Optimized Scheduling of Repetitive Construction Projects under Uncertainty

Ibrahim Bakry

A thesis in the

Department

of

Building, Civil and Environmental Engineering

Presented in Partial Fulfillment of the Requirements

For the Degree of

Doctor of Philosophy (Building Engineering) at

Concordia University

Montreal, Quebec, Canada

June 2014

CONCORDIA UNIVERSITY

School of Graduate Studies

This is to certify that the thesis prepared

By: **Ibrahim Bakry**

Entitled: **Optimized Scheduling of Repetitive Construction Project under Uncertainty**

and submitted in partial fulfillment of the requirements for the degree of

Doctor of Philosophy (Building Engineering)

complies with the regulations of the University and meets the accepted standards with respect to originality and quality.

Signed by the final examining committee:

________________________________Chair

Dr. Constantinos Constantinides

________________________________External Examiner

Dr. Janaka Y. Ruwanpura

________________________________External to Program

Dr. A. Dolatabadi

________________________________Examiner

Dr. A. Hammad

________________________________Examiner

Dr. Z. Zhu

________________________________Thesis Co-Supervisor

Dr. O. Moselhi

________________________________Thesis Co-Supervisor

Dr. T. Zayed

Approved by ________________________________

Dr. M. Elektorowicz

Graduate Program Director

June 16th 2014

____

Dr. C. Trueman, Interim Dean

Faculty of Engineering and Computer Science

Abstract

Optimized Scheduling of Repetitive Construction Projects under Uncertainty

Ibrahim Bakry, Ph.D. Candidate
Concordia University, 2014

Uncertainty is an inherent characteristic of construction projects. Neglecting uncertainties associated with different input parameters in the planning stage could well lead to misleading and/or unachievable project schedules. Many attempts have been made in the past to account for uncertainty during planning for construction projects and many tools and techniques were presented to facilitate modelling of such uncertainty. Some of the presented techniques are widely accepted and used frequently like Project Evaluation and Review Technique (PERT) and Monte Carlo Simulation, while others are more complicated and less popular, such as fuzzy set-based scheduling. Although accounting for uncertainty has been a topic of interest for more than four decades, it was rarely attempted to account for uncertainty when scheduling repetitive construction projects. Repetitive projects impose an additional challenge to the already complicated construction scheduling process that accounts for the need to maintain crew work continuity throughout project execution. This special characteristic necessitates producing scheduling techniques specifically suited to resource driven scheduling.

Therefore, the main objective of this research is to produce a comprehensive scheduling, monitoring and control methodology for repetitive construction projects that is capable of accounting for uncertainties in various input parameters, while allowing for optimized acceleration and time-cost trade-off analysis. The proposed methodology encompasses three integrated models; Optimized Scheduling and Buffering Model, Monitoring and Dynamic Rescheduling Model and Acceleration Model. The first model presents an optimization technique that accounts for uncertainty in input parameters. It employs a modified dynamic programming technique that utilizes fuzzy set theory to model uncertainties. This model includes a schedule defuzzification tool and a buffering tool. The defuzzification tool converts the optimized fuzzy schedule into a deterministic one, and the buffering tool utilizes user's required level of confidence in the produced schedule to build and insert time buffers, thus providing protection against anticipated delays affecting the project. The Monitoring and Dynamic Rescheduling Model capitalizes on the repetitive nature of these projects, by using actual progress on site to reduce uncertainty in the remaining part of the schedule. This model also tracks project progress through comparing the actual buffer consumption to the planned buffer consumption. The Acceleration Model presents an iterative unit based optimized acceleration procedure. It comprises a modified algorithm for identifying critical units of the project to accelerate. This model presents queuing criteria that accounts for uncertainty in additional cost of acceleration and for contractor's judgment in relation to prioritizing critical units for acceleration. Moreover, this model offers six strategies for schedule acceleration and maintains crew work continuity.

Together, the three developed models offer an integrated system that is capable of accounting for uncertainty in different variables through different project stages, aiming at helping managers keep repetitive construction projects on track. The presented optimization technique is automated in an Object Oriented program; coded in C# programming language. A number of case studies are analyzed and presented to demonstrate and validate the capabilities and features of the presented methodology.

Acknowledgement

All praise and gratitude be to Allah, for it is only by his grace that good deeds are accomplished.

I would like to express my deepest gratitude to my mentors, Dr. Moselhi and Dr. Zayed. The years I spent learning from them are years that I will continue to cherish for the rest of my life. Their support and encouragement have often been indispensable. Their continuous guidance and advice on the academic and personal level is something I dearly treasure.

I am grateful to my examining committee members, Dr. Janaka Ruwanpura, Dr. Amin Hammad, Dr. Zhenhua Zhu and Dr. Ali Dolatabadi. Your comments and feedback played a notable role in bringing my work to this level.

I want to thank my colleagues at the Automation and Construction Laboratory at Concordia University, Ahmed Eweda, Hani Alzeraiee, Ali Montaser, Mona Abuhamad, Sasan Golnaraghi, Ahmed Atef, Magdy Omar and Laya Sedghy I would also like to thank Ibrahim Mashhour, Mohamed Gomaa and Ihab ElAghoury for their help and support.

I want to extend my gratitude to the members of the Gezira Sporting Club Taekwondo Team, your encouragement, although usually coming disguised as sarcasm, is much appreciated.

To my beloved mother, my inspiring father, my dear wife and to my best friends, Mariam and Abdullatif

Table of contents

LIST OF FIGURES XII

LIST OF TABLES XV

LIST OF ACRONYMS XVII

CHAPTER 1: INTRODUCTION 1

1.1 Repetitive Construction Projects 1

1.2 Scheduling with Uncertainty 3

1.3 Problem Statement 5

1.4 Research Objectives 5

1.5 Methodology Overview 6

1.6 Thesis Organization 8

CHAPTER 2: LITERATURE REVIEW 9

2.1 Introduction 9

2.2 Scheduling of construction Projects 9

- 2.2.1 Scheduling of Repetitive Construction Projects 10
- 2.2.2 Optimized Scheduling of Repetitive Construction Projects 18

2.3 Using Buffers for Schedule Protection 25

2.3.1 Buffer Sizing Techniques Based on General representation of Uncertainty 32
2.3.2 Buffer Sizing Techniques Addressing Specific Factors 38

2.4 Schedule Acceleration 45

2.5 Findings of Literature Review 51

CHAPTER 3: METHODOLOGY 54

3.1 Introduction 54

3.2 Optimized Scheduling and Buffering Model 56

3.2.1 Modelling Uncertainty 59
3.2.2 Fuzzy Dynamic Programming 61
3.2.3 Schedule Defuzzification and Buffering 66
3.2.4 Buffer Sizing and Insertion 68

3.3 Monitoring and Dynamic Rescheduling Model 80

3.3.1 Updating Activities 82
3.3.2 Reducing Uncertainty 83
3.3.3 Tracking Buffers 86

3.4 Acceleration Model 94

3.4.1 Identifying Activities to Accelerate 95
3.4.2 Uncertainty in Additional Cost 99
3.4.3 Contractors' Judgment 100
3.4.4 Schedule Acceleration 101

CHAPTER 4: IMPLEMENTATION OF SCHEDULE OPTIMIZATION 104

4.1 Introduction 104

4.2 Design **105**

4.2.1 Project Class 106

4.2.2 Activity Class 107

4.2.3 Crew Class 108

4.2.4 Fuzzy Numbers 109

4.3 OSRP **110**

CHAPTER 5: CASE STUDIES **113**

5.1 Introduction **113**

5.2 Optimized Scheduling and Buffering **114**

5.2.1 Least Cost Optimization 117

5.2.2 Least Duration Optimization 129

5.2.3 Defuzzification and Buffering 130

5.3 Schedule Acceleration **134**

5.3.1 Case Study I 134

5.3.2 Case Study II 142

5.4 Results Analysis and Findings **148**

5.4.1 Optimized Scheduling and Buffering 148

5.4.2 Schedule Acceleration 152

CHAPTER 6: CONCLUSION **160**

6.1 Summary and Concluding Remarks **160**

6.2 Research Contributions **163**

6.3 Research Limitations 164

6.4 Opportunities for Future Work 165

6.4.1 Improving Presented Methodology: 165

6.4.2 Extending Presented Methodology: 166

REFERENCES.......... 167

List of Figures

Figure 1.1: Typical and Non-Typical Repetitive Activities 2

Figure 1.2: Research Methodology Overview 7

Figure 2.1: Scheduling of Construction Projects 10

Figure 2.2: LOB and LSM Schedule 13

Figure 2.3: F-RSM Schedule (Maravas and Pantouvakis 2011) 17

Figure 2.4: Student Syndrome (Robinson and Richards 2010) 26

Figure 2.5: Conventional schedule and CC/BM Schedule (Raz, Barnes et al. 2003) 28

Figure 2.6: Project Network with Feeding Buffer Identified (Raz, Barnes, et al. 2003) .. 29

Figure 2.7: Existing Buffer Sizing Techniques 31

Figure 2.8: Safety Time $st_{(j)}$ of Activity j (Long and Ohsato 2008) 37

Figure 2.9: Effect of Acceleration on Repetitive Activities (Bakry et al. 2014) 48

Figure 2.10: Acceleration by Relaxing Converging Activities (Bakry et al. 2014) 49

Figure 3.1: Proposed Methodology Main Components 55

Figure 3.2: Optimized Scheduling and Buffering Model 58

Figure 3.3: General Triangular Fuzzy Number 60

Figure 3.4: Using 3 Values to Represent a Deterministic Number 60

Figure 3.5: Fuzzy Dynamic Programming Flowchart 65

Figure 3.6: Intersection Area of a Fuzzy Number and a Deterministic Number............. 70

Figure 3.7: Buffer Sizing 72

Figure 3.8: Buffer Insertion - Case 1 74

Figure 3.9: Buffer Insertion - Case 2 75

Figure 3.10: Buffer Insertion - Case 3 76

Figure 3.11: Buffer Insertion - Case 4 77

Figure 3.12: Buffer Sizing and Insertion Flowchart 80

Figure 3.13: Main Components of Monitoring Model 82

Figure 3.14: Original Fuzzy Duration, EV and Buffer 86

Figure 3.15: Duration with Reduced Fuzziness, EV and Buffer 86

Figure 3.16: Flowchart for Monitoring and Dynamic Rescheduling....................... 92

Figure 3.17: BCI Thresholds over Project Duration 93

Figure 3.18: Identifying Least Aligned Activities 98

Figure 3.19: Flowchart for Acceleration Model 103

Figure 4.1: Input and Output Overview 106

Figure 4.2: Instances Derived from FuzzyNumber Class 109

Figure 4.3: Project Input Screen 111

Figure 4.4: Activities and Crews Input Screen 112

Figure 4.5: Output Report 112

Figure 5.1: Three Span RC Bridge (Selinger 1980) 114

Figure 5.2: Least Cost Schedule Generated with Deterministic Input........................... 119

Figure 5.3: Defuzzified Least Cost Schedule 124

Figure 5.4: Defuzzified Least Duration Schedule.. 129

Figure 5.5: Defuzzified Least Cost Schedule with Buffers 134

Figure 5.6: Original Schedule (Case Study I).. 138

Figure 5.7: Original Schedule before Acceleration 143

Figure 5.8: Scenario 1 Accelerated Schedule 144

Figure 5.9: Scenario 2 Accelerated Schedule 146

Figure 5.10: Scenario 3 Accelerated Schedule 148

Figure 5.11: Optimization Results Summary.. 152

Figure 5.12: Locating Least Cost Plan for Different Scenarios (for Case Study I)........ 153

Figure 5.13: Locating Least Cost Plan for Different Scenarios (for Case Study II)....... 156

Figure 5.14: Results of 3 Scenarios (Case Study II).. 156

Figure 5.15: Original Schedule Vs Scenario 1 (Case Study II) 157

Figure 5.16: Original Schedule Vs Scenario 2 & 3 (Case Study II) 158

List of Tables

Table 2.1: Buffer Size for Different Activities and Safety Levels (Shou and Yeo 2000) 35

Table 2.2: Buffer Sizing Based on CV (Rezaie et al.'s effort 2009) 35

Table 2.3: Summary of Factors Previously Included in Buffer Sizing............................ 45

Table 3.1: Initial Input for Scheduling Module 59

Table 4.1: Main Data Members of Project Class.. 106

Table 4.2: Main Function Members of Project Class 106

Table 4.3: Main Data Members of Activity Class 107

Table 4.4: Main Function Members of Activity Class 108

Table 4.5: Main Data Members of Crew Class... 108

Table 4.6: Main Function Members of Crew Class... 109

Table 5.1: Original Activity Quantities 115

Table 5.2: Available Crews for Each Activity.. 115

Table 5.3: Original Crews Productivity... 116

Table 5.4: Project Direct Cost Components 116

Table 5.5: Least Cost Optimization (Verification run).. 120

Table 5.6: Factors for Fuzzifying Original Case Study 123

Table 5.7: Fuzzy Numbers after Applying Factors.. 123

Table 5.8: Least Cost Optimization (Under Uncertainty) 125

Table 5.9: Least Duration Optimization 127

Table 5.10: DurEV and DurAI 132

Table 5.11: Deterministic Schedule with Buffers 132

Table 5.12: Total Project Cost 133

Table 5.13: Crews Productivity 135

Table 5.14: Activities Quantities and Units Sequence 136

Table 5.15: Baseline Schedule 137

Table 5.16: Overtime Cost for Case Study I 139

Table 5.17: Fuzzy Overtime Cost for Case Study I 142

Table 5.18: Overtime Cost for Case Study II 144

Table 5.19: Assigned Contractor's Judgment Values for Case Study II 145

Table 5.20: Factors for Fuzzifying Acceleration Cost for Case Study II 147

Table 5.21: Fuzzy Overtime Cost for Case Study II 147

Table 5.22: Optimization Results Summary 151

Table 5.23: Results of Different Acceleration Scenarios (for Case Study I) 153

Table 5.24: Results of Different Acceleration Scenarios (for Case Study II) 156

List of Acronyms

PERT	Program Evaluation and Review Technique
CPM	Critical Path Method
LOB	Line of Balance
RSM	Repetitive Scheduling Method
LSM	Linear Scheduling Method
F-RSM	Fuzzy Repetitive Scheduling Method
OOP	Object Oriented Programming
SQS-AL	Sequence Step Algorithm
GA	Genetic Algorithm
CAD	Computer Aided Design
NPV	Net Present Value
MARR	Minimum Attractive Rate of Return
C&PM	Cut and Paste Method
RD	Relative Dispersion
RCPSP	Resource Constrained Project Scheduling Problem
APRT	Adaptive Procedure Resource Tightness
APD	Adaptive Procedure with Density
STI	Stochastic Task Insertion

FLBM	Fuzzy Logic Buffering Model
FST	Fuzzy Set Theory
AI	Agreement Index
EV	Expected Value
COA	Center of Area
BC	Buffer Consumption
PBC	Planned Buffer Consumption
BCI	Buffer Consumption Index
CAP	Controlling Activity Path
OSRP	Optimized Scheduler of Repetitive Projects

Chapter 1: Introduction

1.1 Repetitive Construction Projects

Repetitive construction projects are identified as construction projects formed of recurring units. Those recurring units are similar work stations or locations each consisting of the same, usually small, number of sequential activities. There are two main types of repetitive projects, linear and non-linear. Linear repetitive projects are characterized by having a linear geometric layout, such as highways, pipeline laying and railroad construction projects. On the other hand, non-linear repetitive projects don't have a linear geographical layout, such as multistory high-rise buildings and typical housing projects (Arditi and Albulak 1979). A construction project may consist of both repetitive and non-repetitive activities simultaneously. An example of this case is constructing a high-rise multistory building, where foundation works are non-repetitive activities performed only once at the beginning of the project, while casting concrete for typical floors is a repetitive activity.

A different classification of repetitive projects is dividing them into typical and non-typical repetitive projects (Vorster and Bafna 1992). Typical repetitive projects consist of activities having same quantity of work for each unit, and utilize resources having same productivity for each unit. This leads to a repetitive schedule formed of activities represented by straight lines with a constant slope. However, the general case is that projects consist of non-typical units. Non-typical units have different quantities for each unit for the same activity, and/or utilize crews and equipment operating with different productivities. Repetitive projects' schedules consisting of non-typical activities are represented by broken lines with varying slopes. Figure 1.1 shows how typical and non-typical activities are represented differently.

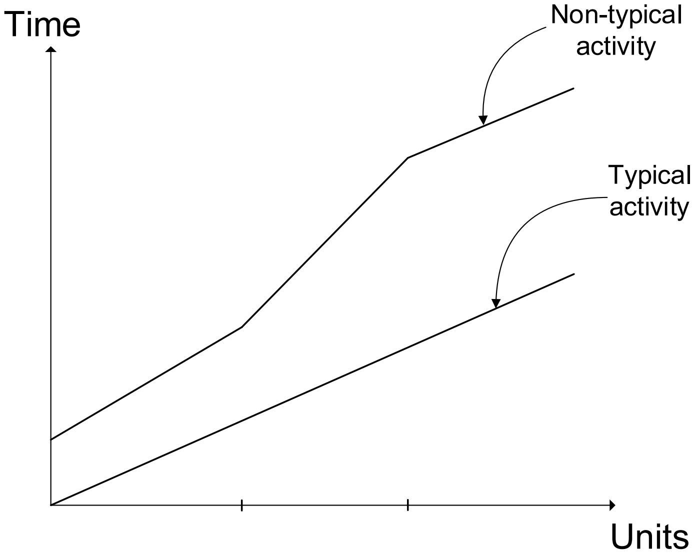


Figure 1.1: Typical and Non-Typical Repetitive Activities

The unique repetitive nature of repetitive construction projects paves the way for making considerable savings on time and cost. By maintaining the continuity of different crews and resources working on different activities in this type of projects several benefits could be achieved, such as maintaining a constant workforce by reducing firing and hiring of labour, retaining skilled labour, maximizing the use of learning curve effect and minimizing equipment idle time (Hassanein 2003). However, maintaining resource continuity forms an additional constraint when planning and managing a repetitive project. Using traditional (non-repetitive) scheduling and planning tools and techniques, such as CPM, to manage repetitive projects has been widely criticized (Reda 1990, Wong

1993, Hegazy and Wassef 2001, Arditi et al. 2002, and Hegazy and Kamarah 2008); CPM techniques do not provide means for scheduling while maintaining resource work continuity, they do not display productivity rates or activities relative locations, and they make it very complicated to view and manage the produced schedule. This calls for developing and utilizing special tools and techniques to properly plan and manage repetitive projects.

1.2 Scheduling with Uncertainty

Construction projects are complex projects that take place in dynamic environments. Accounting for different sources of uncertainty during the planning stage is essential to successful delivery of construction projects. Although many scheduling techniques for repetitive and traditional projects utilize deterministic input for different parameters such as quantities, productivity rates, costs and other input variables, it is safe to say that many of these numbers are subject to a certain amount of uncertainty. Failing to correctly account for uncertainty affecting a certain project could well lead to producing an unrealistic or misleading schedule. As early as the 1950's researchers have started producing scheduling techniques that address uncertainty in the schedule input. Many scheduling techniques have been produced to accommodate uncertainty when scheduling traditional projects, examples of which are PERT, fuzzy set theory and Monte-Carlo simulation (Zadah 1965, Carr 1979, Ayyub and Haldar 1984, Senior 1993, and Lorterapong 1995). However, the case is different when it comes to scheduling repetitive projects, where uncertainty is mostly not accounted for. Therefore, there is a need for the development of a comprehensive yet easy to use scheduling technique for repetitive construction projects that would efficiently account for uncertainty.

