customize their schedules. These features are as follows:

- Defining a correlation between any pair of modes of two or more activities. Experiments show that considering correlation constraint did not necessitate additional cost or time.
- Satisfying persistent milestones. Important feature particularly if these milestones give the owner an opportunity to start using parts of the project.
- Avoiding excessive reworks due to either a high degree of overlapping between two activities, or overlapping of one activity with multiple predecessors. This constraint limits schedule complications, and risks that could be emerge due to excessive overlapping.
- Considering of nonrenewable, doubly constrained resources, in addition to the variability of renewable resource availability. This practical feature enables the project manager to correctly define the available resource limits.
- Single and multi-objective decision environment of time, cost, and resources' variations. The experiments show the impact of using multi-objective optimization function. Even small changes in the objectives priority, or their relative importance considerably impact the activities' modes and sequence. Especially with large scale projects, this impact is more intense.

Handling of complex networks: The framework is able to define all activity relation types (i.e. FS, SS, FF, and SF), and to consider activities with multiple predecessors and successors. The schedules of construction projects are very complex. It is common to see several prerequisites for a task and also have several successors.

Accurate schedule updating: As opposed to the current practices of schedule updating that the remaining work of ongoing activities will follow the planned progress rates, even if the work so far has proceeded at a much slower rate. This framework provides more accurate computation of the remaining durations by considering both the activity progress status, and the level of workers' morale in the calculation of the remaining duration and the elected list of corrective actions of each individual activity. This can lead to better assessment of project deviations, and accordingly, more effective corrective-action plans.

Constraint Programming (CP) framework: Developing the detailed Constraint Programming (CP) model is a very involved task, that requires integrated mathematical modeling of the problem and subsequent coding in the CP environment. Such an effort represents a contribution to both the academia and to the professionals for the following reasons:

- CP model results are more efficient compared with literature case studies.
- CP model produces fast and optimal solutions. The time required for the experiments was in the range between 10 to 20 seconds in all experiments.

- The fully automated CP environment for model implementation is a step forward to the adoption of the proposed schedule optimization features within the commercial scheduling systems.

The encouraging results of this work will hopefully change the practitioners' perception that optimization is a theoretical and complex tool that is difficult to use. In addition, some of the model's features can be introduced into standard project management software,. Accordingly, optimization can become a main stream tool for construction scheduling.

6.4 Future Research

Despite the presented framework features and benefits, it introduces some limitations and potential improvements that are currently being addressed as follows:

- The proposed model assumes that the durations, costs, and resources of project activities are deterministic. Future work can address the uncertainties associated with weather, site conditions, and other factors through probabilistic scheduling optimization.
- Currently, the model does not allow activity splitting. This can be an additional point of improvement in the model to determine the optimum splitting times and durations, consider the expected improvements in resource allocation, and leveling as opposed to the time and cost implications, and the effect of interruptions on productivity.
- Enhancing the model to accommodate the resource-constrained with time-dependent parameters (requesting and availability) to overcome CPM problems associated with the use of multiple resource calendars.

- Extending the framework to analyze the critical path, and total floats using the results of multiple experiments with different constraints and objectives.
- Performing a detailed rework analysis in cases of overlapping design-design, design-construction, and construction-construction activities. This analysis will consider the characteristics of the activities to accurately determine the probability of rework, as well as its time and cost. The analysis also will study adding multiple intermediate reworks to the successor activity depending on frequent exchange of the evolving information during the overlapping, which can also produce a rework to the predecessor activity.
- Revising the representation of flexible relations by defining not only a minimum lag value, but also a maximum lag as well. This will provide a more practical range of overlapping options during optimization.
- Generalizing the model to cover repetitive scheduling is an area for future research: a single project can be considered as a special case of multiple and repetitive projects.
- Adopting financing decisions and project control could be incorporated into the model such as cash flow analysis, earned value, cost and schedule performance indices, and productivity analysis.
- Extending the model formulation to consider construction quality in the optimization problem.
- Future work will include experiments on large-scale projects to test processing time and solution quality, where all the examples in the paper are small in size.
- The application of the developed model to real-life projects is essential future work in order to validate the practicality.

- Incorporate the model into an Integrated Decision Support System (DSS) that serves construction management, supports the decision making process, and enables flexibility and adaptability to accommodate changes in the construction environment.

References

1. Abuwarda, Z., and Hegazy, T., (2016a) “Flexible Activity Relations to Support Optimum Schedule Acceleration,” Technical Note, Journal of Construction Engineering and Management, ASCE, Volume 142, Issue 11, DOI: 10.1061/(ASCE)CO.1943-7862.0001193.

2. Abuwarda, Z., and Hegazy, T., (2016b) “Work-package planning and schedule optimization for projects with evolving constraints,” Journal of Computing in Civil Engineering, ASCE, Volume 30, issue 1, DOI: 10.1061/(ASCE)CP .1943-5487.0000587.

3. Abuwarda, Z., and Hegazy, T., (2018) “Optimum Schedule Compression with and without Simultaneous Activity Crashing, Substitution, and Overlapping” Automation in Construction, accepted and under publishing

4. Afshar, A., Kasaeian Ziaraty, A., Kaveh, A., and Sharifi, F. (2009). “Non-dominated Archiving Multi-colony Ant Algorithm in Time-Cost Trade-Off Optimization.” Journal of Construction Engineering and Management, 135(7), 668-674.

5. Ahsan, S., El-Hamalawi, A., Bouchlaghem, D., and Ahmad, S. (2009) “Applications of converged networks in construction,” International Journal of Product Development 7, 281-300.

6. Ahuja, V., and Thiruvengadam, V. (2004). "Project scheduling and monitoring: current research status." Construction Innovation, 4(1), 19-31.

7. Akpan, E. O. P. (2000). “Resource smoothing: a cost minimization approach.” Journal of Production Planning and Control, 11(8), 775-780.0

8. Allouche, M. A. (2014). Manager’s Preferences Modeling within Multi-Criteria Flow shop Scheduling Problem : A Metaheuristic Approach, International Journal of Business Research and Management, Vol. (1), 33–45.

9. Ashuri, B., and Tavakolan, M. (2012). Fuzzy Enabled Hybrid Genetic Algorithm – Particle Swarm Optimization Approach to Solve TCRO Problems in Construction Project Planning. Journal of Construction Engineering and Management, ASCE, 1065–1075.

10. Ashuri, B., and Tavakolan, M. (2013). A Shuffled Frog-Leaping Model for Solving Time-Cost-Resource Optimization (TCRO) Problems in Construction Project Planning. Journal of Computing in Civil Engineering, 29(1).

11. Ballestín, F., Valls, V., and Quintanilla, S. (2009). Scheduling projects with limited number of preemptions. Computers and Operations Research, 36, 2913–2925.

12. Barták, R., and Cepek, O. (2007). Nested Temporal Networks with Alternatives, Association for Advancement of Artificial Intelligence. 1–8.

13. Baweja, S. S. (2006). "CPM schedules - Why and how." AACE International Transactions, PS.22.1- PS.22.5.

14. Beck, J. C., Feng, T. K., and Watson, J. (2011). "Combining Constraint Programming and Local Search for Job-Shop Scheduling." INFORMS Journal on Computing, 23(1), 1–14,

15. Berthaut, F., Grèze, L., Pellerin, R., Perrier, N., and Hajji, A. (2011). Optimal Resource-Constraint Project Scheduling with Overlapping Modes. Cirrelt, (09), 1–15.

16. Blazewicz, J., Lenstra, J. K., and Rinnooy Kan., A. H. G., (1983) "Scheduling subject to resource constraints: Classification and complexity. Discrete Applied Mathematics, 11-24.

17. Bogus, S. M., Diekmann, J. E., Molenaar, K. R., Harper, C., Patil, S., and Lee, J. S. (2013). Simulation of Overlapping Design Activities in Concurrent Engineering. Journal of Construction Engineering and Management, 137(11): 950-957.

18. Borcherding, J., and Garner, F., (1981) "Work force motivation and productivity on large jobs," Journal of the Construction Division, 107, 443-453.

19. Bradley, S. P., Hax, A. C., and Magnanti, T. L. (1977). Applied Mathematical Programming. Addison- Wesley. Retrieved from http://web.mit.edu/15.053/www/.

20. Brailsford, S. C., Potts, C. N., and Smith, B. M. (1999). "Constraint satisfaction problems: Algorithms and applications." Europe Journal of Operational Research, 119(3), 557–581.

21. Brucker, P., Drexl, A., Mo ¨hring, R., Neumann, K., Pesch, E., (1999). Resource-constrained project scheduling: notation, classification, models, and methods. European Journal of Operational Research 11, 3–41.

22. Castro-Lacouture, D., Süer, G.A., Gonzalez-Joaqui, J., and Yates, J.K. (2009). "Construction Project Scheduling with Time, Cost, and Material Restrictions Using Fuzzy Mathematical Models and Critical Path Method." Journal of Construction Engineering and Management, 135(10), 1096-1104.

23. Chan, W. T., and Hu, H. (2002). "Constraint Programming Approach to Precast Production Scheduling." Journal of Construction Engineering and Management, ASCE, 128(6), 513-521.

24. Chan, W. T., and Zeng, Z. (2003). "Coordinated production scheduling of prefabricated building components." Proc., Construction Research Congress, ASCE, Reston, VA, 1–8.

25. Chassiakos, A. P., and Sakellaropoulos, S. P. (2005). "Time-cost optimization of construction projects with generalized activity constraints." Journal of Construction Engineering and Management, ASCE, 131(10), 1115-1124.

26. Chen, S.-P., and Tsai, M.-J. (2011). Time-cost trade-off analysis of project networks in fuzzy environments. European Journal of Operational Research, 212(2), 386–397.

27. Chen, S.-P., and Tsai, M.-J. (2011). Time-cost trade-off analysis of project networks in fuzzy environments. European Journal of Operational Research, 212(2), 386–397.

28. Cherian, M., and Gopalakrishnan Nair, T. R., (2011) "Multipath Routing With Novel Packet Scheduling Approach In Wireless Sensor Networks, International Journal of Computer Theory and Engineering, Vol. 3, No. 5.

29. Cho, S., and Eppinger, S. D. (2005). A Simulation-Based Process Model for Managing Complex Design Projects. IEEE Transactions on Engineering Management, 52(3), 316–328.

30. Chong, Z. (2012). Dynamic Optimization Formulations for Plant Operation under Partial Shutdown Conditions. PhD's thesis, University of McMaster University

31. Chua, D.K.H., Shen, L.J. and Bok, S.H. (2003) Constraint- based planning with integrated production scheduler over internet. Journal of Construction Engineering and Management, 129 (3), 293–301.

32. CII (2015) "Concepts and Methods of Schedule Compression" https://www.construction-institute.org , June 2015.

33. Dehghan, R., and Ruwnapura, J., 2014, Model of Trade-Off between Overlapping and Rework of Design Activities. Journal of Construction Engineering and Management., 140(2):

34. Dehghan, R., Hazini, K., Ruwanpura, J., (2015) "Optimization of overlapping activities in the design phase of construction projects" Automation in Construction 59, 81–95.

35. Dehghan, R., Ruwnapura, J. Y. (2014). Model of trade-off between overlapping and rework of design activities. Journal of Construction Engineering and Management, 140(1), 1–13. https://doi.org/10.1061/(ASCE)CO.1943-7862.0000786.

36. Demeulemeester, E.L. and W.S. Herroelen. (2002). "Project Scheduling: A Research Handbook". Boston: Kluwer Academic Publishers.

37. Dishan, Q., Chuan, H., Jin, L., and Manhao, M. (2013). A dynamic scheduling method of earth-observing satellites by employing rolling horizon strategy. The Scientific World Journal. http://doi.org/10.1155/2013/304047

38. Elbeltagi, E., (2009). Lecture Notes on Construction Project Management, Structural Engineering Department. Mansoura University, Egypt. 2009.

39. Elbeltagi, E., Hegazy, T., and Grierson, D. (2005). "Comparison among five evolutionary-based optimization algorithms." Journal of Advanced Engineering Informatics, 19(1), 43–53.

40. Elmeghraby, S.E., Salem, A. (1981). Optimal linear approximation in project compression. Operations Research Technical Report 171, North Carolina State University at Raleigh.

41. El-Rayes, K., and Jun, D. H. (2009). Optimizing Resource Leveling in Construction Projects. Journal of Construction Engineering and Management, ASCE, 135(11), 1172–1180.

42. El-Rayes, K., and Jun, D. H. (2009). Optimizing Resource Leveling in Construction Projects. Journal of Construction Engineering and Management, 135(11), 1172–1180. http://doi.org/10.1061/(ASCE)CO.1943-7862.0000097

43. El-Rayes, K., and Kandil, A. (2005). Time-Cost-Quality Trade-Off Analysis for Highway Construction. Journal of Construction Engineering and Management, ASCE, 131(4), 477–486.

44. Eppinger, S. D. (1997). "A planning method for integration of large-scale engineering systems." International Confonference on Engineering Design, Tampere, Finland, 199–204.

45. Feng, C. W., Liu, L., and Burns, S. a. (1997). "Using Genetic Algorithms to Solve Construction Time-Cost Trade-Off Problems", Journal of Computing in Civil Engineering, 11(3), 184–189.