Methodological sizing and insertion of time buffers in a time schedule to protect against different delays is a relatively new approach that still has a lot of grounds to cover. Since 1997, Goldratt presented the Critical Chain scheduling technique for project management. This technique attributes the general unsatisfactory performance of projects' schedules to two main human behaviors, namely student's syndrome and Parkinson law (Goldratt 1997). Student's syndrome suggests that if a task in a project is assigned a longer duration than it strictly needs, then teams responsible for execution will automatically start late and/or work at a relaxed rate, as they realize they have more time than they need. Parkinson law suggests that if execution teams manage to actually complete any task in duration shorter than scheduled, they will not report early completion of the task. Critical Chain scheduling attempts to account for uncertainty while avoiding the two mentioned pitfalls by assigning tasks' duration strictly as calculated, without adding any additional durations for contingency, and inserting separate time buffers in strategic places in the schedule to mitigate delays that might take place due to various reasons. Critical Chain as a scheduling technique did receive a lot of criticism as will be discussed in the coming chapter, however, it served as an eye opener that drew attention to the high potential of protecting schedules by utilizing time buffers. Consequently, many researchers attempted capitalizing on the concept of protecting schedules against uncertainty by inserting time buffers. Many different buffers sizing and insertion techniques were presented with varying degrees of success.

1.3 Problem Statement

After reviewing existing literature, with a focus on literature built to closely address the needs of the industry, it has been established that current practices reveal shortage of existing tools and techniques specifically tailored for optimized scheduling of construction projects; that account for schedule acceleration and time-cost trade-off analysis. Practitioners managing repetitive projects, which form a considerable market share in the construction industry, often manage their projects using tools designed for traditional projects. Such tools are not independently capable of addressing the basic needs of maintaining resource continuity, balancing rates and schedule visualization. Such needs are either met manually or through continuous transfer of project data between different software packages; thus, consuming time and effort and opening the door for an increased amount of errors and mistakes. Accordingly, there is a pressing need for closing the gap between the needs of managers of repetitive projects and available tools and techniques.

1.4 Research Objectives

The main objective of this research has been identified as follows: to produce a comprehensive methodology for managing different phases of repetitive construction projects while accounting for uncertainty. To achieve this general objective, the following several sub-objectives were identified:

1. Develop an optimization model for repetitive construction projects that is capable of accounting for uncertainty in different input parameters, without compromising crew work continuity.
2. Build a model for sizing and inserting time buffers to provide protection to the schedule against anticipated delays.
3. Develop an optimized automated acceleration model for repetitive projects that accounts for uncertainty and for queuing criteria practically accounted for in the industry.

1.5 Methodology Overview

Figure 1.2 shows the steps followed during this research. The research began by establishing a general problem statement that formed the main motivation for this research. An extensive review of literature was performed, mainly addressing areas of scheduling, optimizing and accelerating repetitive projects, and different ways to address uncertainty in the construction industry. After the gaps of existing literature were clearly identified, a set of research objectives were established. Then, the model's different components were built to address the two phases of the industry, the pre-construction and the construction phases. The model was verified and evaluated through analyzing different case studies, and a software prototype was built to better showcase the developed model and its features. Finally, conclusions were drawn, limitations were stated and opportunities for future work were identified.

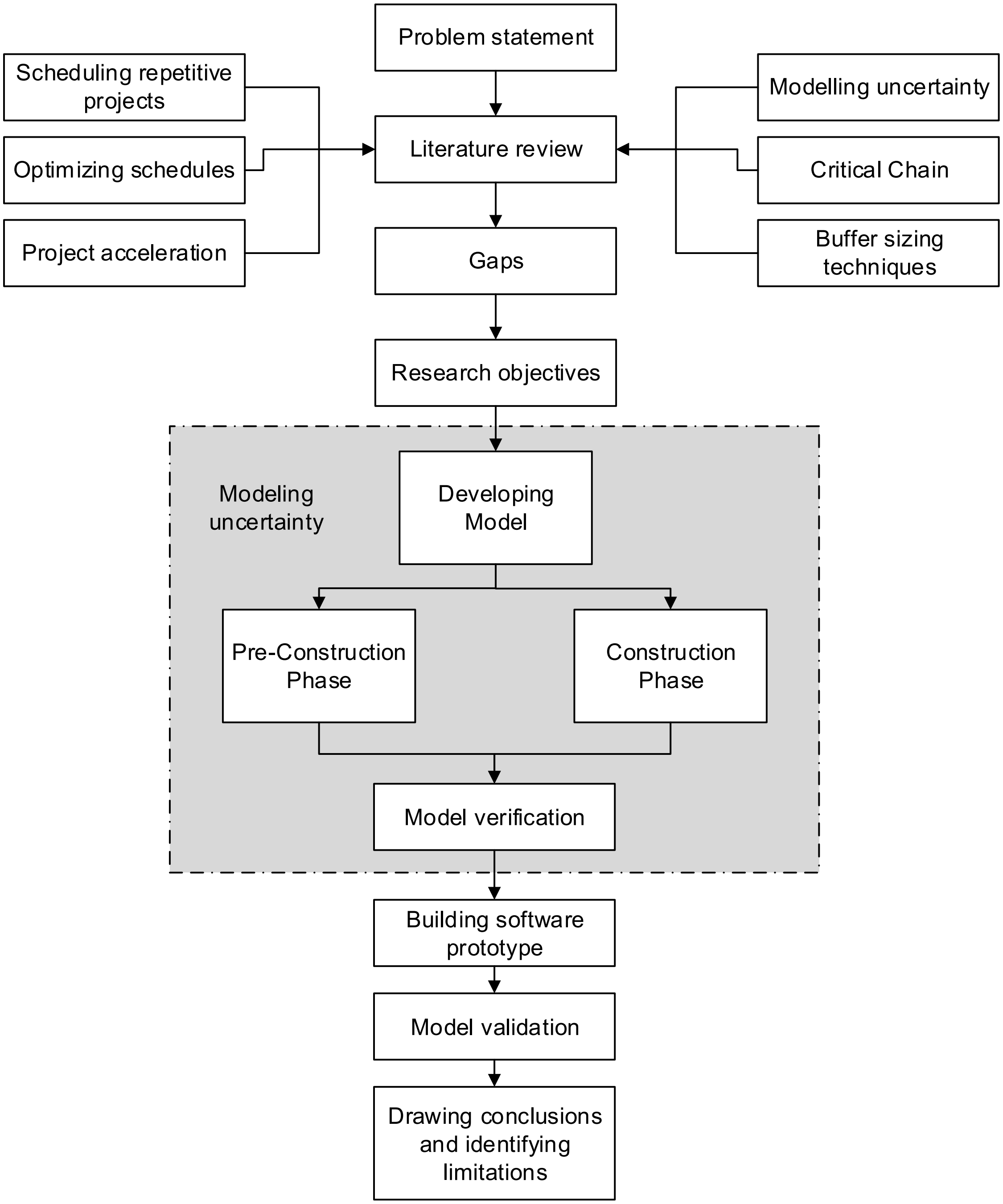


Figure 1.2: Research Methodology Overview

1.6 Thesis Organization

This thesis is presented in six chapters. The second chapter presents a comprehensive literature review. It focused on existing tools and techniques for optimized scheduling of repetitive projects, addressing uncertainty and schedule acceleration. Finally the chapter ends by highlighting the identified gaps in the literature.

The third chapter explains in detail the developed methodology. Following its introduction, the third chapter comprises three main parts; the first part addresses the Optimized Scheduling and Buffering Model, the second part explains the Monitoring and Dynamic Rescheduling Model, and finally the third part describes the schedule Acceleration Model. Chapter four discusses the developed software prototype, how was it designed and implemented and its main features.

Chapter five explains the analyzed case studies. It explains the components of each case and how are the developed models utilized to analyze it. The chapter analyzes the results to evaluate the performance of the developed models. Finally chapter six presents a summary of the thesis, highlights the developments made in the thesis; highlighting the research contributions and limitations. It also lists a set of proposed opportunities for future work.

Chapter 2: Literature Review

2.1 Introduction

Scheduling construction projects is an area that received a lot of attention over the years. This is mainly attributed to the fact that a schedule is the road map to the successful delivery of any construction project. This chapter presents a comprehensive literature review of construction scheduling tools and techniques divided into four main parts.

- The first part reviews literature of the scheduling domain, with more emphasis on optimized scheduling of repetitive construction projects; which is the main focus of this research.
- The second part reviews literature related to different buffer sizing and insertion techniques as an approach to address uncertainty in construction projects.
- The third part reviews existing algorithms for project acceleration and their main features and limitations.
- The fourth and final part of this chapter summarizes the reviewed literature and highlights its gaps and limitations.

2.2 Scheduling of construction Projects

Due to the extensive amount of research performed addressing this area, a classification of scheduling techniques had to be followed to allow for better understanding of the existing tools and techniques and their limitations. In this context, scheduling of construction projects is divided into two main types, scheduling for traditional (non-repetitive) projects and scheduling for repetitive projects. A further subdividing of each of these two main types is the division into deterministic and non-deterministic schedules. This classification is displayed in a hierarchical form in Figure 2.1. Scheduling repetitive construction projects is the main domain that this research addresses.

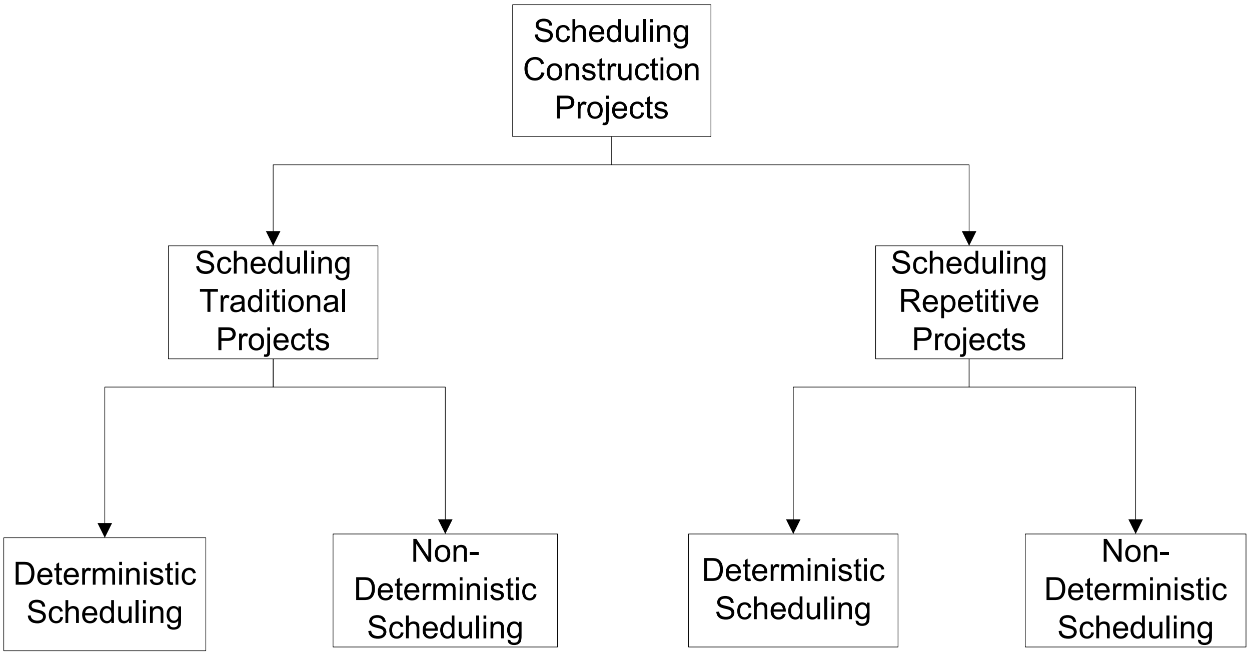


Figure 2.1: Scheduling of Construction Projects

2.2.1 Scheduling of Repetitive Construction Projects

In spite of the big number of scheduling techniques developed for scheduling traditional (non-repetitive) projects offering different features and having different capabilities, none of them is suited for scheduling repetitive projects as they don't address the need to maintain crew work continuity. The nature of repetitive projects entails depending on resource driven scheduling techniques that would allow maintaining resource work continuity and hence benefiting from the repetitive nature of these projects. Another factor supporting not using traditional scheduling techniques is that they tend to be too complicated to visualize when used to schedule repetitive projects, and they lack the means to display activities productivity and locations (Reda 1990, Wong 1993, Hegazy and Wassef 2001, and Arditi et al. 2002). For example a small repetitive project consisting of 100 units each including 5 sequential activities would produce a 500 activity network, which is relatively more complicated to schedule and visualize when compared to any repetitive scheduling technique. Examining the existing literature revealed that there are fewer established techniques for scheduling repetitive projects, in comparison to those serving traditional projects. Furthermore, there is a very limited number of techniques addressing the matter of uncertainty. Following what is displayed in Figure 2.1, repetitive projects schedules were grouped into deterministic and non-deterministic schedules.

2.2.1.1 Deterministic Scheduling of Repetitive Projects

Most of the techniques developed to schedule non-repetitive projects address projects consisting of repetitive activities only, while a fewer number of techniques can accommodate projects containing both repetitive and non-repetitive activities (El-Rayes 1997). Techniques addressing only repetitive activities are all considered variations of the line of balance (LOB) technique, while techniques addressing both types of activities are considered adaptations of the linear scheduling method (LSM).

LOB was first developed by U.S Navy to schedule and monitor manufacturing operations since as early as 1942 (Lumsden 1968). Later in 1966 LOB was adapted to schedule repetitive construction projects. LOB represents a project schedule on a two axis grid, where the horizontal axis represents the time and the vertical axis represents the repetitive units. Each activity is plotted as a bar with its width proportional to the duration each task requires for each unit, and the inclination of the bar represents the production rate of that activity. A comparison was carried out by Arditi and Albulak (1979 and 1986), where a repetitive highway project was scheduled once with CPM and once with LOB. It concluded that scheduling repetitive projects with LOB requires less time and effort, provides a better view of the project and enables producing smoother flow of working crews. However, a disadvantage was reported, which is that LOB fails to clearly represent overlapping activities having the same rate. Literature reveals that many versions of LOB were presented, all adopting the same concepts (Lumsden 1968, Carr and Meyer 1974, Arditi and Albulak 1979, Albulak 1986, and AlSarraj 1990). Nevertheless, LOB has been criticized in literature for more than one reason. The main disadvantage which is thought to limit the spread of such a technique is its failure to schedule big scale repetitive projects and that its use is limited to simple projects (Kavanagh 1985). In addition to that, LOB fails to account for the effect of learning curve which has a considerable impact on some repetitive projects (Ashley 1980), and it also doesn’t have the ability to represent non-typical repetitive activities (Moselhi and El-Rayes 1993).

The other well-established scheduling technique used for scheduling repetitive construction projects is LSM. Although similar to LOB there are few differences in application and in graphical representation of the schedule. In application LSM has the ability to represent repetitive and non-repetitive activities, while graphically the difference is that in LSM the horizontal axis represents the repetitive units while the vertical axis represents time, also in LSM activities are represented by lines instead of bars. Figure 2.2 illustrates the difference between LOB and LSM.

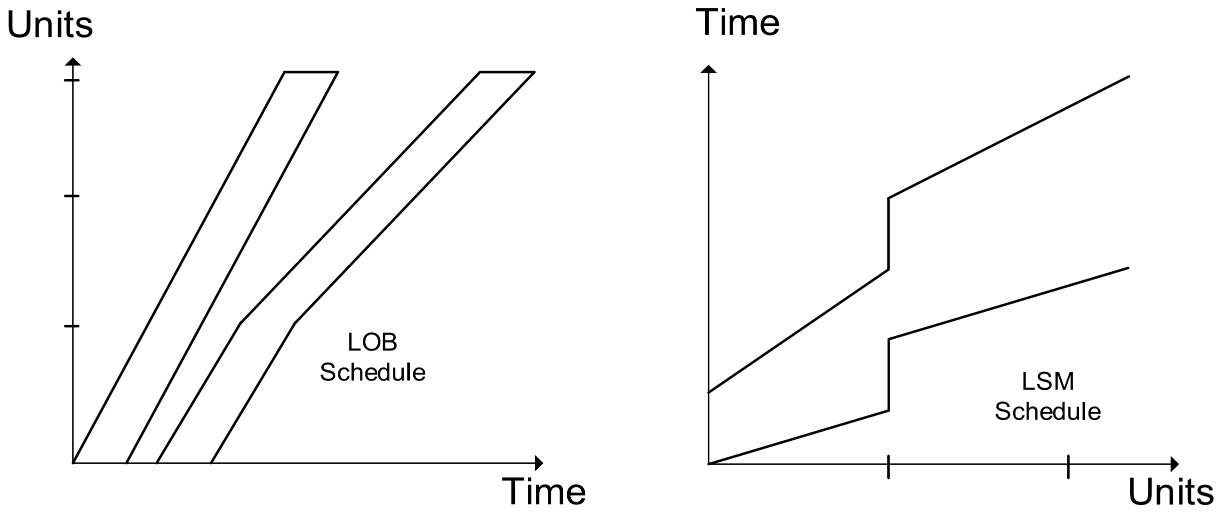


Figure 2.2: LOB and LSM Schedule

Main advantages of LSM include its ability to deliver a complex project in a simpler way, and that it is capable of representing non-repetitive and non-typical activities. However a comparison between LSM and CPM through applying both to a roadway project revealed some disadvantages of LSM (Chrzanowski and Johnston1986) including the fact that LSM is essentially a graphical technique that cannot be readily implemented in numerical computerization like network techniques.

LOB and LSM can be identified as the two main platforms utilized for scheduling repetitive projects, many efforts used them as a base for more evolved scheduling techniques encompassing different features. The following section discusses scheduling techniques that are capable of non-deterministic scheduling, while its successor section focuses on techniques optimizing repetitive projects schedules.

2.2.1.2 Non-Deterministic Scheduling of Repetitive Projects

Although non-deterministic scheduling better captures the dynamic construction environment, it didn't receive as much attention as deterministic scheduling. Exploring the existing literature revealed that there are very few efforts done to schedule repetitive projects while accounting for uncertainty in calculating activities durations. Most of these efforts utilize simulation models. However, in most cases, simulation cannot be utilized by itself to schedule while maintain resource continuity, it has to be accompanied by an external algorithm to cover this shortage of simulation (Lutz 1990 and Yang 2002).

In one of the earliest attempts, Ashley (1980) used simulation to schedule repetitive projects. Ashley's model didn't depend on any external algorithm and had no means of eliminating idle time, his model ended up providing the least possible duration schedule without accounting for resource continuity, thus, yielding a scheduling technique not recommended for repetitive projects. Other simulation based efforts included Kavanagh's (1985) model "SIREN", where he utilized queuing theory to schedule repetitive projects. He proposed two queues, one for ready activities and one for unready activities, depending on resource availability. Through a prioritizing algorithm that checks precedence relations and job logic, activities move from the unready queue to the ready queue to be executed. Similar to Ashley's (1980) work, "SIREN" also utilizes stochastic input and doesn't guarantee resource continuity in the produced schedules.

Simulation based efforts started advancing by utilizing CYCLONE (Halpin 1973). Lutz and Halpin (1992) used CYCLONE to determine mean activity durations, and then used these durations as an input to LOB. Later Wilson (1994) further expanded to include learning curves, in search for a more realistic stochastic schedule. Further trials included PICASO presented by Senior and Halpin (1994) and SimCon (Chehayeb and Abourizk 1998), both offering simulation related improvements, however, all previous works couldn't efficiently maintain resource continuity. Polat et al. (2009) incorporated discrete event simulation with LOB to schedule highway projects. The authors produced an initial deterministic schedule for 4 repetitive activities comprising the highway project, and then developed a simulation model for activities performed by trucks using stochastic historical data, to test whether the used trucks fleet configuration would comply to work continuity established by the initial LOB schedule. The produced model is simple and easy to use, yet it used simulation for a limited purpose after the initial scheduling was complete. Expanding this approach to simulate all activities in a project could be more comprehensive.