46. Feng, Chung-Wei, Liu, L., and Burns, S. A. (2000). Stochastic Construction Time-Cost Trade-Off Analysis. Journal of Computing in Civil Engineering, 14(2), 117. doi:10.1061/(ASCE)0887- 3801(2000)14:2(117)

47. Focacci, F.; Laborie, P.; and Nuijten, W. 2000. “Solving Scheduling Problems with Setup Times and Alternative Resources.” Proceedings of AIPS 2000, American Association of Artificial Intelligence.

48. Fondahl, J. W. (1961). A non-computer approach to the critical path method for construction *industry*, https://mosaicprojects.com.au/PDF/John_W_Fondahl.pdf.

49. Fortrock Construction (2013) “Helpful Construction Tips for Construction Worker Morale”, (http://www.fortrockconstruction.com/general-contractor-blog-eugene-oregon/), accessed in Feb, 2014.

50. Fulkerson, D. R. (1961). A Network Flow Computation for Project Cost Curves. Management Science, 7(2), 167–178. https://doi.org/10.1287/mnsc.7.2.167.

51. Galloway, P. D. (2006). "Survey of the construction industry relative to the use of CPM scheduling for construction projects." Journal of Construction Engineering and Management, ASCE, 132(7), 697–711.

52. Garey M. R. and Johnson D. S. (1979) Computers and intractability: A guide to the theory of NP-completeness. Freeman, San Francisco, California.

53. Gasparini, F. F. B, and Qassim, R. Y. c. (2003). A mixed integer linear programming model for set-up reduction superstructures. International Journal of Production Research, 41(5), 1087–1092.

54. Gerk, J. E. V., and Qassim, R. Y. (2008). “Project acceleration via activity crashing, overlapping, and substitution”, IEEE Transactions on Engineering Management, 55(4), 590–601.

55. Gerk, J. E. V., and Qassim, R. Y. (2008). “Project acceleration via activity crashing, overlapping, and substitution”, IEEE Transactions on Engineering Management, 55(4), 590–601.

56. Ghoniem, A. S. (2002). Static and dynamic job-shop scheduling using rolling- horizon approaches and the Shifting Bottleneck Procedure, Master's thesis, Ecole des Mines de Nantes – France.

57. Goldenhar, L., Williams, L., Swanson, J., (2003) "Modelling relationships between job stressors and injury and near-miss outcomes for construction labourers," An International Journal of Work, Health and Organisations, Vol. 17, NO. 3, 218-240.

58. Golzarpoor, B. (2012). Time-Cost Optimization of Large-Scale Construction Projects Using Constraint Programming, Master thesis, University of Waterloo.

59. Gorman, M., and Kanet, J. (2010). "Formulation and solution approaches to the rail maintenance production gang scheduling problem", Journal of Transportation Engineering, 136(8), 701–708.

60. Grèze, L., Pellerin, R., Leclaire, P., and Perrier, N. (2014). CIGI2011: A heuristic method for resource-constrained project scheduling with activity overlapping. Journal of Intelligent Manufacturing, 25(4), 797–811. https://doi.org/10.1007/s10845-012-0719-5.

61. Gutjahr, W. J., Strauss, C., and Wagner, E. (2000). A Stochastic Branch-and-Bound Approach to Activity Crashing in Project Management. INFORMS Journal on Computing, 12(2), 125–135. doi:10.1287/ijoc.12.2.125.11894

62. Hajdu, M. 1996. "Network Scheduling Techniques for Construction Project Management." Kluwer Academic Publishers, ISBN 0-7923-4309-3

63. Hanna, A. S. (2003). "The effectiveness of innovative crew scheduling techniques." Research Report No. 185-11, Construction Industry Institute, Austin, Tex.

64. Hanna, A. S., Taylor, C. S., and Sullivan, K. T. (2005). "Impact of Extended Overtime on Construction Labor Productivity", Journal of Construction Engineering and Management, 131(6), 734–739.

65. Hariga, M., and El-Sayegh, S. M. (2011). Cost Optimization Model for the Multiresource Leveling Problem with Allowed Activity Splitting. Journal of Construction Engineering and Management, ASCE, 137(1), 56–64.

66. Hazini, K., Dehghan, R., and Ruwanpura, J. (2013). "A heuristic method to determine optimum degree of activity accelerating and overlapping in schedule compression", Canadian Journal of Civil Engineering, 40, 382–391.

67. Hazini, K., Dehghan, R., and Ruwanpura, J. (2014). "An evolutionary optimization method to determine optimum degree of activity accelerating and overlapping in schedule compression." Canadian Journal of Civil Engineering, 41, 333–342.

68. Hegazy, T. (1999). "Optimization of resource allocation and leveling using genetic algorithms." Journal of Construction Engineering and Management, 125(3), 167–175.

69. Hegazy, T. (2002). Computer-Based Construction Project Management. Prentice Hall, Upper Saddle River, NJ, USA.

70. Hegazy, T., and Ersahin, T. (2001). Simplified Spreadsheet Solutions: Schedule Optimization, (December), 469–475.

71. Hegazy, T., and Wassef, N. (2001). Cost optimization in projects with repetitive nonserial activities. Journal of Construction Engineering and Management, ASCE, 127(3), 183–191.

72. Hegazy, T., Elbeltagi, E., and Zhang, K. (2005). Keeping Better Site Records Using Intelligent Bar Charts. Journal of Construction Engineering and Management, ASCE, 131(5), 513-521.

73. Hegazy, T., Petzold, K., (2003). "Genetic optimization for dynamic project control", Journal of Construction Engineering and Management, ASCE 129 (4), pp.396–404.

74. Heipcke, S. (1999). "Comparing constraint programming and mathematical programming approaches to discrete optimization - The change problem." Journal of the Operations Research, 50, 581–595.

75. Hendrickson, C. and Au,T. (1989) Project Management for Construction , Prentice - Hall, Inc., Englewood Cliffs, N.J.

76. Hentenryck, P. V., 2002, "Constraint and Integer Programming," INFORMS Journal on Computing, Vol. 14, No. 4, pp. 345–372.

77. Herrera, C., Berraf, S. B., and Parada, V. (2015). A reactive decision-making approach to reduce instability in a Master Production Schedule

78. Herroelen, W., Reyck, B, Demeulemeester, E., (1998) "Resource-constrained project scheduling: a survey of recent developments," Computers and Operations Research, vol. 25(4), 279–302.

79. Hiroyasu, T.; Miki, M.; Watanabe S. (2000). The new model of parallel genetic algorithm in multi-objective optimization problems - divided range multi-objective genetic algorithm. IEEE Proceedings of the 2000 Congress on Evolutionary Computation, vol. 1, pp. 333-340. DOI: 10.1109/CEC.2000.870314.

80. Hossain, M. a., and Chua, D. K. H. (2014). "Overlapping design and construction activities and an optimization approach to minimize rework." International Journal of Project Management, 32(6), 983–994.

https://doi.org/10.1080/00207543.2015.1078516

81. Hurst, c., (2011). "How others Measure and Manage Employee Morale" (http://beyondmorale.com/), (Jan. 15, 2014).

82. IBM ILOG CPLEX Optimization Studio V12.7. (2017). CP Optimizer User's Manual. International Business Machines Corporation, Armonk, New York.

83. Jaselskis, E. Ruwanpura, J. Becker T., Silva, L.P., Jewell, E. Floyd, Innovation in construction engineering education using two applications of Internet-based information technology to provide real-time project observations, Journal of Construction Engineering and Management. 137 (2010) 829-835.

84. Jergeas, G. F., and Ruwanpura, J. (2010). Why Cost and Schedule Overruns on Mega Oil Sands Projects? Practice Periodical on Structural Design and Construction, 15(1), 40–43. http://doi.org/10.1061/(ASCE)SC.1943-5576.0000024

85. Jergeas, G. F., and Ruwanpura, J. (2010). Why Cost and Schedule Overruns on Mega Oil Sands Projects? Practice Periodical on Structural Design and Construction, 15(1), 40–43. http://doi.org/10.1061/(ASCE)SC.1943-5576.0000024

86. K.R. Apt, Principles of Constraint Programming, Cambridge University Press, Cambridge, 2003, ISBN 0-521-82583-0.

87. Kandil, A., and El-Rayes, K. (2005). "Parallel computing framework for optimizing construction planning in large-scale projects." Journal of Computing in Civil Engineering, 19(3), 304–312.

88. Kaplan, L. 1988. Resource-constrained project scheduling with preemption of jobs. Unpublished PhD Dissertation. University of Michigan, Michigan, USA.

89. Kastner, R., (2006) "Why Projects Succeed: Take Corrective Action"; https://pmhut.com/why-projects-succeed-take-corrective-action; last access; Jan 2018.

90. Ke, H., Ma, W., and Ni, Y. (2009). Optimization models and a GA-based algorithm for stochastic time- cost trade-off problem. Applied Mathematics and Computation, 215(1), 308–313. doi:10.1016/j.amc.2009.05.004.

91. Kelleher, A. (2004). "An investigation of the expanding role of the critical path method by ENR's top 400 contractors." Master's thesis, Faculty of Virginia Polytechnic Institute and State University, Blacksburg, Va.

92. Kelleher, G. Cavichiollo, P. (2001) Supporting rescheduling using CSP, RMS, and POB — an example application, Journal of Intelligent Manufacturing, vol. 12 (4), Springer, pp. 343–357.

93. Kelley, J. E. (1961). Critical-Path Planning and Scheduling: Mathematical Basis. Operations Research, 9(3), 296–320.

94. Kelly, J. E. (1961). "Critical Path planning and scheduling: mathematical basis." Operations Research, 9(3), 296-320.

95. Khoueiry, Y., Srour, I., and Yassine, a. (2013). “An optimization-based model for maximizing the benefits of fast-track construction activities”, Journal of the Operations and Research, 64(8).

96. Kim, J., and Ellis, J. (2008). “Permutation-Based Elitist Genetic Algorithm for Optimization of Large-Sized Resource-Constrained Project Scheduling.” Journal of Construction Engineering and Management, 134(11), 904-913.

97. Kim, K., and de la Garza, J. M. (2003). "Phantom float." Journal of Construction Engineering and Management, ASCE, 129(5), 507-517.

98. Konak, A., Coitb, D. W., & Smith, A. E. (2006). “Multi-objective optimization using genetic algorithms: A tutorial.” Reliability Engineering and System Safety, 91, 992-1007.

99. Koyuncu, E., and Erol, R. (2015). “PSO based approach for scheduling NPD projects including overlapping process”, Computers and Industrial Engineering, 85, 316–327.

100. Krishnan, V., Eppinger, S., and Whitney, D. 1997. A model-based framework to overlap product development activities. Management Science, 43(4): 437–451.

101. Kuhn, A. J. (2006). "CPM not working well on small projects." AACE International Transactions, PS.21.1-PS.21.3.

102. Laborie, P., and Rogerie, J. (2008a). Reasoning with Conditional Time-Intervals. Proceedings of the Twenty-First International Florida Artificial Intelligence Research Society Conference, May 15-17, 2008, Coconut Grove, Florida, USA, 555–560.

103. Laborie, P., and Rogerie, J. (2008b). Reasoning with Conditional Time-Intervals. Proceedings of the Twenty-First International Florida Artificial Intelligence Research Society Conference, May 15-17, 2008, Coconut Grove, Florida, USA, 555–560.

104. Laborie, P., Rogerie, J., Shaw, P., and Vilim, P. (2009). Reasoning with Conditional Time-Intervals. Part II: An Algebraical Model for Resources. FLAIRS Conference, 201–206.

105. Lam, H. C., and Lu, M. (2006). Critical Path Scheduling Under Resource-Availability and Activity-Interruption Constraints. Proceeding Of 2006 Annual CSCE Conference Of The Canadian Society For Civil Engineering, Page No: Ct-035 031-039, Calgary, Canada, May, 2006.

106. Lee, D. E., Yi, C. Y., Lim, T. K., & Arditi, D. (2010). "Integrated simulation system for construction operation and project scheduling." Journal of Computing in Civil Engineering, 24(6), 557–569.

107. Lee, J., and Kim, Y. (1996). "Search Heuristics for Resource Constrained Project Scheduling." The Journal of the Operational Research Society, 47(5), 678-689.

108. Leu, S., and Yang, C. 1999. GA-Based Multicriteria Optimal Model for Construction Scheduling. Journal of Construction Engineering and Management, ASCE, 125(6): 420-427.

109. Li, H., and Love, P. (1997). Using Improved Genetic Algorithms to Facilitate Time-Cost Optimization. Jr. of Construction Engineering and Management, 123(3), 233–237.

110. Li, H., and Zhang, H. (2013). Ant colony optimization-based multi-mode scheduling under renewable and nonrenewable resource constraints. Automation in Construction, 35, 431–438.

111. Liang, L., Burns, S. A., and Chung-Wei, F. (1995). Construction time-cost trade-off analysis using LP/IP hybrid method. Journal of Construction Engineering and Management, 121(4), 446– 454.

112. Liao, T.W., Egbelu, P. J., Sarker, B. R., and Leu, S. S. (2011). “Metaheur- istics for project and construction management—a state-of-the-art review.” Automation in Construction, 20(5), 491–505.

113. Liberatore, M. J., Pollack-Johnson, B., and Smith, C. A. (2001). "Project management in construction: Software use and research directions." Journal of Construction Engineering and Management, ASCE, 127(2), 101–107.

114. Liess, O., and Michelon, P. (2008). “A constraint programming approach for the resource-constrained project scheduling problem.” Annals of Operations Research, 157(1), 25–36.

115. Lim, T., Yi, C., and Lee, D., (2014). Concurrent Construction Scheduling Simulation Algorithm. Computer-Aided Civil and Infrastructure Engineering 29: 449–463.