A more recent effort also utilizing simulation is the sequence step algorithm (SQS-AL) presented by Srisuwanrat (2009), for scheduling repetitive activities with stochastic durations. SQS-AL works by dividing the project into sequential time frames, each frame containing a group of activities that could be executed simultaneously, and each time frame can be scheduled only after all its predecessors are scheduled. In the first step the activities in the first time frame are simulated using Monte-Carlo simulation, the resulting durations are grouped in histograms, and then based on a user input confidence level the activities durations are extracted from the durations' histogram and assigned to the schedule. Then the activities in the next sequence step are simulated and their durations are similarly identified and added to the schedule until all activities are scheduled. Srisuwanrat (2009) also presented ChaStrobe, an application that allows using genetic algorithm as a search method that can be run between consecutive sequence steps to search for near optimum schedules, whether they be of optimum duration or optimum cost.

Although the few simulation based techniques found in literature have the ability to be modified to answer different requirements of different planners, however, they share the need for sufficient amounts of historical data, which is rarely available. Moreover, they provide solutions that are in agreement with the historical data utilized as input, which naturally covers uncertainty factors that has affected projects used for collecting historical data, but they don't provide a schedule specifically protected against uncertainty factors that will particularly affect the project at hand. In other words, some of the historical data utilized was from a project harmed by severe weather conditions, or from a project that suffered significant delays due to labour problems, while the project at hand might be mainly affected by other sources of uncertainty.

To overcome the limitation of complete dependency on historical data, Maravas and Pantouvakis (2011) adopted a different approach. In this approach, the authors tried to account for uncertainty in crews' productivity, which is consequently reflected on the activities durations. Maravas and Pantouvakis based their scheduling algorithm, called F-RSM, on deterministic calculations, but when it came to final schedule representation each activity was represented by three lines corresponding to the three values of fuzzy productivity (a,b,c).

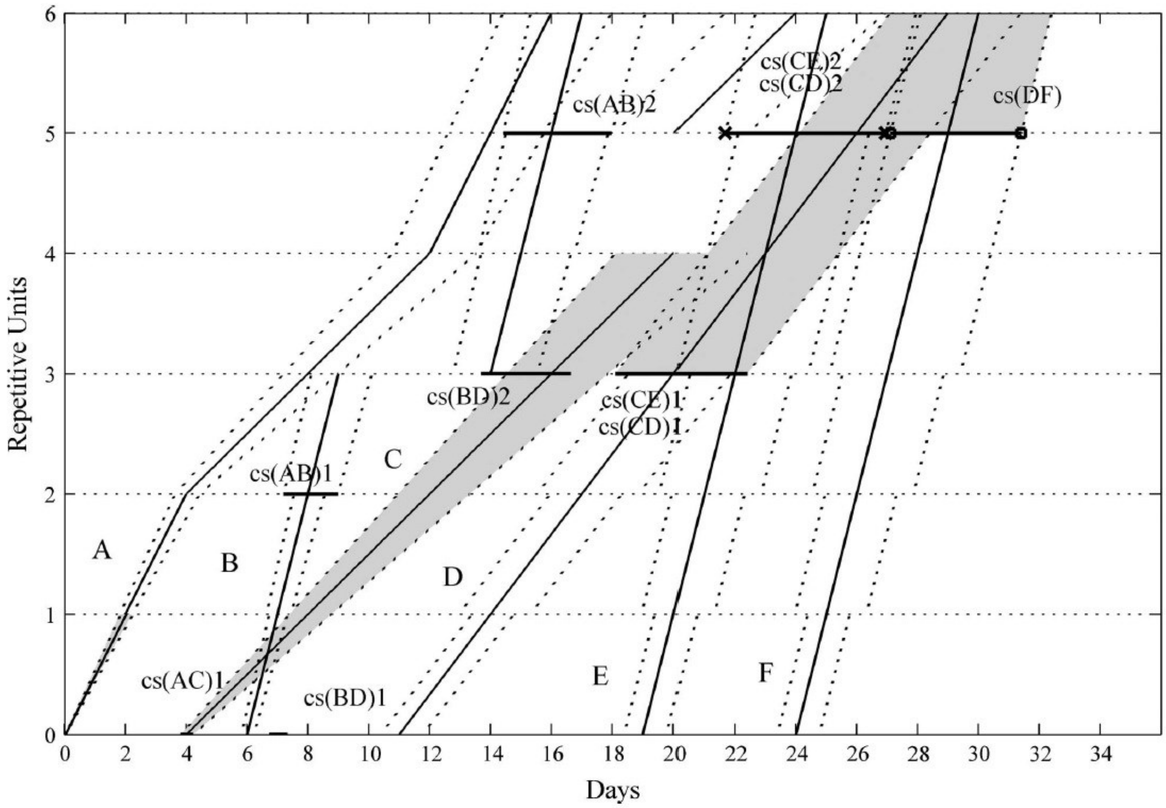


Figure 2.3: F-RSM Schedule (Maravas and Pantouvakis 2011)

F-RSM (Fuzzy Repetitive Scheduling Method) did overcome the need for historical data as it utilizes fuzzy input based on experts evaluation of the project, however, it didn't provide an insight on how this schedule can be used and controlled during construction. Moreover, the schedule display is only valid for simple projects comprising few non overlapping activities, but it is not easy to visualize more complex projects using F-RSM.

2.2.2 **Optimized Scheduling of Repetitive Construction Projects**

Optimizing repetitive construction projects' schedules is a complex process. This is due to the need to maintain resource continuity, which forms an additional constraint on produced schedules. Further difficulty stems from the repetitive nature of these projects which significantly increases the size of the problem. As illustrated by El-Rayes (1997), a repetitive project comprising 20 activities, each having 5 possible crew formations, would result in a 5^{20} possible project schedules. Solving a problem of this size using enumeration methods is clearly infeasible. Examining existing literature revealed many techniques which have been utilized to optimize repetitive construction projects schedules. These techniques include linear programming, dynamic programing, search heuristics (e.g. genetic algorithms) and simulation.

Linear programming has been introduced by Perera (1982 and 1983), where he formulated a set of constraints in an attempt to achieve maximum construction rate and hence least project durations. He considered limits such as resource availability and funding restrictions. His research limitations included giving crew formation in terms of fractions of crews (e.g. 0.556 crew), ignoring costs, not allowing interruptions and assuming all units are typical. In 1990, Reda presented an extension to Perera's (1983) work, where he considered the costs, thus producing a model capable of finding least cost schedule or least duration schedule. Reda's work shared most of the limitations with Perera, as he also didn't solve the crew fractions problem, still assumed typical activities and ignoring interruptions. Linear programming has been utilized later by Ipsilandis (2007) for multi-objective optimization of repetitive projects. His formulation took into consideration cost aspects associated with total project duration, resource idle time, unit completion time (intermediate milestones), slack time and number of units a project is divided into. Ipsilandis model is capable of finding least duration, least cost or least combined impact schedule. He also presented sensitivity analysis for the trade-off between duration and work break costs.

Dynamic programming was first presented through the work of Selinger (1980), where he aimed at reducing project duration while maintaining resource continuity. A more comprehensive dynamic programming based effort was the effort of Russell and Caselton (1988), where they built on the work of Selinger (1980) and added the ability to accommodate typical and non-typical actives and the possibility of having user specified work interruptions.

The capability of optimizing repetitive construction projects while accounting for work interruptions was later well addressed by El-Rayes and Moselhi (2001). In their work, El-Rayes and Moselhi used two algorithms for identifying possible crew formations and presenting a set of feasible interruption vectors for each activity. To avoid generating too many interruption vectors, El-Rayes and Moselhi proposed a set of heuristic rules that reduce the number of feasible interruptions. Later they utilized dynamic programing to identify the optimum schedule for the project at hand in terms of least duration or least cost. Dynamic programming was also coupled with heuristic rules when utilized by Moselhi and Hassanein (2003), where they presented an optimization technique suited for highway projects with several enhancements over previous works. Their technique considered different types of precedence relations between activities with different lag and lead times, in addition their technique considered having crews with multiple availability periods, and added transverse obstructions to the highway root as an additional factor in assigning crews.

Genetic algorithms (GA) were utilized to provide near optimal solutions for repetitive projects. Hegazy and Wassef (2001) utilized GA for cost optimization of non-serial repetitive projects. Their technique searches for the optimum combination of construction methods, crews and interruptions of each activity. Their objective function included direct and indirect costs, bonus and liquidated damages. Their work had a number of limitations such as a limited number of precedence relations for each activity and accommodating only typical activities. A different application of GA is trying to find the optimal duration and interruption days, which was presented by Nassar (2005). Nassar's model's goal was to identify crew formations and interruption vectors that would yield the minimum project duration while keeping the number of interruption days to a minimum. Although this model presented superior results to El-Rayes (1997) work, (in terms of achieving same schedule duration with less interruption days), it had a number of limitations. This model neglected the cost aspect of the project, which is an important aspect when introducing intentional work breaks. Moreover, it evaluated interruption vectors according to the total number of interruption days regardless of the number of interruptions. For example this model would prefer a schedule including ten one day interruptions over an eleven day single interruption, although the later might be a much more practical solution. A further step was taken by El-Rayes and Kandil (2005), where they presented an algorithm for 3 dimensional optimization of construction projects, addressing time, cost and quality. They used GA with 3 different objective functions to produce a set of Pareto optimal solutions for highway projects. Evaluating each solution's time and cost was performed in a straight forward manner, however, they presented a novel approach for quantifying and evaluating the quality performance of each schedule alternative. As shown in Equation 2.1 (El-Rayes and Kandil 2005), the authors provided a methodological approach for evaluating quality of activity when performed by different resources (i.e. crews). Although their algorithm is introduced for highway projects, yet its ability to incorporate repetitive and non-repetitive activities makes it possibly usable for other repetitive projects.

$$Maximize\ Project\ Quality = \sum_{i=1}^{I} wt_i \sum_{k=1}^{K} wt_{i,k} \times Q_{i,k}^{n} \qquad \text{Equation 2.1}$$

Where:

$Q^{n}_{i,k}$ is the value of quality aspect (k) in activity (i) using resource (n);

$wt_{i,k}$ is the weight of quality aspect (k) in comparison to other aspects in activity (i)

wt_i is the weight of activity (i) in comparison to other activities in the project

A not so common utilization of GAs was presented by Georgy (2008), where he integrated GA with CAD to level the resources usage in a repetitive project with a pre-set duration. This optimized resource allocation technique could be used to meet physical resource limits and minimize fluctuations to maintain an even flow of resources through the project's life. GAs were also used by Hegazy and Kamarah (2008) were they presented a cost optimization algorithm that is especially suited for high-rise buildings. The algorithm divides activities into two groups; vertical dimension activities which are activities in the same floor, and horizontal dimension activities which are activities progressing continuously through successive floors (e.g. structural core activities. The calculations are carried out over two stages, initial calculations of a floor's activities are carried out using CPM and the cheapest alternative for executing each activity is identified, and secondary calculations schedule the structural core activities using LOB where all activities progress with the same rate throughout the project. GA is later used for optimization to select between different crews and construction methods and interruption vectors. Other than being applicable to high-rise project only, this algorithm has a number of disadvantages, it doesn't stress on the importance of the continuity of horizontal activities, and doesn't shed light on how interruption vectors are investigated (i.e. whether the user has to suggest interruptions to be investigated or every possible interruption is automatically considered).

Hyari et al. (2009) also utilized GA as an optimization tool for repetitive projects. Their work is distinguished by its bi-objective optimization, where unlike other optimization techniques, they addressed cost and time as objectives for their optimization procedure. Authors incorporated all project costs in a cost module, and through combining it with a scheduling module, they used GA to produce a set of Pareto optimal solutions presenting the trade-off between project time and cost. Their optimization output is favorable in the sense that it offers planners a set of solutions to choose from, however, the produced set of solutions are near optimum, with no guarantee that it includes the optimum solution. Huang and Sun (2009) presented a tool for optimizing repetitive projects also utilizing GAs. Their approach is based on workgroup-based scheduling instead of the traditional unit-based scheduling. Accordingly the schedule they optimize is built of repetitive workgroups, where each workgroup consists of activities using same resource group. Huang and Sun (2009) used the Net Present Value (NPV) of the project, based on a given Minimum Attractive Rate of Return (MARR), to evaluate the fitness of different schedule alternatives. Although, the presumed motivation behind workgroup-based scheduling is simplifying and adding more flexibility to the scheduling and optimization process, yet the activities details within each group is overlooked, which could compromise the efficiency of this tool for actual projects.

A different algorithm was presented by Long and Ohsato (2009), also using GAs as an optimization technique. Long and Ohsato (2009) managed to incorporate a big number of the features desired in an algorithm for optimizing repetitive construction projects. The authors accounted for typical and non-typical activities, activities with and without interruptions, and discrete and continuous relationships between direct cost and duration on the activity level. This algorithm performs single objective optimizations to find near optimal solutions for least cost and least duration schedules. It also performs bi-objective optimizations addressing cost and duration objectives simultaneously, using Equation 2.2 (Long and Ohsato 2009) to calculate the combined impact.

$$TC = \sqrt{\left[W_t.\left(\frac{TP- TP^{Min}}{TP^{Min}}\right)^2 + W_c.\left(\frac{CP- CP^{Min}}{CP^{Min}}\right)^2\right]} \qquad \text{Equation 2.2}$$

Where:

TC	is the combined time and cost impact to be minimized
W_t and W_c	are the relative assigned weights of time and cost factors
TP and CP	are the time and cost of current solution
TP^{Min} and CP^{Min}	are the minimum possible time and cost for project

This equation aims at identifying the solution having the shortest distance to the optimum solutions having minimum cost and duration. This algorithm is very comprehensive in terms of the included feature, only disadvantaged at not guaranteeing optimum solutions and working only in deterministic input. Ezeldin and Soliman (2009) presented a hybrid technique for time-cost optimization of repetitive projects. Their technique used GAs to find local optimum and near optimum solutions, then dynamic programming was used to expand and explore the vicinity of the local optimums and converge on a global optimum.

Another technique used for optimizing repetitive projects is simulation. Hegazy and Kassab (2003) combined flow-chart based simulation and GAs to find the least costly and the most productive resource plan resulting in the best possible benefit/cost ratio for a project. Liu et al. (2005) also combined simulation with GAs, their approach modeled different types of activities and precedence relations and accounted for learning curve effect. Their aim was to identify the optimum crew formations, activity start times and slowdown/interruption rules. Yang and Chang (2005) depended on simulation to convert their stochastic input into deterministic values, which enabled them to use classical linear programming as an optimization technique. Their stochastic input allowed modelling uncertainty associated with resource and funding availability. Srisuwanrat et al. (2008) depended on simulation to model uncertainty in activities durations and hence in resource idle times. Through sequential simulations of activities, their model is capable of optimizing repetitive projects schedules without depending on an external algorithm to maintain resource continuity.

In summary, optimization techniques found in literature and discussed above can be grouped in two main groups. The first group includes deterministic optimization techniques; such as linear and dynamic programming and heuristic search algorithms. These techniques addressed different aspects pertinent to optimizing repetitive projects; however, they all operate under the assumption of deterministic input without considering uncertainties inherent in the project environment. The other group includes non-deterministic optimization techniques; which are simulation based approaches. This group managed to address uncertainty in different input variables, yet they depend on the availability of relevant historical data.

2.3 Using Buffers for Schedule Protection

Critical Chain scheduling is a scheduling technique that aims at producing reliable schedules protected against various uncertainties affecting different projects (Goldratt 1997). This protection is provided through inserting time buffers in strategic places in the schedule; those time buffers are expected to absorb time overruns that happen during project execution. What Goldratt (1997) actually presented was an eye opener that drew attention to the promising potentials of effectively using buffers to address uncertainty. However, he didn't, present a ready to use scheduling technique that would appeal to planners to start using. His initial technique was considered a too simple approach that could easily be refined and built on, and that left many aspects for researches to investigate and enhance (Grey 2007).

Goldratt's (1997) motivation for his research was to try to answer the difficult question of why projects' frequently experience budget and cost overruns. His research concluded that overruns are attributed to two behaviors inherent in human nature, namely student syndrome and Parkinson law. Student syndrome suggests that if a task in a project is assigned a longer duration than it strictly needs, then teams responsible for execution will automatically start late and/or work at a relaxed rate, as they realize they have more time than they need, which is also known as procrastination. This means that when planners add an amount of time to a task's duration under the name of contingency during the planning stage, which almost all planners regularly do, this added amount of time will be wasted by execution teams on site. Moreover, if the need should arise to utilize this contingency duration, it would have already been wasted, thus resulting in an inevitable delay. Figure 2.4 illustrates the effect of student syndrome.

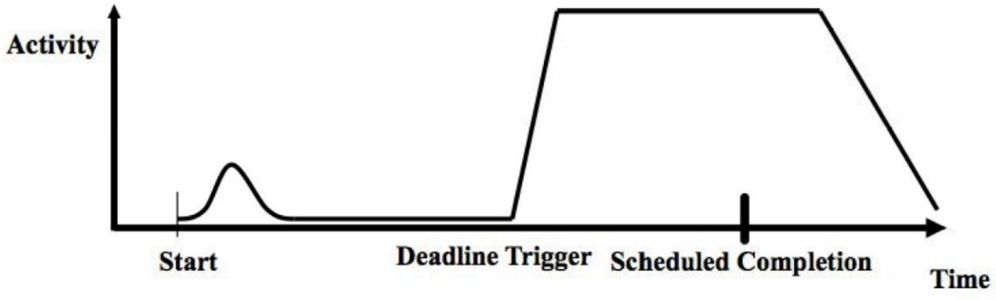


Figure 2.4: Student Syndrome (Robinson and Richards 2010)

Parkinson law (Cyril Northcote 1957) suggests that work will usually stretch to consume all available time. This means that if an execution team starts working on a task on time, and manages to avoid student syndrome, proceeding with a good rate and finds themselves in a good position to deliver the task at hand ahead of time, the team will likely slow down, and will occasionally waste time on unnecessary refinement to reach uncalled for quality levels, and will never report early completion of the task. Also because reporting early completion would likely result in more pressure from middle and upper management when scheduling for the coming projects. Moreover, in construction projects if a task is completed ahead of schedule, successor tasks are almost never ready to begin ahead of schedule.

Critical chain scheduling starts by scheduling activities in a project based on precedence relations, then identifying the critical chain of sequential activities in the schedule. This chain is characterized by having the longest total duration, thus forming the project's total duration. It has been given the name "critical chain" rather than critical path in CPM because it is believed to be more reliable than CPM's critical path; this stems from the fact that critical chain is identified taking into consideration resource dependencies as well as task durations. This means that if another chain joins the critical chain at some point in the schedule and it utilizes the same critical resources, then it is considered part of the critical chain as well (Goldratt 1997). Next step in critical chain scheduling is calculating shortened activities durations. Shortening activities durations is theoretically justified by the following claims (Grey 2007):

1. Each activity is subject to some degree of uncertainty,
2. Planners add contingency durations to each task's calculated duration,
3. It is only logical to assume that not all activities will consume all of their added contingency durations, and
4. If the contingency duration is not needed it will definitely be wasted, since this contingency is locally embedded in each activity's durations.

To quantify how much each task will be shortened, Goldratt (1997) stated that planners assign tasks durations that provide 95% likelihood of completion and that this percentage should be reduced to 50%. The difference between the two percentiles of each activity on the critical chain is pooled at the end of the chain in one project buffer. Figure 2.5 illustrates the basic concept of critical chain.