116. Liu, S. S., and Wang, C. J. (2011). Optimizing project selection and scheduling problems with time-dependent resource constraints. Automation in Construction, 20(8), 1110–1119. http://doi.org/10.1016/j.autcon.2011.04.012

117. Liu, S. S., Wang, C. J., Liu, S., and Wang, C. (2008). Two - stage profit optimization model for linear scheduling problems considering cash flow, 6193(July). http://doi.org/10.1080/01446190903233111

118. Liu, S., and Shih, K. (2009). “Construction Rescheduling based on a manufacturing Rescheduling framework.” Automation in Construction, 18(6), 715–723.

119. Livengood, J. and Anderson, M. (2006). "A continuously changing as-planned baseline?" AACE International Transactions, CDR 08.1-CDR08.8

120. Lowsley, S., and Linnett, C. (2006). About Time: Delay Analysis in Construction. RICS Business Services Limited, UK.

121. Lu, M., and Lam, H.-C. (2008). "Critical path scheduling under resource calendar constraints." Journal of Construction Engineering and Management, ASCE, 134(1), 25-31.

122. Lu, M., and Li, H. (2003). "Resource-activity critical-path method for construction planning." Journal of Construction Engineering and Management, ASCE, 129(4), 412-420.

123. Markos, S., (2010) "Employee Engagement: The Key to Improving Performance" international Journal of Business and Management, Vol. 5, No. 12

124. Martel, J. M., and Aouni, B. (1990). Incorporating the Decision-maker's Preferences in the Goal-programming Model. Journal of the Operational Research Society, Vol. 41(12), 1 121-1 132. http://doi.org/10.1139/er-2016-0032

125. McCormick, C. R. (2002). Liquidated Damages : The Do's and Don'ts from an Owners Perspective. 30th Annual CSCE Conference 2002 proceedings.

126. Menesi, W. (2010). Construction Scheduling Using Critical Path Analysis with Separate Time Segments. Structure, PhD's thesis, University of Waterloo.

127. Menesi, W., and Hegazy, T. (2014). "Multimode Resource-Constrained Scheduling and Leveling for Practical-Size Projects." Journal of Construction Engineering and Management, ASCE, ISSN 0742-597X/04014092(7).

128. Menesi, W., Golzarpoor, B., and Hegazy, T. (2013). "Fast and near optimum schedule optimization for large-scale projects." Journal of Construction Engineering and Management ASCE, 139(9), 1117–1124.

129. Menesi, W., Golzarpoor, B., and Hegazy, T. (2013). "Fast and near optimum schedule optimization for large-scale projects." Journal of Construction Engineering and Management ASCE, 139(9), 1117–1124.

130. Meyer, W. L., and Shaffer, L. R. (1963). "Extensions Of the critical path method through the application of integer programming." Civil Engineering and Construction Research Series No.2, University Of Illinois, Urbana-Champaign.

131. Moselhi, O. (1993). Schedule compression using the direct stiffness method. Canadian Journal of Civil Engineering, 20(1), 65–72.

132. Nasira, H., Ahmeda, H., Haas, C., and Goodrumb, M. (2013) "An analysis of construction productivity differences between Canada and the United States," Journal of Construction Management and Economics.

133. Ng, S. T., and Zhang, Y. (2008). Optimizing Construction Time and Cost Using Ant Colony Optimization Approach. Journal of Construction Engineering and Management, ASCE, 134(9), 721–728.

134. Pagnoni, A. (1990). "Project Engineering: Computer Oriented Planning and Operational Decision Making", Springer, Berlin.

135. Pena-Mora, F., and Li, M. (2001). "Dynamic planning and control methodology for design/build fast track construction projects." Journal of Construction Engineering and Management.

136. Peteghem, V. V., and Vanhoucke, M. (2010). "A genetic algorithm for the preemptive and non-preemptive multi-mode resource-constrained project scheduling problem." European Journal of Operational Research, 201(2), 409–418.

137. Phillips, S., and Dessouky, M. I. (1977). Solving the Project Time/Cost Tradeoff Problem Using the Minimal Cut Concept. Management Science, 24(4), 393–400.

138. Pritsker A. B.; and Happ. W. W., (1966) GERT: Graphical evaluation and review technique - Part I: Fundamentals. Journal of Industrial Engineering, 17: 267-274.

139. Pritsker, B., Watters, L. J., and Wolfe, P. M., (1969) Multiproject scheduling with limited resources: A zero-one programming approach. Management Science, 16:93-107.

140. Project Management Solutions. (2011). Strategies for Project Recovery, A PM solutions Research Report; http://www.pmsolutions.com/resources/view/strategies-for-project-recovery/

141. Rao, (2010) "Human Resource Management," (Third Edition), Publisher Excel Books India

142. Roemer, T., and Ahmadi, R. (2004). Concurrent Crashing and Overlapping in Product Development. Operations Research, 52(4), 606–622.

143. Russell, D., Robinson, A. (1994). Automated Corrective Action Selection Assistant, Journal of Construction Engineering and Management, 120(1), 11–33.

144. Sabuncuoglu, I. and Kizilisik, O.B., (2003) "Reactive Scheduling in a Dynamic and Stochastic FMS Environment," International Journal of Production Research, 41, 17, 4211-4231.

145. Scavino, N. J. (2003). "Effect of multiple calendars on total float and critical path." Cost Engineering Journal, AACE International, 45(6), 11-15.

146. Senouci, A. B., and Eldin, N. N. (2004). "Use of Genetic Algorithms in Resource Scheduling of Construction Projects." Journal of Construction Engineering and Management, 130(6), 869-877.

147. Shahriari, M., (2016) "Multi-objective Optimization of Discrete time–cost Tradeoff Problem in Project Networks Using Non-dominated Sorting Genetic Algorithm" Journal of Industrial Engineering International 12(2), 159–169 springer.

148. Siemens, N. (1971). "A Simple CPM Time-Cost Tradeoff Algorithm," Management Science, 17(6), B: 354-363.

149. Siemiatycki, M. (2009). Academics and Auditors: Comparing Perspectives on Transportation Project Cost Overruns. Journal of Planning Education and Research. 29: 142-156.

150. Son, J., and Mattila, K. (2004). "Binary resource leveling model: Activity splitting allowed." Journal of Construction Engineering and Management, 130(6), 887–894.

151. Speranza, M. G., and Vercellis, C., (1993) Hierarchical models for multi-project planning and scheduling. European Journal of Operational Research, 64:312-325.

152. Sridharan, V. and Laforge, R.L., "The effect of safety stock and freezing on schedule instability", in: Proceedings of 1989 Decision Sciences Institute Meeting, Decision Sciences Institute, Atlanta, GA, November 1989.