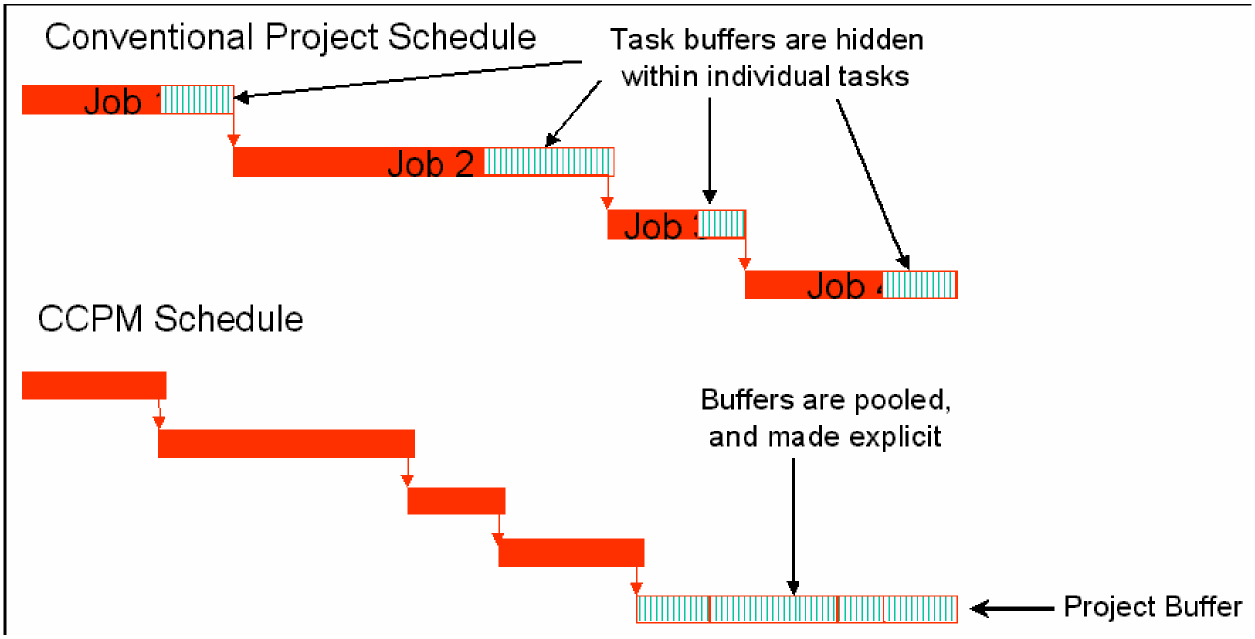


Figure 2.5: Conventional schedule and CC/BM Schedule (Raz, Barnes et al. 2003)

Now buffers are pooled separately from activities durations, but the total project duration is still the same. Shorter project duration is achieved by shortening the project buffer. This is based on the logical argument that it is unlikely that all activities will exceed the 50% likelihood duration; rather only 50% of the activities will exceed that duration. This is similar to the concept of insurance companies; those companies realize their profits by grouping together small payments made regularly by many clients, and then having to compensate only a small percentage of those clients. Critical chain scheduling similarly states that by grouping together contingency portions of each activity, the project buffer can be shortened while still providing adequate protection to overruns that will take place during execution. A valid statistical theory that backs up this approach is the fact that the standard deviation of the sum of a group of independent variables is less than the sum of the standard deviation of each variable. However, the week point in this theory is the assumption that variables (in this case: different activities durations) are statistically independent (Grey 2007).

To complete critical chain scheduling, the final step is to insert two other kinds of buffers, feeding buffers and resource buffers. Feeding buffers are added to all paths feeding into the critical chain to prevent the delays on those paths from affecting the on-time start of critical activities. These paths are then called feeding chains. Resource buffers are described as warning systems or wake up calls or reminders that make sure the resources are ready when it is time to work on a critical activity. Contrary to the project and feeding buffers, resource buffers are not safety times added to project and they do not change the planned time of the project (Goldratt 1997, Li et al. 2007). Figure 2.6 illustrates the use of project and feeding buffers in critical chain schedules.

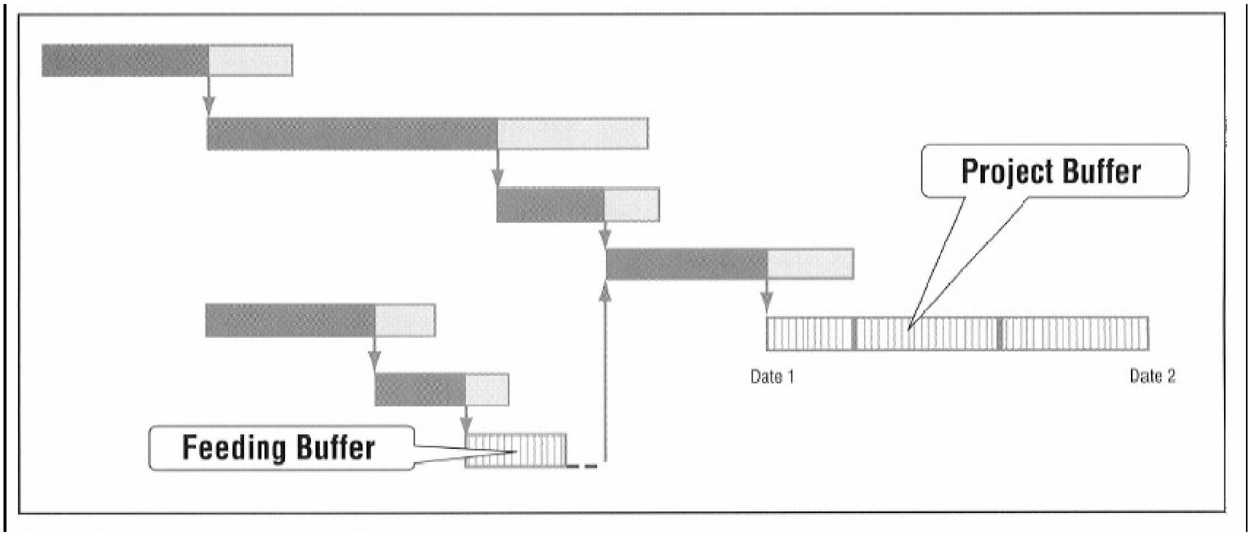


Figure 2.6: Project Network with Feeding Buffer Identified (Raz, Barnes, et al. 2003)

Goldratt (1997) presented critical chain scheduling, later given the name cut and paste method (C&PM), it grabbed the attention of many researches and practitioners. This launched a wave of discussions, criticism and attempted implementations in different fields, naturally including construction industry (Herroelen and Leus 2001 and Herroelen, et al. 2002).

The idea of using different buffers in the construction industry is not new. Time and resource buffers have been used in planning for different purposes, however, not necessarily in the context of critical chain scheduling. For example one of the most common uses of buffers in construction scheduling is in scheduling repetitive construction projects using LOB. Buffers are an essential ingredient in repetitive projects schedules, because resources are already on site, so any delay in an activity has a more costly impact than in traditional projects. Consequently, LOB utilizes buffers to protect continuity, force minimum distance and/or time between consecutive activities to represent technological limitations (like the need to wait for a specific duration for poured concrete to harden) and to avoid crew overlap and site congestions during execution.

The coming sections provide a detailed review of the literature pertinent to existing buffer sizing techniques. To allow better evaluation of those techniques they were classified into two main groups. The first group contains techniques that build buffers based on general representation of uncertainty, without addressing specific sources of uncertainty. These techniques either depend on historical data or fuzzy set theory to assess uncertainty in the schedule at hand. Alternatively, the second group contains techniques that build buffers specifically to provide protection to activities durations while taking into consideration specific sources of uncertainty. Specific sources of uncertainty are further divided into two main groups, a group of factors related to project environment, and a group of factors related to project schedule. Factors related to project schedule don't cause uncertainty by themselves; however, they magnify the impact of uncertainties on project schedule, hence, have an impact on the size of the needed buffer. Figure 2.7 illustrates the hierarchical classification of existing buffer sizing techniques that will be reviewed.

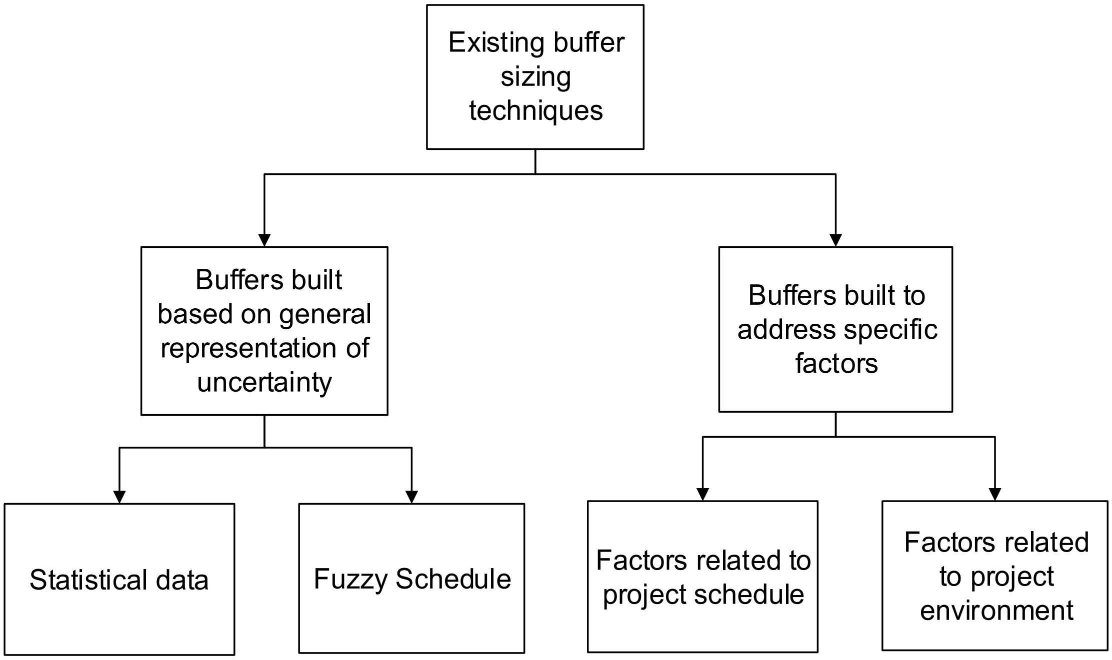


Figure 2.7: Existing Buffer Sizing Techniques

2.3.1 Buffer Sizing Techniques Based on General representation of Uncertainty

The majority of existing buffer sizing techniques fall under this category. Research in this area utilizes schedules that generally account for uncertainty; usually this is done by utilizing historical data, presented in the form of statistical distributions, as input for different scheduling techniques. Also researchers have tried building buffers based on fuzzy schedules. Reviewed literature in this section shows how researches extract uncertainty amount from initial schedules and identify what amount of the normal activities duration is to be assigned to the activity and what amount is to be added to the buffer at the end of the chain.

Hoel and Taylor’s (1999) effort represents one of the very basic approaches based on statistical data. In their research they utilized historical data in the form of statistical distributions representing different activities durations. Through running Monte-Carlo simulation, a distribution of project’s total duration and the corresponding probabilities of completion are plotted. The authors assume the 90% completion probability for a project is a satisfactory probability, and consequently calculate the project buffer as the difference between the duration having 90% completion probability and the mean duration.

Through an application to a simple example, Hoel and Taylor (1999) conducted a comparison between their presented buffer sizing technique and Goldratt’s (1997) original critical chain known as C&PM. They concluded that their technique resulted in relatively shorter total project duration (35 weeks instead of 39 for the studied example).

Their research claims superiority to original critical chain in the way that simulation accounts for critical and near critical chains, whereas critical chain scheduling only based its calculations on a single critical chain. The main criticism of this technique is that it bases buffer length on chain length. Accordingly a 2 year project will have a 6 months project buffer, which is too long. Another limitation worth mentioning here is that this research didn’t address the issue of feeding buffers properly and just relied on the slack (float) at the end of each non-critical chain to protect it from overruns that could extend to affect the critical chain itself and hence compromise the total project duration.

A more realistic approach for buffer sizing that also needs statistical data of activity durations was presented by Shou and Yeo (2000). They identified two limitations of critical chain scheduling that strongly restricts its usability. The first limitation is that it is not reasonable to assume that all activities in a project are subject to the same amount of uncertainty and need the same percentage of their durations as a buffer for protection, and the second limitation is that it is also not reasonable to assume that all planners want to a schedule that provides the same safety represented as the likelihood of completion percentage (identified as 95% by Goldratt).

To beat these limitations the authors did two things. Firstly they divided activities into four categories A, B, C and D, based on the level of uncertainty. This identification of the level of uncertainty in based on a factor called Relative Dispersion (RD) which is similar to the coefficient of variation; RD is equal to the activity’s standard deviation divided by its mean duration. Activities with low uncertainty will have a lower value for standard deviation and will consequently have a lower RD factor, thus would be located in category A and vice versa. Secondly the authors identified three levels of safety for planners to choose from, low safety where projects would have a 68% likelihood of completion, medium safety where projects would have a 95% likelihood of completion, and high safety where project would have a 99.7% likelihood of completion. A planner would use the desired level of safety and the activities' categories to access a table determining the percentage buffer needed for each activity. The figures in this table are based on authors' estimation, see Table 2.1. Although calculating project buffers according to this approach makes more sense than the original C&PM (Goldratt 1997), however, it is a very theoretical approach based on the authors' own perspective of safety and personal estimation of buffers. The research didn't present any validation or case study to compare results.

Another approach that also utilized the characteristics of the statistical distribution of historical data to capture the uncertainty level of activities is Rezaie et al.'s effort (2009). Their methodology also relates the size of buffer to the coefficient of variation as an indication of uncertainty. They identified an optimistic (O) and a pessimistic (P) duration equivalent to the 95% and 5% likelihood of completions respectively. Then they produced a table that defines activities durations according to their coefficient of variation. Table 2.2 illustrates duration's calculations. The research followed a rather simplistic approach when it came to adding the project buffer; it was calculated as 25% of the critical chain duration.

Table 2.1: Buffer Size for Different Activities and Safety Levels (Shou and Yeo 2000)

	Low Safety (68%)	**Medium Safety (95%)**	**High safety (99.7%)**
A	4%	8%	12%
B	12%	24%	36%
C	20%	40%	60%
D	28%	57%	85%

Table 2.2: Buffer Sizing Based on CV (Rezaie et al.'s effort 2009)

Coefficient of Variation (CV)	**Uncertainty Level**	**Activity Duration**
CV < 0.5	Low	(P+O)/2
0.5 < CV < 1	Medium	P/3
CV > 1	High	(P+O)/4

Estimating uncertainty based on statistical distribution of historical data was also used by Fallah et al. (2010) in their research. They identified three parameters to be used as indicators of the amount of uncertainty. The parameters are coefficient of variation as a measure of dispersion in proportion to the mean, skewness as a measure of the lack of symmetry and kurtious as a measure of flatness. An activity affected by uncertainty would have bigger values of these three parameters than an activity less affected by uncertainty. Fallah et al. (2010) assumed all activities have a right skewed lognormal distribution, in such a distribution the mean value always has a greater value than the median value. They calculated this difference and multiplied it by the three parameters identified earlier. Finally authors collected all three terms in an empirical equation to size the project buffer.

A different line of research utilizes fuzzy schedules as a general representation of uncertainty, where the availability of historical data is not required. Few efforts were directed in this direction, first of which was the comprehensive approach presented by Long and Ohsato (2008). Although their research started by trying to solve resource constrained project scheduling problem (RCPSP), they ended it by a buffer building technique. The first part of their research utilized genetic algorithm (GA) to find a deterministic solution for RCPSP. The end result was a schedule having a near optimal make span, where each activity j had the "suitable" duration called t*(j). The second part of the research aimed at inserting a project buffer at the end to provide better protection against uncertainty. To do this, authors replaced deterministic tasks' durations in the schedule with trapezoidal fuzzy numbers. Then for every fuzzy duration authors identify t(h) which is called the high agreement duration. This high agreement duration is calculated based on the number having an agreement index with the fuzzy duration equal to 0.9. Agreement index represents the area of the intersection between two fuzzy numbers in relation to the area of the first these two numbers (Kaufmann and Gupta 1991). The buffer needed for activity j is st(j), it is equal to the difference between t(h) and t*(j). In other words it is the amount of time needed to be added to the suitable duration to increase confidence in achieving this duration to 0.9. Figure 2.8 illustrates the relation between t*(j), t(h) and st(j). After calculating st(j) for all activities, the project buffer at the end is calculated as the square root of the sum of the squares of st(j) of activities on the critical chain.

Although this research doesn't discuss what happens to the resource profile produced in the first part of the research after the introduction of fuzzy durations and project buffers, yet it remains one of the comprehensive approaches that effectively integrated the use of buffers with other scheduling tools.

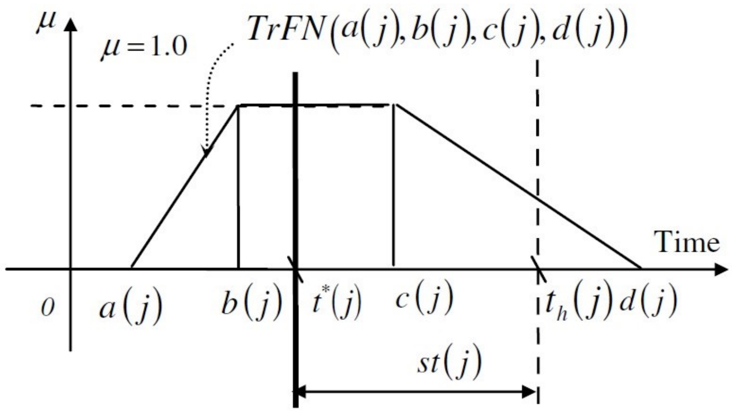


Figure 2.8: Safety Time $st_{(j)}$ of Activity j (Long and Ohsato 2008)

A minor addition to Long and Ohsato's (2008) technique was provided by Min and Rongqiu (2008), as they followed the same methodology but when it came to calculating the project buffer at the end, they multiplied the square root of the some of the squares of safety times of each activity by "ß". "ß" was identified as coefficient varying between 1 and 0 reflecting the risk preference of the project based on cognitive style. Equation 2.3 illustrates the only contribution of Min and Rongqiu (2008).

$$PB = ß\sqrt{\sum St(j)^{2}} \qquad \text{Equation 2.3}$$

Where:

PB	is project buffer
ß	is a cognitive style coefficient ranging from 0 to 1
st(j)	is the difference between high agreement index duration and low agreement index duration

Authors claim that adding the ß value allows better capturing of the risk level of the project based on the planner's intuition. Although this is a simple modification, but it highlights the fact that proper accounting for uncertainty needs to be fine-tuned to correctly convey the conditions surrounding project's execution.

Two different minor improvements to Long and Ohsato's (2008) technique were suggested by Zhao et al. (2008) in their technical note. The first improvement was using a different GA for solving the RCPSP that produced slightly different results, and the second improvement was adding feeding buffers which were neglected by the previous technique. Their approach for inserting feeding buffers depends on shifting activities on feeding chains to the left and inserting feeding buffers behind them, in an approach similar to float time.

Although techniques discussed in this section present a more methodological approach compared to initial C&PM, they have two main limitations. Majority of the discussed techniques try to quantify buffers based on statistical data, which might not be available. Moreover, discussed techniques don't produce schedules specifically accounting for uncertainties pertinent to the project at hand. In other words, using the above mentioned techniques will produce almost the same schedule for similar projects, even if they are to be executed in completely different conditions or if a different level of confidence is required in the final schedule.

2.3.2 Buffer Sizing Techniques Addressing Specific Factors

The second category of buffer sizing techniques found in literature is the category of techniques that quantify buffers to account for specific sources of uncertainty. The factors studied are divided into two main types, factors related to the environment and conditions affecting the project at the execution stage, and factors related to the schedule. Factors related to the schedule don't cause uncertainty themselves, but they are characteristics of the schedule that lead to increasing the impact of delays, hence increasing the size of the needed buffer.

Early buffer sizing techniques addressing specific schedule related factors included Tukel et al.'s (2006) effort, where they introduced two buffer sizing techniques that would take into consideration project characteristics as well as uncertainty. The first technique is called adaptive procedure resource tightness (APRT). This technique addressed resource tightness expressed in the form of resource usage as a ratio to maximum resource availability. This is based on the fact that a project where activities resource consumption is closer to maximum resource availability is more likely to be effected by delays than a project with less resource tightness where activities can be overlapped or accelerated to make up for delays. In APRT buffers are calculated as per Equation 2.4

$$RF_{(q)} = [r_{(i)} \times d_{(i)}] / [T \times Rav_{(q)}]$$

$$r = \max \{RF\}$$

$$\text{Buffer Size} = \sum [(1+r) \times \sqrt{Var_{(i)}}] \qquad \text{Equation 2.4}$$

Where:

$RF_{(q)}$	is resource factor conveying tightness of resource q.
$r_{(i)}$	is resource requirements by activity i
$d_{(i)}$	is duration of activity i
T	is total duration of chain
$Rav_{(q)}$	is total availability of resource q
$Var_{(i)}$	is variance of activity i

The second part of their research presented the adaptive procedure with density (APD), which took into consideration another characteristic of project schedule, which is schedule complexity. Schedule complexity is indicated by the ratio of the total number of precedence relations to the total number of activities. This is based on the argument that in schedules having more precedence relations per activity, when an activity is delayed this delay will affect more successor activities and will have a bigger effect on the project schedule. APD buffers are calculated as per Equation 2.5.

$$K = 1 + (TOTPRE/NUMTASK)$$

$$\text{Buffer Size} = \sum [(1+K) \times \sqrt{Var_{(i)}}] \qquad \text{Equation 2.5}$$

Where:

TOTPRE	is the total number of precedence relation in the network
NUMTASK	is the number of tasks
$Var_{(i)}$	is variance of activity i

Both presented techniques account for different schedule characteristics along with general representation of uncertainty accounted for by including activity variance. The authors didn't justify why they didn't attempt to integrate both factors in a single buffer sizing technique, instead, they compared APRT and ADP to Goldratt's C&PM through application to different size schedules. The comparison revealed the superiority of APRT and ADP, as both techniques produce shorter schedules than Goldratt's that still show more than 90% likelihood of completion.

Developing a buffer sizing technique addressing resource tightness and schedule complexity simultaneously was attempted by Shi and Gong (2009), where they integrated APRT and ADP in one buffer sizing technique. Moreover, they stated that designing a buffer to protect a project schedule is not a procedure that can be completely based on numerical factors, and will always need input from project manager based on his own experience and perception of risk in the project. They presented Equation 2.6 to develop a factor representing project manager's perception of risk.

$$\tau = [D - \mu] / [D` - \mu] \qquad \text{Equation 2.6}$$

Where:

D is task's duration based on required completion probability

μ is the mean value of task duration

$D`$ is task's duration based on company's benchmark of completion probability

Accordingly they developed a buffer sizing equation that integrates all three parameters, resource tightness, schedule complexity and project manager perception of risk (Equation 2.7).

$$\text{Project Buffer} = \sqrt{}\ [\sum (1 + A) \times \text{ß} \times \tau \times \sigma]^2 \qquad \text{Equation 2.7}$$

Where:

A is factor conveying resource tightness

ß is a factor conveying schedule complexity, calculated as per (Tukel et al. 2006)

τ is as calculated in Equation 2.4

σ is the standard deviation of task duration.

Literature shows that time buffers have been also used to solve specific scheduling problems, like stochastic task insertion problem (STI). This problem occurs when certain tasks have known durations, but tasks are not definitely going to occur. In other words the uncertainty is in the probability of the task occurring rather than the task duration. An example of this case is rework. Rework duration can be calculated, but the uncertainty here is if rework is going to be needed or not. Grey (2007) suggested inserting buffers to protect the schedule subject to STI. For that she proposed two different buffer sizing techniques that could be generalized to be used outside the context of STI. In her first approach she argues that buffers should be relevant to activity location in schedule. Delays occurring for activities early in project life can be recovered easier than delays taking place near the end of the project. For that she used the buffer sizing equation represented by Equation 2.8.

Buffer size for activity i = (Task Duration) x (1 - % of Total Project Activity Time Remaining) Equation 2.8

The other technique presented by Grey (2007) related the buffer size to activity's duration, assuming that longer durations make activities subject to longer delays and hence need larger buffers. To represent that, she used Equation 2.9.

Buffer Size = (Duration of Current Task) / (Sum of all tasks durations) Equation 2.9

Farag et al. (2010) included four factors in their buffer sizing technique named fuzzy-logic-buffering-model (FLBM). However, all factors were related to the schedule itself regardless of conditions of execution. To develop FLBM authors depended on feedback from questionnaires to establish which factors to include and what percentage of buffer is needed for each activity based on linguistic evaluation of the chosen factors. The factors included were activity duration, confidence in duration, uncertainty level effect, and influence level. They developed rules to convert factors linguistic values to buffer size. Collecting input for their technique is convenient as people are more comfortable with verbal evaluation of factors. Based on evaluating those rules, authors assigned buffers ranging from 6% to 56%. An example of the developed rules is:

***"IF** duration is very small (VS) **AND** the degree of confidence related to its estimation is very low (VL) **AND** uncertainty level has a medium effect (M) **AND** the activity has a very high influence degree (VH) **THEN** the consequent buffer size should be very large"* (Farag et al. 2010).

Three different factors were included by Zhang and Zuo (2011) in their buffer sizing technique. The factors they included were chain length, human behavior, and external environment. They claimed that longer chains have a higher overrun probability, so they introduced a factor equal to chain length as a ratio to sum of durations of all tasks in schedule. They introduced a factor representing human behavior to account for the bias of estimates provided by different team members. The human behavior factor is calculated as per Equation 2.10. The third and final factor is a factor developed using PEST technique. In this technique a planner evaluates law, economic, social and technological environments by choosing one of the linguistic terms: favorable, good, fair or bad. PEST technique assigns values corresponding to the linguistic evaluation and then gathers them in one factor representing the project environment. A shortcoming of this approach is that it gives the same weight to different factors. Finally Zhang and Zuo (2011) relied on a group of experts to quantify the effect of the three factors on the buffer size using Delphi method.

$$\eta = \sum (t_i - t) / ft \qquad \text{Equation 2.10}$$

Where:

η is the factor representing human behavior

t_i is the duration estimate by team member i

t is the duration estimation of project manager

ft is the number of team members

In summary, Table 2.3 lists and classifies factors included in reviewed buffer sizing techniques. It can be seen that buffer sizing techniques discussed form scattered attempts, each attempt addressing a specific source, or a limited number of sources, of uncertainty, with no comprehensive holistic approach that protects the project from against major sources of uncertainty. Furthermore, a recent effort was performed in an attempt to form a better understanding of how buffers are built by practitioners in the industry. Russell et al. (2013) distributed a nationwide survey among contractors and tradesmen in the industry attempting to identify the main factors taken into consideration while building buffers, and the average size of buffer more commonly added to address each factor. When analyzing the 180 replies they had received, they concluded that planners in the industry do not build buffers separately to address each source of uncertainty, but rather build a single buffer for each activity in view of all relevant sources of uncertainty. Such a conclusion highlights the fact that the approach of building buffers to address specific sources of uncertainty is not the closest approach to what is followed in the industry.

Table 2.3: Summary of Factors Previously Included in Buffer Sizing

No.	**Factor**	**Class**	**Reference**
1	Difference between 90% probability and mean value	Statistical	(Hoel and Taylor’s 1999)
2	Relative Dispersion (standard deviation / mean)	Statistical	(Shou and Yeo 2000)
3	Coefficient of variation	Statistical	(Rezaie et al. 2009 & Fallah et al. 2010)
4	Skewness as a measure of the lack of symmetry	Statistical	(Fallah et al. 2010)
5	Kurtious as a measure of flatness	Statistical	(Fallah et al. 2010)
6	Difference between value at 0.9 agreement index and deterministic duration	Fuzzy	(Long & Ohsato 2010, Min & Rongqiu 2008 and Zhao et al. 2008)
7	Resource usage as a ratio to maximum resource availability	Schedule	(Tukel et al. 2006)
8	Ratio of total number of precedence relations to the total number of activities	Schedule	(Tukel et al. 2006 and Shi & Gong 2009)
9	Project manager own perception of risk	Environment	(Shi & Gong 2009 and Min & Rongqiu 2008)
10	Activity location in schedule	Schedule	(Grey 2007)
11	Activity duration	Schedule	(Grey 2007 and Farag et al. 2010)
12	Confidence in activity duration	Schedule	(Farag et al. 2010)
13	Uncertainty level effect	Schedule	(Farag et al. 2010)
14	Influence level	Schedule	(Farag et al. 2010)
15	Chain length	Schedule	(Zhang and Zuo 2011)
16	Human behavior during estimation	Schedule	(Zhang and Zuo 2011)
17	External environment of law, economy, social and technological	Environment	(Zhang and Zuo 2011)

2.4 Schedule Acceleration

In the construction industry, many parties are involved in complex projects taking place in highly dynamic environments, affected by risks and uncertainties. Throughout projects’ course, different involved parties often need to accelerate the project. Contractors may need to accelerate projects in order to recover from delays experienced during the execution of the work, benefit from contractual bonus, avoid penalties, and/or avoid undesirable weather and site conditions. Owners, on the other hand, may need to accelerate projects to take advantage of market opportunities and/or to meet fiscal and business requirements. Schedule acceleration addresses the issue of finding the delicate balance between the increase in direct cost, due to assigning additional resources and the decrease in indirect cost, due to shorter project duration. Consequently, the main challenge is to identify the acceleration strategy that would require the least amount of additional cost (Moselhi and Alshibani, 2011).

Literature reveals that a wide range of methods were introduced to accelerate projects' schedules at both, the strategic and tactical levels. Techniques utilized to develop such methods can be summarized into five groups: (1) heuristic procedures (Siemens 1971, Moselhi 1993, Hazini et al. 2013); (2) mathematical programming (Henderickson 1989, Pagnoni 1990); (3) simulation (Wan, 1994); (4) genetic algorithms (Hazini et al. 2014); (5) integration of simulation with genetic algorithms (Wei Feng et al. 2000, Ding 2010, Zheng et al 2004), and (6) genetic algorithms and fuzzy set theory (Eshtehardian et al. 2008 a and b).

Although heuristic- based methods were the first to be introduced, their performance is problem-dependent. They are capable of providing good solutions, however, optimum or near optimum solutions are not guaranteed (Wei Feng et al. 2000, and Ammar 2010). Mathematical programming finds better solutions than search heuristics. Its main limitation is the excessive computational effort once the number of options to complete an activity increases or the network becomes too complex. Computer simulation based methods require dedicated simulation professionals (Hajjar and AbouRizk, 2002) and experts' opinions in absence of sufficient numeric data (Chung 2007). Fuzzy set theory (FST) is an alternative approach to the deterministic and probabilistic approaches. FST offers a computationally simpler alternative in comparison to mathematical programming and simulation. FST is also less sensitive to moderate changes in the shapes of input distributions, and does not require the user to make assumptions pertinent to correlations among input parameters (Shaheen et al, 2007).

Those above mentioned techniques address traditional construction projects. Accelerating repetitive construction projects follows a different path as it faces an additional challenge, which is the need to (1) identify critical activities, which is not as simple as in the case of regular, non-repetitive, projects, (2) comply with the crew work continuity constraint. Identifying the activities to be accelerated is made easier in traditional projects due to the existence of the critical path, accelerating any activity on this path would shorten a project's duration (Moselhi 1993). On the contrary, repetitive projects don't have a critical path that can be used for accelerating activities. Two algorithms have been found in literature identifying activities to accelerate in repetitive projects schedules. These algorithms identify activities to accelerate based on the relative alignment of successive activities (Hassanein and Moselhi 2005). Least aligned activities are activities progressing at a lower rate than their successor. Accelerating such activities would result in shortening their duration, advancing their successor's start date, and eventually shortening the project's total duration, as displayed in Figure 2.9. If an activity having a higher rate than its immediate successor is accelerated, project direct cost would increase without any reduction in total project duration.

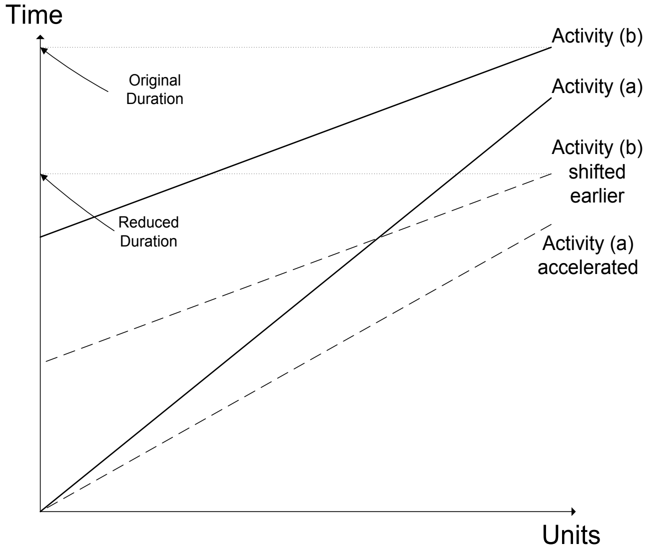


Figure 2.9: Effect of Acceleration on Repetitive Activities (Bakry et al. 2014)

On the other hand, activities having a higher rate than their predecessor and a lower rate than their successor are identified as converging activities. As shown in Figure 2.10, by relaxing a converging activity's rate or introducing an intentional break it can start earlier, and its successor can start earlier (Hassanein and Moselhi 2005). Relaxing an activity might cost less money as it leads to assigning fewer resources, however it might cost more. For example relaxing an activity could mean increased renting period for equipment and increased supervision man hours. Similarly introducing intentional breaks comes at an increased cost, especially in equipment extensive projects like highway projects as rented or procured equipment would be left idle on site.

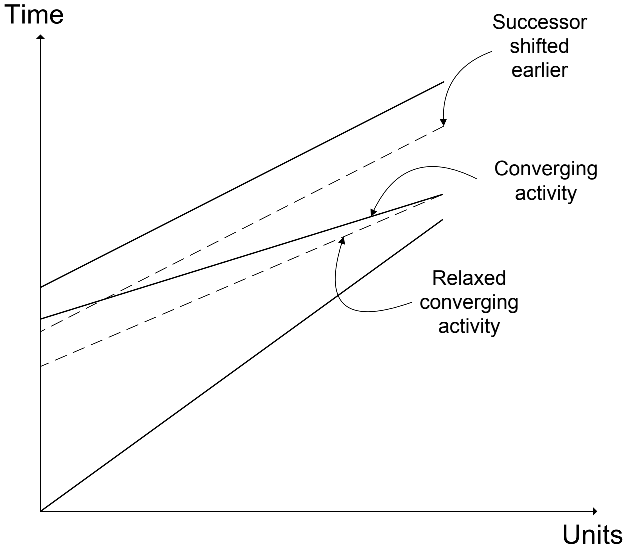


Figure 2.10: Acceleration by Relaxing Converging Activities (Bakry et al. 2014)

The first of the two algorithms identifying activities to accelerate based on relative alignment of successive activities was presented by Hassanein and Moselhi (2005), where they identify less aligned activities using a technique similar to minimum moment algorithm used for resource leveling (Hiyassat 2001). The algorithm calculates the areas trapped between lines representing successive activities, and then calculates the moment these areas cause around an imaginary centerline. Ω is a value revealing activity alignment; it is calculated by subtracting the moment of area of an activity from the moment of area of the predecessor activity. Less aligned activities result in bigger areas with bigger eccentricities, hence resulting in bigger moments, and bigger values of Ω, and vice versa. This algorithm identified an activity to accelerate throughout all project units, thus more suited for typical repetitive projects. When it came to queuing the identified activities for accelerations, the value of Ω was the only used criteria, so this algorithm completely neglected the cost aspect.

The other algorithm identifying activities to accelerate based on relative alignment of successive activities was that of Bakry et al. (2014). Their algorithm utilized the same concept for evaluating alignment through calculating Ω, but introduced two modifications. Though calculating the alignment of successor activities for each unit separately, they insured more efficient assignment of acceleration resources, and through utilizing cost slope as a queuing criteria for activities to be accelerated they are capable of performing least cost optimization. Their research limitations include overlooking other queuing criteria such as those identified by Moselhi and Roofigari-Esfahan (2011).

The main challenge in accelerating project schedule is to reduce project duration with the least amount of extra cost (Moselhi and Alshibani, 2010). Schedule acceleration is a process that requires additional resources, therefore the risk related to this procedure has to be taken into account (Shankar et al, 2011). Through their survey, Moselhi and Roofigari-Esfahan (2011) established other factors commonly taken into consideration by contractors while prioritizing activities for acceleration. These factors include resource availability, risk involved, complexity and logistics, sub-contractor related concerns, number of successors, cash flow constraints, weather and a few other factors. Such risks and influential factors have not been addressed while accelerating repetitive projects.

2.5 Findings of Literature Review

Literature was reviewed in search for methods and techniques that perform the processes of scheduling, optimizing and accelerating repetitive construction projects. Existing scheduling techniques were divided into deterministic and non-deterministic techniques, with more focus given to non-deterministic scheduling techniques. Many optimization methods were located in literature, the detailed review discussed optimization techniques adopting linear and dynamic programming techniques, genetic algorithms, simulation, and in some cases the combined usage of two of those techniques. This chapter also addressed the utilization of time buffers in the construction industry. The basic principles and the advantages of Critical Chain were highlighted, and existing buffer building and insertion techniques were reviewed after being grouped into two main groups; those two groups are buffers built based on general representation of uncertainty, and buffers built addressing specific sources of uncertainty. The reviewed literature also covered the area of accelerating construction projects. The evaluation of acceleration techniques for traditional projects included heuristic procedures, mathematical programming, simulation and genetic algorithms. As for repetitive projects the two existing acceleration algorithms were reviewed. At the end of the literature review, the following facts were established:

1. Although deterministic scheduling is more widely spread and presents easier to follow schedules, it is not as realistic as non-deterministic scheduling and could be misleading.

2. Established scheduling techniques developed for scheduling non-repetitive projects, like PERT for example cannot be used for repetitive projects which necessitate maintaining crew continuity.

3. All existing techniques for scheduling and optimizing repetitive projects while accounting for uncertainty are simulation based techniques relying on historical data, which might not be available and doesn't guarantee providing a schedule particularly representing the project at hand.

4. Critical Chain scheduling is a technique developed based on a sound theory, and provides a tempting approach to modify and apply in scheduling repetitive construction projects.

5. Prolonging activities durations to account for anticipated delays doesn't often prevent delays, due to Students Syndrome and Parkinson Law. Isolating durations added for contingency in separate buffers allows more efficient use of those durations.

6. Existing buffer sizing techniques based only on general representation of uncertainty don't guarantee producing a schedule specifically accounting for uncertainty affecting the project at hand.

7. Existing buffer sizing techniques addressing only specific factors are scattered attempts, with no holistic approach addressing main sources of uncertainty, and does not follow the same approach widely accepted in the industry.

8. Schedule acceleration techniques produced for traditional construction projects are not usable for repetitive projects. While the few techniques produced addressing repetitive projects include limited queuing criteria and neglect risks associated with the schedule acceleration process.

Chapter 3: Methodology

3.1 Introduction

After existing literature has been carefully reviewed and its gaps have been identified, this methodology was designed to address the recognized research objectives. The presented methodology comprises 3 integrated models, each performing a unique task. The first model is the Optimized Scheduling and Buffering Model, which is to be utilized during the pre-construction phase of the project. This first model has two main objectives, firstly it aims at modelling uncertainty in different input parameters while carrying out an optimization procedure that would generate the least cost or the least duration schedule. This step of the first model produces an optimized fuzzy schedule, having fuzzy durations and costs for different activities. Secondly it defuzzifies the optimized schedule and sizes and inserts time buffers between successive activities. These time buffers capture the user's desired confidence level in the final schedule and utilize it to provide protection to the schedule against anticipated delays.