153. Srour, I. M., Abdul-Malak, M. A. U., Yassine, A. a., and Ramadan, M. (2013). A methodology for scheduling overlapped design activities based on dependency information. Automation in Construction, 29, 1–11.

154. Srour, I. M., Abdul-Malak, M. A. U., Yassine, A. a., and Ramadan, M. (2013). A methodology for scheduling overlapped design activities based on dependency information. Automation in Construction, 29, 1–11.

155. Standish Group International 2015 https://www.infoq.com/articles/standish-chaos-2015

156. Street, I. A. (2000). The pitfalls of CPM scheduling on construction projects. Cost Engineering, 42(8), 35-37.

157. Suhail, S. and Neale R. (1994). "CPM/LOB: New Methodology to Integrate CPM and Line of Balance." Journal of construction Engineering and Management, ASCE, 120(3), 667-684.

158. Su-Ling, F., Kuo-Shun, S., Yu-Ren, W., (2012). "GA optimization model for repetitive projects with soft logic," Automation in Construction (21), 253–261.

159. T.Y. ElMekkawy, H.A. ElMarraghy, Real-time scheduling with deadlock avoidance in flexible manufacturing systems, International Journal of Advanced Manufacturing Technology, vol. 22(3–4), Springer, 2003, pp. 259–270.

160. Talbot, F. B. (1982). Resource-Constrained Project Scheduling with Time-Resource Tradeoffs: The Non-preemptive Case. Management Science, 28(10), 1197–1210.

161. Tamimi, S., Diekmann, J. (1988). “Soft logic in network analysis.” Journal of Computing in Civil Engineering, ASCE 2(3), 289–330.

162. Thomas, H. R., Horman, M. J., de Souza, U., and Završki, I. (2001). “Reducing variability to improve performance as a lean construction principle.” Journal of Construction Engineering and Management, 1282, 144–154.

163. Tsamardinos, I., Vidal, T. and Pollack, M.E. 2003. “CTP: A New Constraint-Based Formalism for Conditional Temporal Planning.” Constraints, 8(4):365-388.

164. United Construction (2014) “Keep up Morale with the Construction Crew”, (http://constructionunited.biz/); Feb. 25, 2014).

165. Vanhoucke, M. (2012). Project Management with Dynamic Scheduling and Risk Analysis. https://doi.org/10.1007/978-3-642-25175-7

166. Vanhoucke, M. (2012). Project Management with Dynamic Scheduling: Baseline Scheduling, Risk Analysis and Project Control. The Measurable News, (2), 315. http://doi.org/10.1007/978-3-642-25175-7

167. Vanhoucke, M., Debels, D., and Sched, J. (2008). The discrete time/cost trade-off problem: Extensions and heuristic procedures. Journal of Scheduling, 10(4–5), 311–326.

168. Veronik, A., Riantini, L., and Trigunarsyah, B., (2006). Corrective Action Recommendation for Project Cost Variance in Construction Material Management, The Tenth East Asia-Pacific Conference on Structural Engineering and Construction August 3-5, 2006, Bangkok, Thailand

169. Vieira, G. E., Herrmann, J. W., and Lin, E. (2003). Rescheduling Manufacturing Systems : a Framework of Strategies, Policies, and Methods, 39–62.

170. Vrat, P., and Kriengkrairut, C. (1986). A goal programming model for project crashing with piecewise linear time-cost trade-off. Engineering Costs and Production Economics, 10(1), 161–172. http://doi.org/10.1016/0167-188X(86)90037-6

171. Wang, L., Lin, Y., and Lin, P., (2007) “Dynamic mobile RFID-based supply chain control and management system in construction,” Advanced Engineering Informatics; 21(4), 377-390.

172. Wei-xin, W., Xu, W., Xian-long, G., and Lei, D. (2014). Advances in Engineering Software Multi-objective optimization model for multi-project scheduling on critical chain. Advances in Engineering Software, 68, 33–39. http://doi.org/10.1016/j.advengsoft.2013.11.004

173. Wickwire, J. M., and Ockman, S. (2000). "Industry crisis: Construction scheduling software." 2000 AACE International Transactions, CDR.02.1-CDR.02.8.

174. Xiong, Y., and Kuang, Y. 2008. Applying an ant colony optimization algorithm- based multiobjective approach for time–cost trade-off. Journal of Construction Engineering and Management, 134(2): 153–156. Doi: 10.1061/(ASCE) 0733-9364(2008)134:2(153).

175. Yano, C. and Carlson, R., 1985. An analysis of scheduling policies in multiechelon production systems. IEEE Transactions, 17, 370–377.

176. Yu, S. Sun, J. Hau, Flexible Dynamic Scheduling Based on Immune Algorithm, PROLAMAT, Shanghai, 2006, pp. 887–895.

177. Zahraie, B., and Tavakolan, M. (2009). Stochastic Time-Cost-Resource Utilization Optimization Using Nondominated Sorting Genetic Algorithm and Discrete Fuzzy Sets. Journal of Construction Engineering and Management, ASCE, 135(11), 1162–1171.

178. Zhang, Y., and Fan, Z.-P. (2014). An optimization method for selecting project risk response strategies. International Journal of Project Management, 32(3), 412–422.

179. Zhao, Leeb, T. S. (1993). Freezing the master production schedule for material requirements planning systems under demand uncertainty, Journal of Operations Management, 11 (1993) 185-205.

180. Zheng, D. X. M., Ng, S. T., and Kumaraswamy, M. M. (2004). Applying a Genetic Algorithm Based Multiobjective Approach for Time-Cost Optimization. Journal of Construction Engineering and Management, ASCE, 130(2), 168–176.

181. Zia, S., (2011) “Effects of organizational team building on employees' morale and job retention,” Business Management Dynamics Vol.1, No.7, pp.31-37.