The second and the third models are utilized after the construction phase has begun. The Monitoring and Dynamic Rescheduling Model captures the progress of the project on site to date and uses it for two main tasks; to update the baseline schedule using exact dates and durations of completed activities, and to improve the estimate of the remaining part of the project. This model also encompasses a buffer tracking tool, which measures actual buffer consumption and compares it to the planned consumption. Through a specially developed index, buffer consumption is used to track project progress to date. The third and final model is the Acceleration Model, it is to be used to address the frequently faced challenge of selecting between different options to speed up the delivery of the project, while addressing the issue of balancing the added cost with the corresponding reduction in duration.

Collectively, these three models comprise the needed tools to generate an optimum project schedule and monitor and control project progress. This section explains in detail the presented methodology through the three coming sections, each addressing one of the three models. Figure 3.1 illustrates schematically how the three models are organized. The scheduling technique used to represent the produced schedule is LSM, this is due to the main reasons described earlier in literature, LSM has the ability to schedule projects comprising repetitive and non-repetitive activities, typical and non-typical activities, and sequential and non-sequential activities.

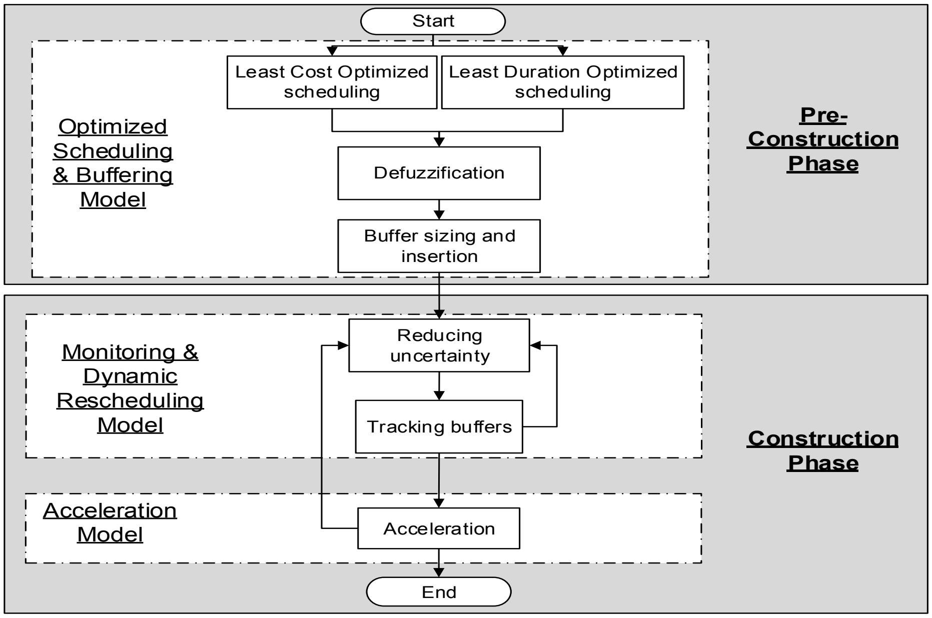


Figure 3.1: Proposed Methodology Main Components

3.2 Optimized Scheduling and Buffering Model

This is the first model to be run in the sequence of the presented methodology. This model is responsible for producing the base line schedule. After the gaps in existing literature have been identified, and to properly address the shortages highlighted in the findings of the literature review, the goal of this model has been generally identified as follows: to present an algorithm for the optimized scheduling of repetitive projects under uncertainty, that can be fine-tuned based on user's preferences to provide different levels of protection against delays. Based on this goal, features required to be inherent in the produced schedules have been identified as listed below:

- Meets basic scheduling needs, of calculating tasks' durations, determining activities start and end times, and calculating total project's duration and cost.
- Maintains resource work continuity, which is essential for repetitive projects.
- Has the ability to find the optimum crew formation that yields the least duration or least cost schedules.
- Has the ability to isolate durations added for contingency in separate time buffers, which can be customized to meet user defined level of protection for the scheduled activities against various anticipated delays.
- Provides a clear visual representation of activities dates and rates and different buffers.

- Has the ability to accommodate repetitive and non-repetitive activities within the same project.
- Has the ability to optimize typical and non-typical repetitive activities within the same project.

The model is divided into two main integrated components: the scheduling component, which is responsible for the optimized scheduling process and schedule consistency, and the defuzzification and buffering component, which is responsible for sizing and inserting time buffers. An overview of the developed scheduling model is displayed in Figure 3.2. This model starts by acquiring basic project data (summarized in Table 3.1), which is classified into three groups: project general data, activities data, and crew related data.

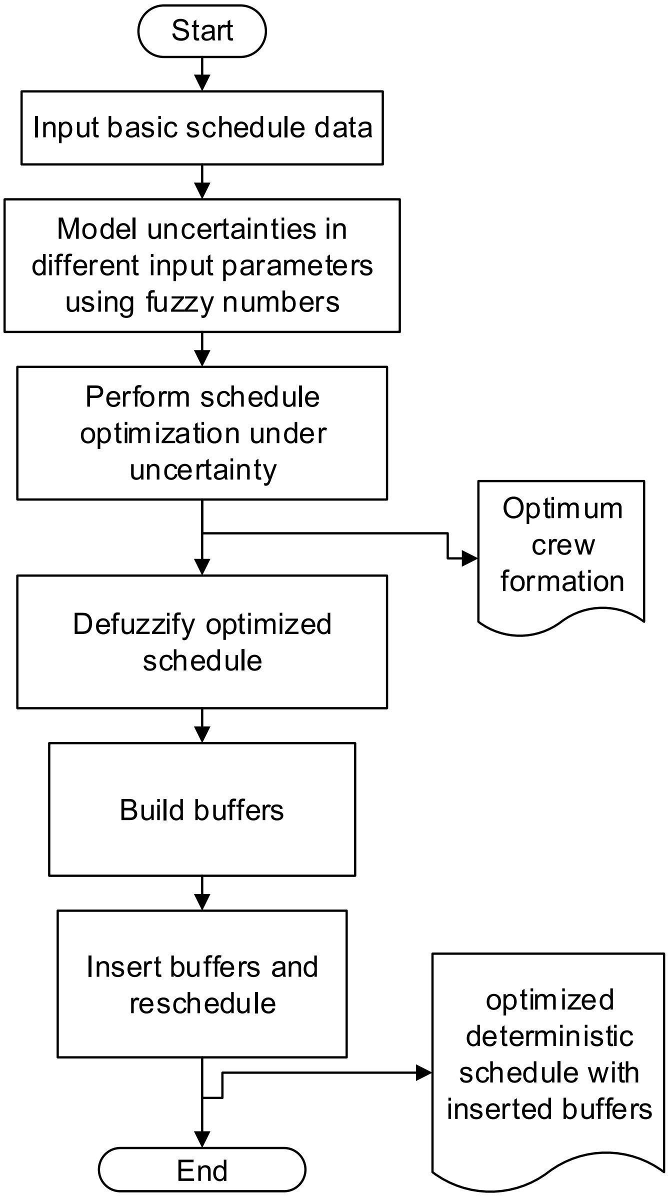


Figure 3.2: Optimized Scheduling and Buffering Model

Table 3.1: Initial Input for Scheduling Module

General Project Data	Project start date
	Indirect Cost
	Number of repetitive units
Optimization Data	Optimization objective (total cost or total duration)
	Desired level of confidence in generated schedule
Activities Data	Names
	Sequence
	Logic (precedence relations)
	Quantities
Resources	Names
	Related activities
	Productivity rates
	Availability times
	Direct Costs

3.2.1 Modelling Uncertainty

To overcome the limitations of existing techniques, the developed algorithm utilizes fuzzy set theory to model uncertainty associated with different input parameters at the activity level (Bakry et al. 2013). In this model, uncertainties are considered for all input parameters, accept job logic. Fuzzy set theory is used to model crew productivity rates, quantities of work, direct costs and project indirect cost. By doing so, the user is allowed to represent each input parameter by a range of values rather than a single deterministic number in a more comprehensive approach. Triangular fuzzy numbers are used to model uncertainties in the context of this research. Unlike simulation models, fuzzy modelling of uncertainty does not depend on availability of relevant historical data. The use of triangular membership is not only simpler and requires less computations in comparison to other functions, but also facilitates the process of input data for users in the construction industry. Triangular fuzzy numbers consist of three numbers a, b and c, having membership value of 0, 1 and 0 respectively. Triangular fuzzy numbers may be symmetrical or anti-symmetrical. Figure 3.3 illustrates an example of a generic triangular fuzzy number used to model a duration, where the three values on the X-axis represents the possible duration range, and the Y-axis represents their respective membership values. In this illustration “a” represents the shortest possible duration, “b” represents the most possible duration and “c” represents the longest possible duration. In case the user is certain about an input variable and wishes to express it deterministically, he can use three equal values for “a”, “b” and “c”, in this case the fuzzy number converges to a deterministic number as illustrated in Figure 3.4.

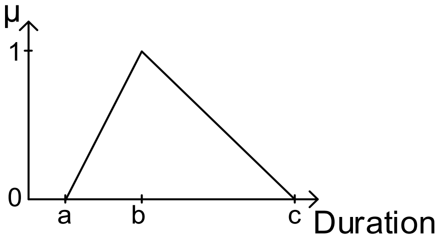


Figure 3.3: General Triangular Fuzzy Number

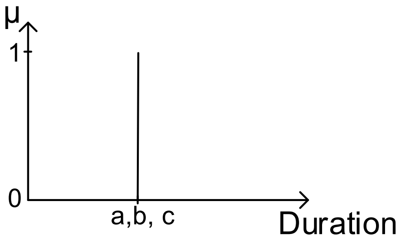


Figure 3.4: Using 3 Values to Represent a Deterministic Number

As the input is modelled in fuzzy numbers, the processing for the optimized scheduling operations and the output are comprised of triangular fuzzy numbers as will be explained in details later on. Although accounting for uncertainties in the proposed algorithm using fuzzy set theory is based on users' judgment and relies significantly on the experience and foresight of the user, yet it is more structured and realistic than deterministic scheduling that tends to overlook uncertainties or account for them in an oversimplified approach.

3.2.2 **Fuzzy Dynamic Programming**

The Optimized Scheduling and Buffering Model's first task is to optimize the schedule while accounting for uncertainty. The purpose of this optimization procedure is to find the optimum crew formation that would yield, based on user's preference, either the least cost or the least duration schedule for the project at hand. The optimization procedure is performed using a modified dynamic programming optimization technique that is adapted to compute fuzzy numbers. Dynamic programming is particularly suited for the problem at hand for two reasons; (1) crew selection for successive activities is dependent and is performed at sequential stages, and (2) the elimination policy employed by the technique allows for reducing the number of alternatives in the solution space which allows handling bigger and more complex schedules. This procedure is identified as a single objective, single state, N stage problem. It is a single state because there is only one selection made at each stage, which is the optimum crew to be used. The N stages represent the number of steps forming the project, where each step is a group of activities having the same predecessors and performed simultaneously.

The proposed modified dynamic programming algorithm works typically through a forward path stage and a backward path stage. The model's forward path calculations start with optimizing a small part of the schedule and then gradually expands until the whole schedule is optimized. In details, the model first considers the first two activities in the project, and lists all possible crew formations for these two activities. For each possible crew formation for these first two activities, the model uses the activities' quantities and crews' productivities to calculate each activities' duration for each unit, Equation 3.1 shows how fuzzy quantities are divided by fuzzy productivity rates to produce fuzzy durations. Using the projects start date and the calculated durations, Equation 3.2 shows how the model calculates activities initial starting dates. Adding the duration of each unit to its initial starting date gives the initial finish date for that unit (Equation 3.3). These initial start and finish dates are calculated respecting precedence relations and crew availability. To address crew work continuity activity dates are shifted to avoid overlapping with predecessor activities which leads to work interruptions. This shift is calculated separately for each activity using Equation 3.4. The shift for an activity is calculated as the maximum intersection between any two successive units in this activity. Adding the shift calculated for each activity to the initial start and end dates calculated earlier results in the final start and end dates for each unit in an activity (Equation 3.4 and Equation 3.6). After these basic calculations are performed for all possible crew formations for the first two activities, based on user's preferences, the model utilizes one of the two objective functions, (Equation 3.7 and Equation 3.8) to evaluate different crew formations. In case of least cost optimization, Equation 3.7 calculates the total cost of the first two activities for each of the possible crew formations. For each crew of the second activity the local optimum predecessor crew is the one that would generate the least total cost. While in case of shortest duration optimization, Equation 3.8 calculates the nearest finish date. For each crew of the second activity the local optimum predecessor crew is the one that would generate the least value for the selected objective function. In both cases non-optimal predecessors are ultimately excluded. For example, if there are two crews available for performing the first activity in a project (A1 and A2), and three crews available for the second activity (B1, B2 and B3), six different crew combinations can be listed to execute these two activities (A1B1, A2B1, A1B2, A2B2, A1B3 and A2B3). The above calculations will determine the optimum local predecessor for each of the three crews of the second activity and non-optimal predecessors will be excluded. Consequently, the number of crew combinations considered for the next activity will be three instead of six (for example A1B1, A1B2 and A2B3). This exclusion enables proceeding only with local optimum predecessor crews, which reduces the problem size and offers savings in computational time and effort as the size of the problem expands later on.

$$d^{i}_{[a,b,c]} = [(Q^{i}_{[a]} / P_{[c]}), (Q^{i}_{[b]} / P_{[b]}), (Q^{i}_{[c]} / P_{[a]})] \qquad \text{Equation 3.1}$$

$$IS^{i,j}_{[a,b,c]} = \text{Max}\ (IF^{i-1,j}_{[a,b,c]}, IF^{i,j-i}_{[a,b,c]}) \qquad \text{Equation 3.2}$$

$$IF^{i,j}_{[a,b,c]} = [(IS^{i,j}_{[a]} + d^{i}_{[a]}), (IS^{i,j}_{[b]} + d^{i}_{[b]}), (IS^{i,j}_{[c]} + d^{i}_{[c]})] \qquad \text{Equation 3.3}$$

$$\text{Shift}^{\,j}_{[a,b,c]} = \text{Max}_{i=1}^{i=n}\ [(IS^{i,j}_{[a]} - IF^{i-1,j}_{[a]}), (IS^{i,j}_{[b]} - IF^{i-1,j}_{[b]}), (IS^{i,j}_{[c]} - IF^{i-1,j}_{[c]})] \quad \text{Equation 3.4}$$

$$S^{i,j}_{[a,b,c]} = [(IS^{i,j}_{[a]} + \text{Shift}^{\,j}_{[a]}), (IS^{i,j}_{[b]} + \text{Shift}^{\,j}_{[b]}), (IS^{i,j}_{[c]} + \text{Shift}^{\,j}_{[c]})] \qquad \text{Equation 3.5}$$

$$F^{i,j}_{[a,b,c]} = [(IF^{i,j}_{[a]} + \text{Shift}^{\,j}_{[a]}), (IF^{i,j}_{[b]} + \text{Shift}^{\,j}_{[b]}), (IF^{i,j}_{[c]} + \text{Shift}^{\,j}_{[c]})] \qquad \text{Equation 3.6}$$

Where:

$d^{i}_{[a,b,c]}$	is the fuzzy duration of unit i
$Q^{i}_{[a\ a,b,c]}$	is the fuzzy quantity of work of unit i
$P_{[a,b,c]}$	is the fuzzy productivity of crew
$IS^{i,j}_{[a,b,c]}$	is the fuzzy initial start date of unit i in activity j
$IF^{i-1,j}_{[a,b,c]}$	is the fuzzy initial finish date of unit i in activity j
$S^{i,j}_{[a,b,c]}$	is the fuzzy start date of unit i in activity j
$F^{i-1,j}_{[a,b,c]}$	is the fuzzy finish date of unit i in activity j
$\text{Shift}^{\,j}_{[a,b,c]}$	is the needed fuzzy shift to ensure crew work continuity

The next step is to consider the next activity, where each of the available crews available for the third activity is put in a crew combination with the second crew activities and their pre-identified optimum predecessors. Similarly for all possible crew combinations for the third, second and optimum first activity crews are listed. Equation 3.1 to Equation 3.6 are used to calculate durations and dates. Again the relevant objective function (Equation 3.7 for least total cost and Equation 3.8 for least duration) is retrieved and utilized to identify the local optimum predecessor for each of the third activity crews. This gradual expansion and local optimum predecessor identification continues until all the activities have been scheduled and all local optimum predecessors have been identified.

The backward path stage identifies the optimum crew formation for the last activity and then scans all optimum predecessor activities' crews to identify the global optimum crew formation. The end result of the dynamic programming is the crew formation that would yield the least cost or the least duration schedule, the corresponding schedule with fuzzy durations and dates, and the fuzzy cost. Figure 3.5 illustrates how the fuzzy dynamic programming works.

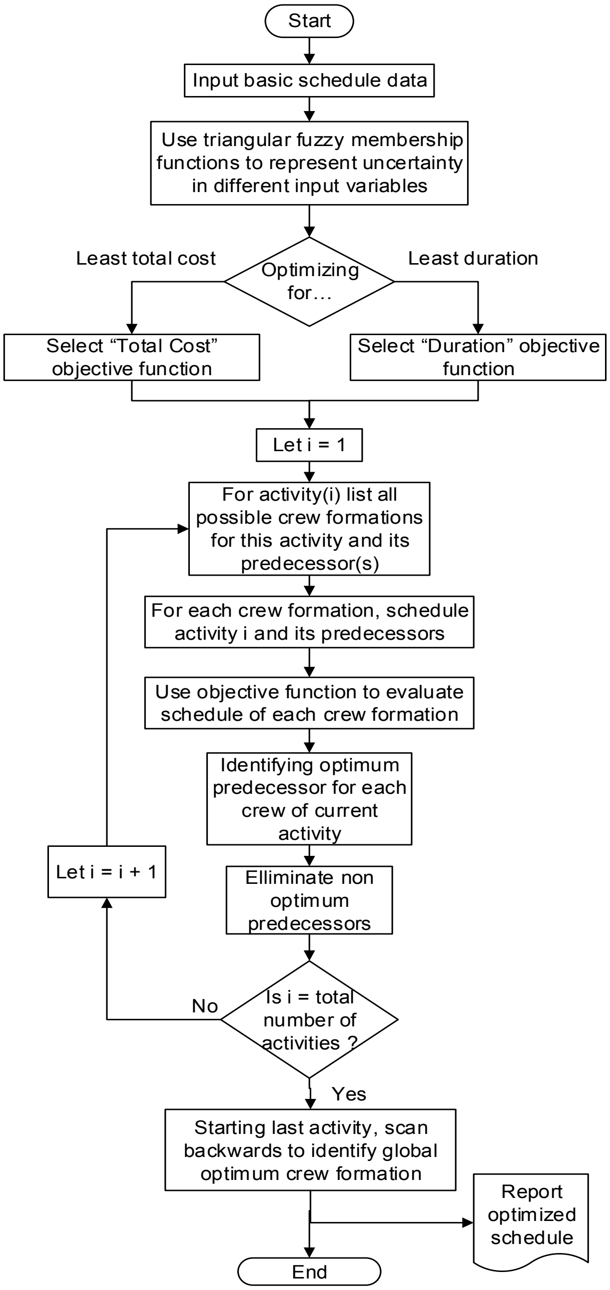


Figure 3.5: Fuzzy Dynamic Programming Flowchart

$$\mathrm{TC}_{[a,b,c]} = \left[\sum_{i=0}^{m}\sum_{j=0}^{n}\left(\mathrm{M}_{[a,b,c]}\times \mathrm{Q}^{i,j}{}_{[a,b,c]} + (\mathrm{L}^{i,j}{}_{[a,b,c]} + \mathrm{E}^{i,j}{}_{[a,b,c]})\times \mathrm{d}_{[a,b,c]}\right)\right] + \mathrm{IC}_{[a,b,c]}\times \mathrm{D}_{[a,b,c]} \qquad \text{(Equation 3.7)}$$

$$\mathrm{D} = \mathrm{F}^{m,n}{}_{[a,b,c]} \qquad \text{(Equation 3.8)}$$

Where

$\mathrm{TC}_{[a,b,c]}$	is the fuzzy total project cost
i	is the number of unit
j	is the number of activity
m	is the total number of units
n	is the total number of activities
$\mathrm{M}_{[a,b,c]}$	is the fuzzy material cost per quantity unit $\mathrm{TotalCost} = \left[\sum_{i=0}^{m}\sum_{j=0}^{n}(\mathrm{Material}\times \mathrm{Q} + (\mathrm{Labour} + \mathrm{Equip.})\times \mathrm{D})\right]\mathrm{Duration} = \mathrm{MaxF}_{j}^{i}$
$\mathrm{Q}^{i,j}{}_{[a,b,c]}$	is the fuzzy quantity per unit
$\mathrm{L}^{i,j}{}_{[a,b,c]}$	is the fuzzy labour cost per unit time
$\mathrm{E}^{i,j}{}_{[a,b,c]}$	is the fuzzy equipment cost per day
$\mathrm{d}_{[a,b,c]}$	is the fuzzy duration of unit "i" for activity "j"
$\mathrm{D}_{[a,b,c]}$	is the fuzzy total project duration
$\mathrm{F}^{m,n}{}_{[a,b,c]}$	is the fuzzy finish time of activity "m" at unit "n"
$\mathrm{IC}_{[a,b,c]}$	is the fuzzy project indirect cost per unit time

3.2.3 Schedule Defuzzification and Buffering

The optimization model utilizes fuzzy arithmetic operations to calculate activity durations, direct, indirect and total project costs, evaluate alternatives using the applicable objective function, and to generate fuzzy schedules. After the optimum crew formation is identified and the corresponding optimum fuzzy schedule is generated, schedule durations are defuzzified. Defuzzifying means generating a deterministic value to represent a fuzzy number. This deterministic value is often called the Expected Value (EV) of the fuzzy number. This step is necessary to convert the produced fuzzy schedule to a deterministic one. A deterministic schedule is easier to visualize and manage, also a deterministic schedule is often needed when addressing milestones for different contractual purposes. This algorithm utilizes the center of area (COA) method for defuzzification. The COA method is used because it generates an expected value (EV) that matches the probabilistic mean of a normalized fuzzy number. The EV of a general trapezoidal fuzzy number can be expressed by Equation 3.9 (Shaheen et al. 2007).

$$EV = a + \frac{2(c-b)(b-a) + (b-c)^2 + (b-a)(d-a) + (d-a)^2}{3(c-b+d-a)} \qquad \text{Equation 3.9}$$

In the context of this research triangular fuzzy number are utilized, which is equivalent to a trapezoidal fuzzy number where it has the same value for (b) and (c). Accordingly, for a triangular fuzzy number (a,b,c) Equation 3.9 can be simplified and presented as follows:

$$\mathrm{EV} = \frac{a+b+c}{3} \qquad \text{Equation 3.10}$$

Applying Equation 3.10 to the duration or cost of each activity gives the expected deterministic value of that duration or cost. The next section explains how this EV is used to generate the final schedule.

3.2.4 **Buffer Sizing and Insertion**

3.2.4.1 Buffer Sizing

Different projects are run in different environments and under different circumstances, they are subject to different uncertainties and demand a different level of confidence in the produced schedule. The previous sections showed how uncertainty is modelled in different input parameters and how the schedule optimization is performed under uncertainty. This section sheds the light on how protection is provided to the generated schedule against delays, and how can this protection be customized to satisfy different users' requirements. In the context of this research time buffers are built and inserted into the schedule to protect against delays affecting construction projects. Buffers here are defined as isolated activities, having a duration and requiring no resources, inserted to raise the confidence in meeting the project deadline and intermediate milestones. Buffers are built taking into consideration uncertainties affecting each activity and the user desired confidence in activity and project completion. This approach is based on the recent findings of Russell et al. (2013), where they concluded, after a thorough nation-wide survey, that in industry a single buffer is built for each activity in view of all possible risks, rather than building a buffer for each corresponding source of risk.

The buffer building calculations utilize the fuzzy input by the user to assess the amount of uncertainty affecting an activity. Agreement Index (AI) is used for such a purpose; AI is an index reflecting the possibility of two fuzzy events (Kaufmann and Gupta 1991). It represents the percent of the first event inside the second event (Long and Ohsato 2007), it is calculated as the ratio between the intersection area of the two shapes representing the two events (A) and (B) and the area of the shape representing the first event (A). Equation 3.11 (Kaufmann and Gupta 1991) shows how AI is calculated for two fuzzy events (A) and (B). AI value ranges between 0.0 and 1.0. If two fuzzy events have an AI equal to 1, this means the complete conformance of the two fuzzy events, while an AI equal to 0 represents the lack of any conformance between the two events. AI has been utilized in fuzzy scheduling due to its ability to consider the shapes and areas of fuzzy events (Okada and Gen 1994, Lorterapong and Moselhi 1996, and Long and Ohsato 2007).

$$AI(A,B) = Area(A \cap B) / Area(A) \qquad \text{Equation 3.11}$$

In the case of a fuzzy event and a deterministic one, AI can be calculated using the area of the fuzzy event membership shape and the area before or after the deterministic number. By doing so, the calculated AI represents the conformance of the fuzzy number and the range of values equal to and greater than or equal to and less than the deterministic number. Figure 3.6 shows the case of the overlap of the area of a fuzzy event (a,b,c) and the area before a deterministic number (t). As the value of t increases, it moves towards the value c of the fuzzy number, and results in a bigger value for AI. The value of AI reaches 1.0 when t is equal to c. For the general case of a trapezoidal fuzzy number and a deterministic number, Equation 3.12 shows how AI is calculated.

In the context of this research, AI is used to capture the user's desired confidence in the duration of the generated schedule. This approach was established after realizing that fuzzy durations represent a range of possible durations for an activity. Realistically, different managers in different projects would be inclined to utilize different values out of the range of values included in a fuzzy number. For example a more risk-averse manager would use the longest duration of a fuzzy number, thus covering the worst case scenario, while a risk-prone manager might be willing to utilize a value closer the middle of the possible durations range.

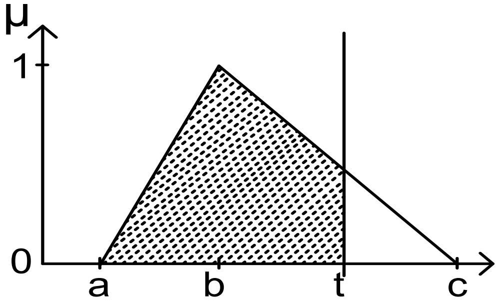


Figure 3.6: Intersection Area of a Fuzzy Number and a Deterministic Number

$$\mathrm{AI} = \frac{\frac{(c-b+d-a)}{2} - \frac{(d-t)^2}{2(d-c)}}{\frac{(c-b+d-a)}{2}}$$ Equation 3.12

Where:

AI is the agreement index

a, b, c and d are the four values representing the fuzzy event

t is the deterministic event

The model utilizes a modified form of Equation 3.12, where the input is the fuzzy event (representing the duration) and the user expresses the desired AI, the generated output of the modified equation is the deterministic duration (t). Instead of the original trapezoidal fuzzy numbers, the equation is modified to be applicable to the triangular fuzzy numbers utilized in this research as displayed in Equation 3.13. For example, if a user decides to input an AI equal to 0.85, this equation would return a deterministic duration for each activity that has a possibility of 0.85 being met during actual construction. Accordingly the user is accepting a 0.15 possibility of finishing behind schedule for that activity.

$$\mathrm{Dur}_{\mathrm{AI}} = \mathrm{c} - \sqrt{(1-\mathrm{AI})(\mathrm{c}-\mathrm{b})(\mathrm{c}-\mathrm{a})} \qquad \text{Equation 3.13}$$

Where:

$\mathrm{Dur}_{\mathrm{AI}}$	is the required deterministic duration
a, b and c	are the three value representing triangular fuzzy duration
AI	is the agreement index specified by user

After different uncertainties have been modelled originally in different input parameters, and in the durations and costs of the generated optimized fuzzy schedule, the model breaks down activities durations into a deterministic duration placed in the schedule representing the start and end of an activity, and an isolated deterministic buffer inserted in the schedule as a contingency duration. The duration used to represent the activity is the EV calculated using the COA method through Equation 3.10. According to the buffer definition adopted in this research, buffers are numerically identified as the difference between the deterministic duration meeting the user specified AI, and the EV of the fuzzy durations. Figure 3.7 illustrates how buffers are sized based on the values of durations matching the required AI and the calculated EV. The figure shows how specifying a higher value for AI requires inserting a bigger time buffer and vice versa. Equation 3.14 shows how the buffer is calculated.

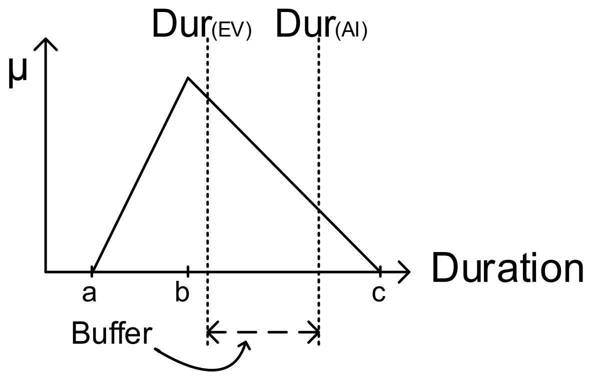


Figure 3.7: Buffer Sizing

$$\text{Buffer}^{i} = \left(c - \sqrt{(1-AI)(c-b)(c-a)}\right) - \left(\frac{a+b+c}{3}\right) \qquad \text{Equation 3.14}$$

Where:

Buffer^{i}	is the buffer calculated for unit i
a, b, and c	are the three values forming the triangular fuzzy duration of the unit
AI	is the agreement index specified by user

Once the optimization is complete, and the optimum crew formations is identified, fuzzy schedule is generated and EV of activities durations and costs are calculated, the user can experiment with different values for AI to choose what best suits the project at hand. The user can fine tune the schedule through monitoring the change in AI and the corresponding change in size of buffers and project total duration. Moreover, as different activities have different characteristics and are performed under different circumstances, the user can utilize different values of AI for different activities. For example for activities that have a critical successor, or associated with critical milestones the user might be inclined towards utilizing a value for AI closer to 1.0. While less values for AI might be used for less critical or less resource intensive activities.

These calculated buffers are sized to address uncertainties at the activity level. This is because the buffer sizing approach so far captures uncertainties modelled in input for different activities. However, other unaccounted for uncertainties could still affect the project, these are uncertainties at the strategic level. Strategic level uncertainties deliver their impact through impacting the company or the project, without being tied to a specific activity, examples of such uncertainties are financial, political, cultural, and market uncertainties (Zayed et al. 2008). To allow for accounting for such uncertainties, the user is given the opportunity to add an extra amount to the size of the buffer. The user can add to the very last buffer of the project, which provides protection the final project delivery date as will be explained in the next section, or to any buffer preceding an intermediate activity.

3.2.4.2 Buffer Insertion

Similar to activity durations, each unit's buffer is calculated separately, then buffers are aggregated into a single buffer after each activity, thus, providing protection to that activity's immediate successor and the rest of the project against anticipated delays of the previous activity. In general, buffers are inserted at the least duration between any two successive activities. This point is the base for positioning the successor activity, and hence it will be affected the most in case the predecessor activity is delayed. The least duration between successor activities is identified according to the shift calculated earlier to maintain resource continuity (Equation 3.4). The unit at which the biggest shift was needed for a successor activity to maintain its crew's work continuity is the unit at which the least duration between successor activities exists. However, this second shift will be calculated in a deterministic form (Equation 3.19) as it resembles the additional time extension needed to meet the deterministic duration matching user' required confidence level. To allow for buffer insertion, successor activities must be shifted again. In total two shifts are needed for each activity, the first shift to maintain resource continuity and the second shift to make room for inserting intermediate buffers. The buffer insertion and activity re-positioning can take one of four forms, depending on the relative productivity rates of each activity and its direct successor. Each of these four forms is explained and illustrated through the following 4 cases.

The first case represents the situation where the predecessor activity (A) has a lower productivity rate than its successor (B). In this case, the successor activity is usually positioned utilizing the end date of the last unit of the predecessor. In this case the buffers calculated separately for each of the units of activity (A) are aggregated and inserted after the end date of the last unit; accordingly, the successor is scheduled backwards. This case is illustrated in Figure 3.8.

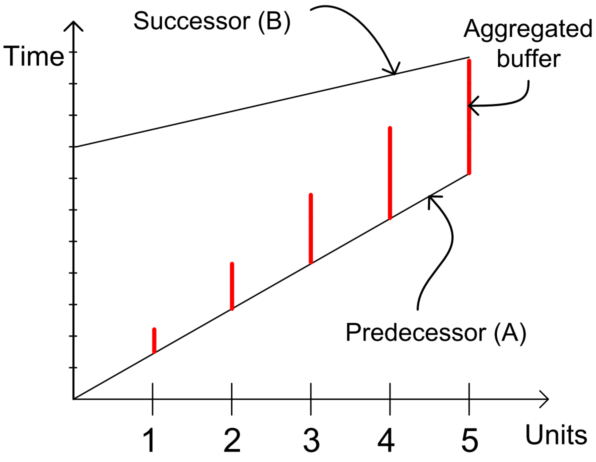


Figure 3.8: Buffer Insertion - Case 1

The second case represents the situation where the predecessor activity (A) has a higher productivity rate than its successor (B). In this context, having a lower rate refers to the worst possible rate, based on calculating the duration plus any possible delays (worst case delay). Accordingly, the successor isn't affected by the predecessor, as it starts after the predecessor and proceeds at a slower rate. As illustrated in Figure 3.9, the successor is positioned after the first unit of the predecessor.

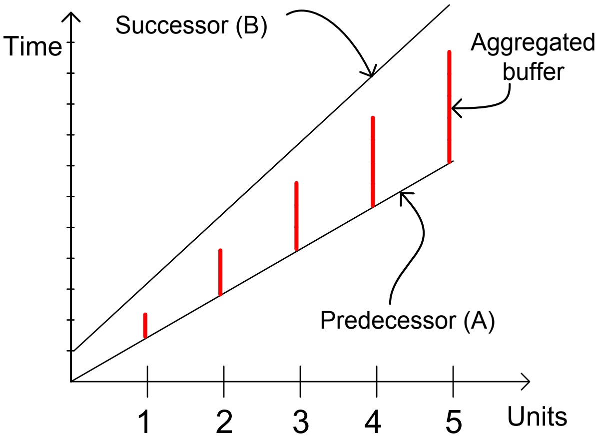


Figure 3.9: Buffer Insertion - Case 2

The third case addresses the scenario where one or both of the two activities is a non-typical activity. Non-typical activities are activities having different quantities for different units and/or are performed by crews with varying productivities, and are accordingly represented by a broken line with varying slope. In this case the least distance between the two activities is to be located, and this is where the cumulative buffer of the successor is inserted. Figure 3.10 shows an example of this case, where activity B has a higher productivity for the first 3 units and then a lower productivity for the remaining two units. The least distance between activities A and B is after unit 3, so the buffer is inserted in this place, and its value is equal to the summation of the buffers of the 3 first units.

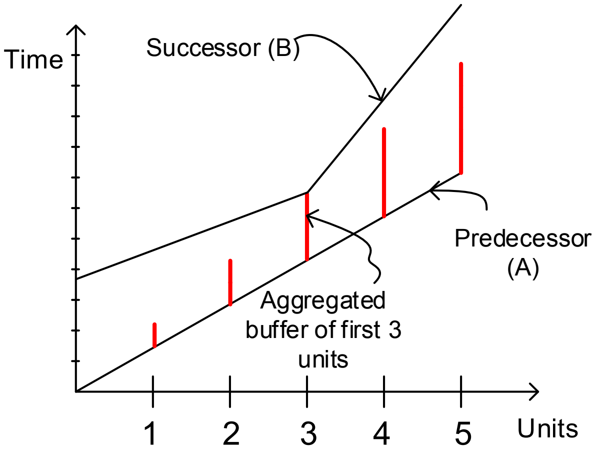


Figure 3.10: Buffer Insertion - Case 3

The fourth and final case is the special case of the last activity in the schedule. In this case, activity buffer is placed after the end date of the last unit. This buffer provides protection for the project end date against possible delays affecting the last activity. This buffer is illustrated in Figure 3.11.

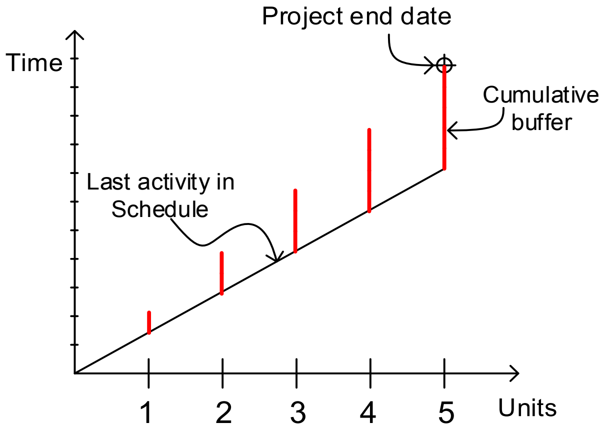


Figure 3.11: Buffer Insertion - Case 4

These described cases illustrate how buffers are inserted to provide protection against delays. The cumulative buffers between successive activities provide protection for each activity against the maximum delay of its predecessor, and the buffer after the last activity extends this protection to protect the project final delivery date. After buffers have been inserted the optimized schedule is re-generated once again, however, this time 2 main changes are introduced:

1. The deterministic value EV is used for activities durations.
2. A second shift is formulated to delay activities dates to accommodate inserted buffers, including buffer extension to account for strategic level buffers.