Appendix A

OPL Model of Project Scheduling Problem for IBM ILOG

Optimization Studio

```
/*********************************************
 * OPL 12.7 Model
 * Author: Zinab Abuwarda Mohamed
 * Creation Date: Dec 25, 2014 at 1:06:29 AM
 *********************************************/
using CP;
int Indirect= …;
int Incentive = …;
int OriginalProjectDuration = …;
int Deadline = …;
int GivenBudget=…;
int CapRsrcL = ...;
int CapRsrcM = ...;
int CapRsrcE = ...;
int unit=…;
          int   Todate=…;
          float todateCost=…;
          int strategy= …;//
          int NRepPeriods = …;
tuple Task {
            key int id;
            int    compulsory;
            int   FS1; int  lag1;  int  MPL1; float Rework1;float Cost1;
            int   FS2; int  lag2;  int  MPL2; float Rework2;float Cost2;
            int   FS3; int  lag3;  int  MPL3; float Rework3;float Cost3;
            int   SS4; int  lag4;  int  MPL4; float Rework4;float Cost4;
            int   FF5; int  lag5;  int  MPL5; float Rework5;float Cost5;
            int   NT;int   NC;
            int   SF1;int   SF2;int   SF3;int   SS;int   FF;
           int    mile; float   milecost; float   milebonus;
           int   NTBaseline; float NCBaseline;int   startBaseline;
           int ongoing;
 }
{Task} Tasks = ...;

tuple Mode {
          key int taskId;
          key int id;
          int   NT;  float  NC;   int CT;  float CS;
       int dmdL; int dmdM; int dmdE;float s;
```

```
}
{Mode} Modes = ...;

tuple Correlation {
  int taskId1; int taskId2;
  int id1; int id2;
}
{Correlation} Correlations = ...;

tuple AlternativePath{
key int AltId;
{int} subs;}
{AlternativePath} alternatives=...;

int factor[t in Tasks]=0;
int Norm [t in Tasks]=0;
float R[i in 1..NRepPeriods]= ceil(i*OriginalProjectDuration/NRepPeriods);
execute{if(strategy==1){
  for (var t in Tasks){var i=1;while (i<=NRepPeriods){
  if (t.startBaseline<=R[i]){factor[t]= i;break}i=i+1}}}
  else{if(strategy==-1){ for (var t in Tasks){var i=1
  while (i<=NRepPeriods){
  if (t.startBaseline<=R[i]){factor[t]= (NRepPeriods-i+1);break}i=i+1}}}};
  for (var t in Tasks){if (factor[t]==1){Norm[t]=1}else{var i=1
  while (i<=factor[t]){Norm[t]=Norm[t]*(Nbtask-1)+1;i=i+1}}}}

dvar interval task[t in Tasks];
dvar interval hard[t in Tasks] size t.NT;
dvar interval mode[m in Modes] optional;
dvar int x[t in Tasks]in 0..1;
dvar int y[t in Tasks]in 0..1;;
dvar int z[m in Modes]in 0..m.NT;
dvar int crashAmount[t in Tasks];
dvar interval alternatives(r in Alternatives) optional;
dvar interval alternativeFinal;
cumulFunction RsrcUsageL = sum (t in Modes: t.dmdL>0) pulse(mode[t], t.dmdL);
cumulFunction RsrcUsageM = sum (t in Modes: t.dmdM>0) pulse(mode[t], t.dmdM);
dexpr int RsrcUsageE = sum (m in Modes: m.dmdE>0) (m.dmdE * presenceOf(mode[m]));
//Nonnenewable resource
var p = cp.param;
                    cp.param.TimeMode = "CPUTime"
                    cp.param.SearchType = "auto";
                    cp.param.Workers=4;
}
dexpr float overhard1 [i in Tasks]=(i.FS1!=0)?overlapLength(hard[i],hard[<i.FS1>]):0;
dexpr float overhard2 [i in Tasks]=(i.FS2!=0)?overlapLength(hard[i],hard[<i.FS2>]):0;
dexpr float overhard3 [i in Tasks]=(i.FS3!=0)?overlapLength(hard[i],hard[<i.FS3>]):0;
dexpr float overhard4 [i in Tasks]=(i.SS4!=0)?overlapLength(hard[i],hard[<i.SS4>]):0;
dexpr float overhard5 [i in Tasks]= (i.FF5!=0)?overlapLength(hard[i],hard[<i.FF5>]):0;
```

```
dexpr float overlapp1 [i in Tasks]= (i.FS1!=0)?overlapLength(task[i],task[<i.FS1>])-overhard1[i]:0;
dexpr float overlapp2 [i in Tasks]= (i.FS2!=0)?overlapLength(task[i],task[<i.FS2>])-overhard2[i]:0;
dexpr float overlapp3 [i in Tasks]= (i.FS3!=0)?overlapLength(task[i],task[<i.FS3>])-overhard3[i]:0;
dexpr float overlapp4 [i in Tasks]= (i.SS4!=0)?overlapLength(task[i],task[<i.SS4>])-overhard4[i]:0;
dexpr float overlapp5 [i in Tasks]= (i.FF5!=0)?overlapLength(task[i],task[<i.FF5>])-overhard5[i]:0;
dexpr float overlapp6 [i in Tasks]= (i.SF1!=0)?overlapLength(task[i],task[<i.SF1>])-overhard1[i]:0;
dexpr float overlapp7 [i in Tasks]= (i.SF2!=0)?overlapLength(task[i],task[<i.SF2>])-overhard2[i]:0;
dexpr float overlapp8 [i in Tasks]= (i.SF3!=0)?overlapLength(task[i],task[<i.SF3>])-overhard3[i]:0;
dexpr float overlapp9 [i in Tasks]= (i.SS!=0)?overlapLength(task[i],task[<i.SS>])-overhard4[i]:0;
dexpr float overlapp10 [i in Tasks]= (i.FF!=0)?overlapLength(task[i],task[<i.FF>])-overhard5[i]:0;
dexpr float overlappMax[i in Tasks]= maxl(overlapp1 [i],overlapp2 [i],overlapp3 [i],overlapp4
[i],overlapp5 [i])+maxl(overlapp6 [i],overlapp7 [i],overlapp8 [i],overlapp9 [i],overlapp10 [i]);

dexpr int rework [i in Tasks]=
ftoi(round(overlapp1[i]*i.Rework1+overlapp2[i]*i.Rework2+overlapp3[i]*i.Rework3+overlapp4[i]*i.Rewo
rk4+overlapp5[i]*i.Rework5));

dexpr float  Actualrework [i in Tasks]=
(overlapp1[i]*i.Rework1+overlapp2[i]*i.Rework2+overlapp3[i]*i.Rework3+overlapp4[i]*i.Rework4+overl
app5[i]*i.Rework5)/unit;

dexpr float overlappingCost= (sum(t in Tasks)t.Rework1*t.Cost1*overlapp1[t]+ sum(t      in
                                                    Tasks)t.Rework2*t.Cost2*overlapp2[t]+sum(t in
                                                    Tasks)t.Rework3*t.Cost3*overlapp3[t]+ sum(t in
                                                    Tasks)t.Rework4*t.Cost4*overlapp4[t]+sum(t in
                                                    Tasks)t.Rework5*t.Cost5*overlapp5[t])/unit;

dexpr float crashAmount[i in Tasks]= sum(m in Modes: m.taskId==i.id)(m.NT-
z[m])*presenceOf(mode[m]);

dexpr float smax [t1 in Tasks]=crashAmount2[t1]!= 0? (overlappMax[t1]+sum(t in Modes:
t.taskId==t1.id)minl(t.s*crashAmount2[t1],t.CT)* presenceOf(mode[t])):0;

dexpr float crashingCost= sum(i in Tasks)(sum(t in Modes: t.taskId==i.id)(t.NC-
i.NC+t.CS*crashAmount[i])*presenceOf(mode[t]));

 dexpr float substitutionCost= sum(t in Modes)t.NC*presenceOf(mode[t])sum(t in
Modes)t.NC*presenceOf(mode[<t.id,1>]);

dexpr float TotalCost= todateCost + sum(t in Modes)t.NC*presenceOf(mode[t]) -
maxl(OriginalProjectDuration*unit-max(m in Modes)endOf(mode[m])*presenceOf
(mode[m]),0)*Incentive/unit+ Penalty* maxl(max(t in Tasks)endOf(task[t])-Deadline,0)-max(tt in
Tasks) endOf(task[tt])*Indirect/unit + sum(t in Tasks)(t.milecost * maxl(endOf(task[t])-t.mile,0))-sum(t
in Tasks)(t.milebonus * maxl((t.mile)- endOf(task[t]),0))+overlappingCost + crashingCost;

dexpr float scheduleCompressionCost = overlappingCost+crashingCost+ substitutionCost;
```

```
dexpr float netBenefit= maxl(OriginalProjectDuration*unit-max(tt in
Tasks)endOf(task[tt]),0)*Incentive/unit-overlappingCost-crashingCost- substitutionCost;

dexpr float starttimeDeviation[t in Tasks]= (startOf(task[t])>= t.startBaseline)?Norm[t]:0;
dexpr float modeVariation [t in Tasks]= (presenceOf(mode[<t.id,1>])!=
1&&presenceOf(mode[<t.id,2>])!= 1)?Norm[t]:0;
dexpr float scheduleDisruption= sum(t in Tasks)(scheduleVariation[t]+modeVariation[t]);
AlternativeObjectiveFunctions:
        minimize TotalCost;

        maximize netBenefit;

        minimize scheduleCompressionCost;

        minimize max(tt in Tasks) endOf(task[tt]);

        minimize staticLex(TotalCost, max(tt in Tasks) endOf(task[tt]));

        minimize staticLex(max(tt in Tasks) endOf(task[tt]),TotalCost );

        minimize scheduleDisruption;

subject to{

max(tt in Tasks) endOf(task[tt])<=unit*Deadline;

TotalCost<=GivenBudget;

forall (t in Tasks)if (t.compulsory == 1)presenceOf(task[t]);

cModes:

forall (t in Tasks) alternative(task[t], all(m in Modes: m.taskId==t.id) mode[m]);
forall(t in Modes, i in Tasks: i.id==t.taskId){if (t.CT!=t.NT){
z[t]<=t.NT*presenceOf(mode[t])&&z[t]>=(t.NT*(1-x[i])+t.CT*x[i])*presenceOf(mode[t]);
sizeOf(mode[t])==presenceOf(mode[t])*(z[t]+rework[i]);
}else{z[t]==t.NT;sizeOf(mode[t])==(t.NT+rework[i])*presenceOf(mode[t]);
};};
RsrcUsageL <= CapRsrcL;
RsrcUsageM <= CapRsrcM;

RsrcUsageE <= CapRsrcE;

forall (t1 in Tasks){
```

```
if (t1.FS1 != 0)endBeforeStart(task[<t1.FS1>],task[t1], t1.MPL1*y[t1]*unit+t1.lag1*(1-y[t1])*unit);
if (t1.FS2 != 0)endBeforeStart(task[<t1.FS2>],task[t1], t1.MPL2*y[t1]*unit+t1.lag2*(1-y[t1])*unit);
if (t1.FS3 != 0)endBeforeStart(task[<t1.FS3>],task[t1], t1.MPL3*y[t1]*unit+t1.lag3*(1-y[t1])*unit);
if (t1.SS4 != 0)startBeforeStart(task[<t1.SS4>],task[t1],t1.MPL4*y[t1]*unit+t1.lag4*(1-y[t1])*unit);
if (t1.FF5 != 0)endBeforeEnd(task[<t1.FF5>],task[t1], t1.MPL5*y[t1]*unit+t1.lag5*(1-y[t1])*unit);}
forall (t1 in Tasks)(x[t1]+ presenceOf(mode[<t1.id,3>])+ presenceOf(mode[<t1.id,4>])<=1);
forall (t1 in Tasks)(y[t1]+presenceOf(mode[<t1.id,3>])+presenceOf(mode[<t1.id,4>])<=1);
forall(t1 in Tasks)(sizeOf(task[t1])==t1.NT)=> presenceOf(mode[<t1.id,1>]);
forall (t1 in Tasks)(sizeOf(task[t1])>=smax [t1]);

forall (t1 in Tasks){
if (t1.FS1 != 0)endBeforeStart(hard[<t1.FS1>],hard[t1],t1.lag1);
if (t1.FS2 != 0)endBeforeStart(hard[<t1.FS2>],hard[t1],t1.lag2);
if (t1.FS3 != 0)endBeforeStart(hard[<t1.FS3>],hard[t1],t1.lag3);
if (t1.SS4 != 0)startBeforeStart(hard[<t1.SS4>],hard[t1],t1.lag4);
if (t1.FF5 != 0)endBeforeEnd(hard[<t1.FF5>],hard[t1],t1.lag5);}
min (t in Tasks)startOf (hard[t])==0&&max (t in Tasks)endOf (hard[t])<=OriginalProjectDuration;}
forall(r in alternatives: r.AltId==j) forall (i in r.subs)
presenceOf(alternatives[<r,j>] =>presenceOf(tasks[<i>]);
alternative(alternativeFinal, alternatives);
forall(i in Correlations){
presenceOf(mode[<i.taskId1, i.id1>]) => presenceOf(mode[<i.taskId2, i.id2>]);
presenceOf(mode[<i.taskId2, i.id2>]) => presenceOf(mode[<i.taskId1, i.id1>]);}}
min(tt in Tasks) startOf(task[tt])==Todate;
forall (t in Tasks){if (t.ongoing==1){startOf(task[t])==Todate;};}
float taskOriginalDuration[i in Tasks]=(i.NT);

float scheduleDuration= max(tt in Tasks) endOf(task[tt]);

float OptimumDegreeOfCompression= scheduleDuration/OriginalProjectDuration*100;

execute {

writeln("                                                    ");

writeln("taskOriginalDuration= ",taskOriginalDuration);

writeln("crash= ",crashAmount,"days");

writeln("Rework= ",Actualrework,"days");

writeln("Overlap= ",overlapp1,overlapp2,overlapp3,overlapp4,overlapp5);

writeln("OptimumDegreeOfCompression= ",OptimumDegreeOfCompression,"%");

writeln("TotalCost= ",TotalCost);
```

```
writeln("crashingCost= ",crashingCost);

writeln("overlappingCost= ",overlappingCost);

writeln("netBenefit= ",netBenefit);

writeln("scheduleDuration= ",scheduleDuration);

for (var m in Modes) {
if (mode[m].present){
writeln( m.taskId + "     " + m.id + "    " + mode[m].size + "      " + mode[m].start +"      " +
mode[m].end);
}}}
```