A set of equations similar to the set used in section 3.2.2 is utilized for the regeneration of schedule. Equation 3.15 shows how the duration are recalculated deterministically, according to (COA) method. Equation 3.16 and Equation 3.17 show the calculations of initial start and finish dates for each unit, respecting job logic. The first shift calculated in

Equation 3.18 maintains resource continuity, it represents the deterministic form of Equation 3.4. The value U_{ld} is calculated for each activity except the first one, used to identify the location of the least duration between an activity and its predecessor. U_{ld} is the unit at which the maximum first shift was calculated. The second shift in Equation 3.19 is the activity buffer, it is the summation of included unit's buffers and an additional component input by user accounting for delays at the strategic level. Equation 3.21 and Equation 3.22 calculate the final start and finish dates after accounting for resource continuity and intermediate buffers. Finally, total project cost is recalculated according to deterministic durations, new total project duration and EV of direct cost components and project indirect cost as shown in Equation 3.22. Figure 3.12 illustrates the steps of the buffer sizing and insertion process. This step concludes the first model. The generated schedule is the project base-line schedule. It is the result of modelling uncertainties in different input parameters, optimizing the schedule under uncertainty, defuzzifying the schedule, and sizing and inserting buffers.

$$d^{i,j}{}_{[EV]} = \frac{a+b+c}{3} \qquad \text{Equation 3.15}$$

$$IS^{i,j} = \text{Max}\ (IF^{i-1,j}, IF^{i,j-i}) \qquad \text{Equation 3.16}$$

$$IF^{i,j} = IS^{i,j} + d^{i,j} \qquad \text{Equation 3.17}$$

$$\text{Shift\_I}^{\,j} = \text{Maxi}_{i=1}^{i=n}(IS^{i,j} - IF^{i-1,j}) \qquad \text{Equation 3.18}$$

$$\text{Shift\_II}^{j} = \sum_{i=1}^{i=U}\left[\left(c - \sqrt{(1-AI)(c-b)(c-a)}\right) - \left(\frac{a+b+c}{3}\right)\right] + \text{BST} \qquad \text{Equation 3.19}$$

$$S^{i,j} = IS^{i,j} + \text{Shift\_I}^{j} + \text{Shift\_II}^{j} \qquad \text{Equation 3.20}$$

$$F^{i,j} = IF^{i,j} + \text{Shift\_I}^{j} + \text{Shift\_II}^{j} \qquad \text{Equation 3.21}$$

$$TC = \left[\sum_{i=0}^{m}\sum_{j=0}^{n}\left(M_{[EV]} \times Q^{i,j}{}_{[EV]} + \left(L^{i,j}{}_{[EV]} + E^{i,j}{}_{[EV]}\right) \times d^{i,j}{}_{[EV]}\right)\right] + [IC_{[EV]} \times F^{m,n}] \qquad \text{Equation 3.22}$$

Where:

$d^{i,j}_{[EV]}$	is the EV of duration of unit i in activity j
a, b and c	are the three values forming fuzzy duration (Equation 3.1)
$IS^{i,j}$	is the initial start date of unit i in activity j
$IF^{i,j}$	is the initial finish date of unit i in activity j
n	is the total number of units
$Shift\_I^{j}$	is the needed shift to ensure crew work continuity
$Shift\_II^{j}$	is the needed shift to accommodate inserted buffers
U_{ld}	is the sequence of the unit at which maximum shift-I (Equation 3.18) was located
AI	is the user specified Agreement index
BST	the buffer component account for strategic level delays
$S^{i,j}$	is the start date of unit i in activity j
$F^{i,j}$	is the finish date of unit i in activity j
TC	is the total project cost
m	is the total number of units
$M^{i,j}_{[EV]}$	is the EV of material cost
$Q^{i,j}_{[EV]}$	is the EV of quantity of work of unit i in activity j
$L^{i,j}_{[EV]}$	is the EV of labour cost per unit time
$E^{i,j}_{[EV]}$	is the EV of equipment cost per unit time
$IC_{[EV]}$	is the EV of the indirect cost per unit time
$F^{m,n}$	is the finish date of the last unit of the last activity

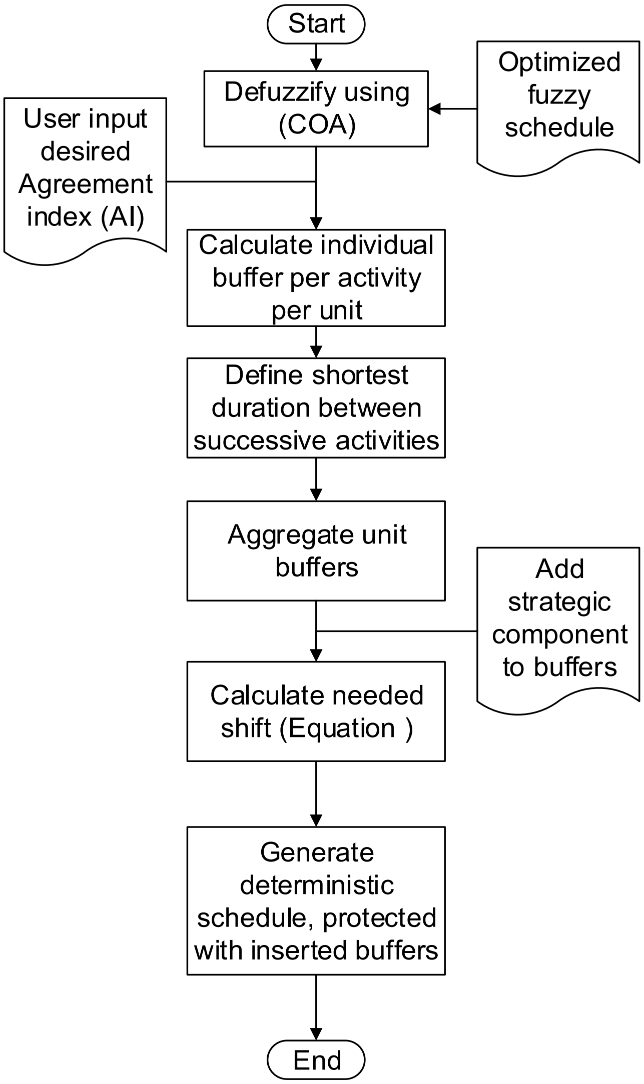


Figure 3.12: Buffer Sizing and Insertion Flowchart

3.3 Monitoring and Dynamic Rescheduling Model

Planning construction projects is a process that depends on many estimates and assumptions. Once construction starts, a deviation is often experienced between the plan and how the project actually progresses. This makes the process of monitoring the project and accordingly updating the baseline schedule a basic component of any project management process. This is the second Model in the presented methodology. It functions during the execution phase of a project. The general aim of this model is to extend the abilities of typical monitoring processes to enhance the chances of successful project delivery. Such extension in capabilities is directed in two main directions, to capitalize on the repetitive nature of the project to reduce uncertainty in the remaining part of the schedule, and to use buffer consumption rate as an early warning if project needs corrective measures. The main responsibilities of this model are identified through the following points:

1- Track actual progress on site.
2- Evaluate the performance of the project.
3- Re-generate a more accurate schedule for the remaining part of the project.

The input for this model is the base-line schedule generated before the construction phase, and the site reports capturing the actual progress on site. This model monitors the two main components of the schedule, these are activities and buffers. Monitoring activities progress is through capturing exact dates of completed activities or parts of activities, and using the exact durations of partially completed activities to improve the estimate for their remaining parts. While monitoring buffers is through observing buffer consumption and using it to evaluate the schedule performance of the project. Figure 3.13 illustrates the main components of the schedule and the associated tracking steps. Those steps are detailed in the coming subsections.

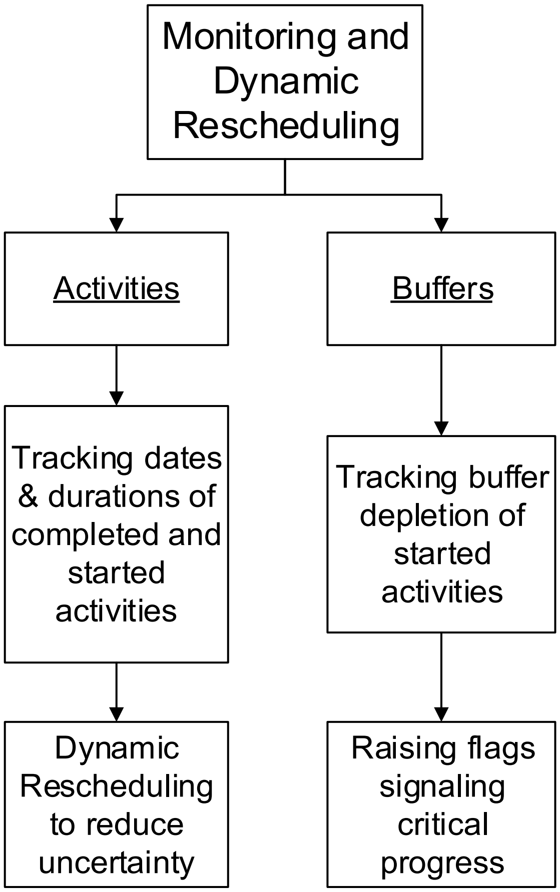


Figure 3.13: Main Components of Monitoring Model

3.3.1 **Updating Activities**

In this model updating activities durations is the first task. This can be done at any point in time after project execution commences and data from site is available. Data from site is typically periodic data (usually daily reports) documenting progress on site. At the update point, each activity in a repetitive project is in one of three states, completed, in progress or not started yet. For activities that are already completed the user manually replaces the start and end dates of each unit in the base-line schedule with dates capturing actual progress on site. This is the basic form of manual update practiced in any project.

For activities in progress the user manually enters the start and end dates for the completed units and the start dates for units in progress. For the units that had not started of ongoing activities and for activities that had not started yet, there are no actual dates to enter, however, the user is offered the choice if he needs to revise the initial input (work quantities or crews productivities). Now that the user is in the middle of the project, he is likely to be able to provide more accurate quantities or productivity rates for repetitive activities, in comparison to the figures he estimated before the project begins. Based on any change in quantities or rates of the remaining activities, the schedule durations will change and the schedule dates will be recalculated accordingly.

3.3.2 Reducing Uncertainty

Reducing uncertainty is the second task in this model, and is one of the main components of the monitoring process. During the preconstruction phase the methodology aimed at providing users with enough flexibility to allow modelling uncertainty in all input parameters. After construction had commenced, uncertainties start unravelling and users are more certain about different parameters they had estimated during pre-construction. This task aims at learning from the completed part of the project and using this knowledge to provide a more accurate plan for the remaining part of the project. By doing so, the remaining part of the schedule is based more on how different resources are actually performing on site, rather than on educated guesses during the planning stage. This is done for ongoing activities, where a number of the activities' units have been completed, yet more units are not executed yet. Using weighted average, actual durations of completed units and fuzzy unit durations used for the base-line schedule are merged to fine tune the estimate for the coming units. Weights are assigned based on the number of the completed units to the total number of units. Using weighted average to combine each of the three values forming the triangular fuzzy number (original duration) and the crisp number (actual duration of completed units) results in a less fuzzy number. Equation 3.23 shows how the weighted average is used. This equation shows that as the number of completed units increase, the weight of the crisp number increases, and the resulting duration for the remaining units converges more towards a deterministic number with less uncertainty.

$$d_{[a,b,c]} = \left[\left(\frac{a\times U_r + t \times U_c}{U_t}\right), \left(\frac{b\times U_r + t \times U_c}{U_t}\right), \left(\frac{c\times U_r + t \times U_c}{U_t}\right)\right] \quad \text{Equation 3.23}$$

Where:

$d_{[a,b,c]}$	is the new fine-tuned fuzzy duration
a,b and c	are the three parameters of the initial fuzzy duration
U_r	is the number of remaining units in an ongoing activity
U_c	is the number of competed units in an ongoing activity
U_t	is the total number of units in an ongoing activity

Numerical Example:

This hypothetical numerical example is presented to further demonstrate how weighted average is used to reduce uncertainty. A repetitive activity is formed of 40 units, each having the same fuzzy duration of (20,22,25) days. Initial scheduling generates an EV of 22.3 days, and a user input AI of 0.9 results in a 1.5 days buffer. After execution commences and 20 out of 40 units of that activity have been completed, an average actual duration of 23 days is recorded. To get a more accurate number for the duration of the 20 remaining units, Equation 3.23 is used to combine the original fuzzy estimate and the actual duration from site. Using weighted average, fuzzy duration for the remaining 20 units will be (21.5, 22.5, 24). Thus having an EV equal to 22.7 days and using the same AI of 0.9 the buffer needed will be 0.9 day.

The new EV is 22.7 days, instead of 22.3 days for the original estimate, which is closer to the actual duration on site (23 days). Accordingly as the estimate was more accurate, the size of the needed buffer decreased from 1.5 to 0.9 days. Figure 3.14 shows the original fuzzy duration, EV and buffer, while Figure 3.15 shows the updated fuzzy duration, EV and buffer. This result shows how the three values forming the fuzzy number converged towards the actual duration from site, resulting in a less fuzzy estimate for the remaining units' durations. The increased certainty was revealed by the closer to reality EV and the corresponding shorter buffer. By doing so, the repetitive nature of the project is a source of knowledge, as it generates data that is used to give a more accurate schedule for the remaining units of the project. This numerical example reduced uncertainty in the activity's duration, it can be similarly used to reduce uncertainty in crews' productivity, work quantities and costs.

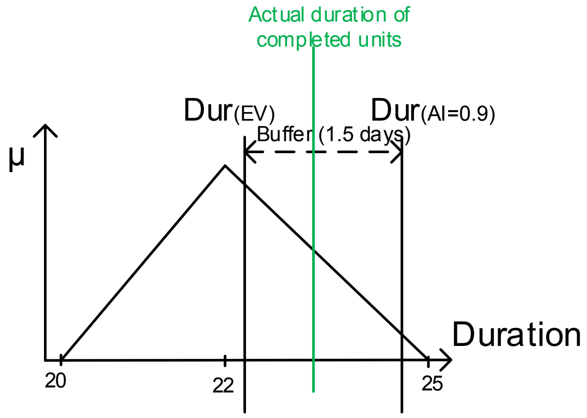


Figure 3.14: Original Fuzzy Duration, EV and Buffer

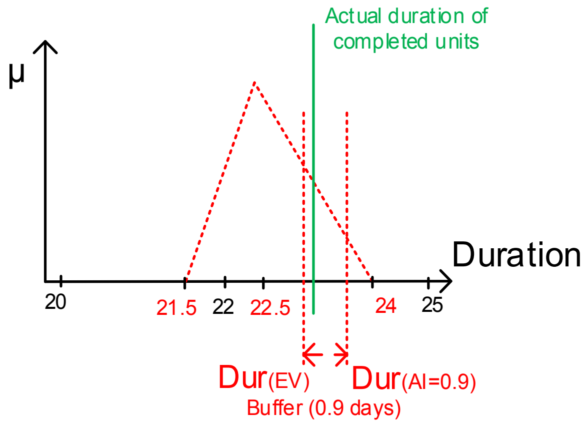


Figure 3.15: Duration with Reduced Fuzziness, EV and Buffer

3.3.3 Tracking Buffers

In the presented methodology, activities durations are calculated strictly as needed, without including any prolongation to serve as contingency. The contingency is being added as isolated units named buffers. Accordingly, activities are likely to extend beyond the listed durations and consume part of or the complete buffer. This is accounted for, and will not impact successor activities or final project delivery date. However, the schedule will start being affected if an activity is delayed beyond the size of its directly succeeding buffer. This deems the process of monitoring buffer consumption an essential part of monitoring the project performance.

Tracking the inserted buffers is the third and last task of the monitoring and dynamic scheduling Model. This part tracks the consumption of the inserted time buffers through two main steps explained through the two following subsections. The first step addresses the quantification of buffer consumption as project progresses, and the second step evaluates project performance based on buffer consumption rates.

3.3.3.1 Monitoring Buffer Consumption

It is a given fact that during project execution buffers will be consumed, either partially or completely. During construction, it is misleading to consider that the activity is protected against delays just because a part of the succeeding buffer is not yet consumed. Buffer consumption has to be distributed along the duration of the activity. To be able to evaluate the activities progress and the adequacy of the remaining buffer to provide protection, close monitoring of the consumption of the inserted time buffers is needed, as well as comparison to the planned buffer consumption.

As the presented methodology supports non-typical activities having different durations for different units, buffers cannot be depleted equally among activity's units. Therefore this model establishes a buffer consumption plan based on the total activity duration. After a buffer is sized and inserted for an activity during the preconstruction phase, the buffer is redistributed over activity days for monitoring consumption during the construction phase. Buffers are distributed equally among activity days only till the point of buffer insertion, identified by unit U in Equation 3.18.

At the point of update, after updating the base-line schedule with the exact dates of complete units, buffer consumption is quantified and the consumption is compared to the planned consumption till this point in time. Buffers are consumed by delay in activity finish times, accordingly, buffer consumption is calculated as the difference between the last completed unit's actual end date and its planned end date. This difference is the magnitude of the delay in the activity's progress so far, which is delay that the buffer should mitigate. Equation 3.24 shows how the buffer consumption is calculated.

$$BC^{j} = F^{ij}_{A} - F^{ij}_{P} \qquad \text{Equation 3.24}$$

Where:

BC^{j} is the buffer consumption for activity j

F^{ij}_{A} is the actual finish date of unit i in activity j

F^{ij}_{P} is the planned finish date of unit i in activity j

The calculated buffer consumption can have one of three states, positive, zero and negative. Having a positive value for buffer consumption is more likely to occur, where the project is progressing behind schedule. This means there is a delay in finishing dates of units, which results in buffer consumption. The resulting buffer consumption has to be compared to the planned rate to evaluate if the project delay so far is within or exceeds the originally anticipated delay. A zero buffer consumption means the project is advancing exactly as planned. While a negative buffer consumption occurs in the rare case that the project is progressing ahead of schedule. To evaluate the rate of the buffer consumption, it is compared to the planned consumption rate. At the point of update, planned buffer consumption is the percentage of the total activity buffer that is proportional to the planned duration of the units completed to date, as shown in Equation 3.25.

$$\mathrm{PBC}^{j} = \mathrm{B}^{j} \times \frac{\sum_{i=1}^{u} d^{i,j}}{\sum_{i=1}^{m} d^{i,j}} \qquad \text{Equation 3.25}$$

Where:

PBC^{j}	is the planned buffer consumption of activity j
B^{j}	is the inserted buffer for the activity j (Equation 3.14)
$d^{i,j}$	is the duration of unit i in activity j
u	is the number of completed units
m	is the total number of unit in activity j

3.3.3.2 Monitoring Activity and Project Progress

As discussed in earlier sections, buffers are inserted for contingency, hence are likely expected to be consumed. Closely monitoring actual buffer consumption and comparing it to the planned consumption rate is used to evaluate the performance of the project to date. This model uses a guide for monitoring buffer consumption rate, which is the Buffer Consumption Index (BCI). BCI, as calculated through Equation 3.26, is the ratio of the actual buffer consumption (Equation 3.24) to the planned consumption (Equation 3.25). BCI is principally an indication to how much of the buffer was consumed versus how much was supposed to be consumed.

$$\mathrm{BCI}^{j} = \frac{\mathrm{BC}^{j}}{\mathrm{PBC}^{j}} \qquad \text{Equation 3.26}$$

Where:

BCI^j is the buffer consumption index for activity j

BC^j is the actual buffer consumption for activity j

PBC^j is the planned buffer consumption for activity j

The value of BCI calculated at any point of update, indicates the schedule performance. The smaller the value of BCI the better the project is progressing. If the buffer consumption is less than the planned, this represents the favorable status where the project is ahead of schedule, and BCI will have a value less than 1.0. In case the buffer consumption is exactly equal to 1.0, this means the complete match between the exact delays anticipated for each unit during the pre-construction phase actually occurred on site, thus exactly consuming the pre-inserted buffers. If the project starts running behind schedule, this will result in a buffer consumption higher than the planned, resulting in BCI values more than 1.0.

So far, BCI is calculated for each activity, and consequently used to monitor project progress at the activity level. This is more useful in repetitive projects, as they usually comprise less activities, or are represented by activity groups each representing a bundle of sequential activities. However, there is definitely an undeniable need to monitor progress at the project level. The difficulty of such task stems from not having a critical path as the case is in CPM projects. The critical path is the sequence of activities having the longest cumulative duration, thus determining the project total duration. These activities are critically monitored because, unlike non-critical activities, any change in critical activities' durations directly impacts the total project duration.

To determine the critical path of activities for repetitive projects, Harmelink and Rowings (1998) presented the Controlling Activity Path (CAP). Their method adopted a forward and backward pass calculations to determine the sequence of activities or parts of activities that determine total project duration. The identified CAP for a repetitive project consists of critical activity segments and critical links between activities, shifting the control from one activity to its successor. Those critical links are either least duration or least distance between any two neighboring activities.

In the context of the presented method, time buffers are inserted at the least duration between successive activities, hence acting as the controlling links in CAP. Accordingly, monitoring buffers serves the same purpose as monitoring the controlling and critical paths. This is the equivalent of quantifying accumulating delays on the critical path of a project, and hence delays in the project as a total. To allow for monitoring the schedule performance at the project level, BCI is calculated for the summation of the individual activity buffers, and similarly compared to the planned consumption. Any delay in any critical activity segment will start consuming the successive buffer, and will be reflected on the consumption of the summation of buffers at the project level. As the case in monitoring individual activities, favorable project progress is reflected in a less than 1 value for BCI, and vice versa. Tracking buffer consumption at the project level alone is not sufficient, as the aggregation masks the progress of individual activities. If an activity is progressing ahead of schedule and another activity is behind schedule, summing up buffer consumption for both can result in a misleading BCI equal to or greater than 1. Therefore monitoring should be performed at both levels, activity and project, throughout the project. Figure 3.16 abstractly illustrates the flowchart of the schedule monitoring and dynamic rescheduling process.

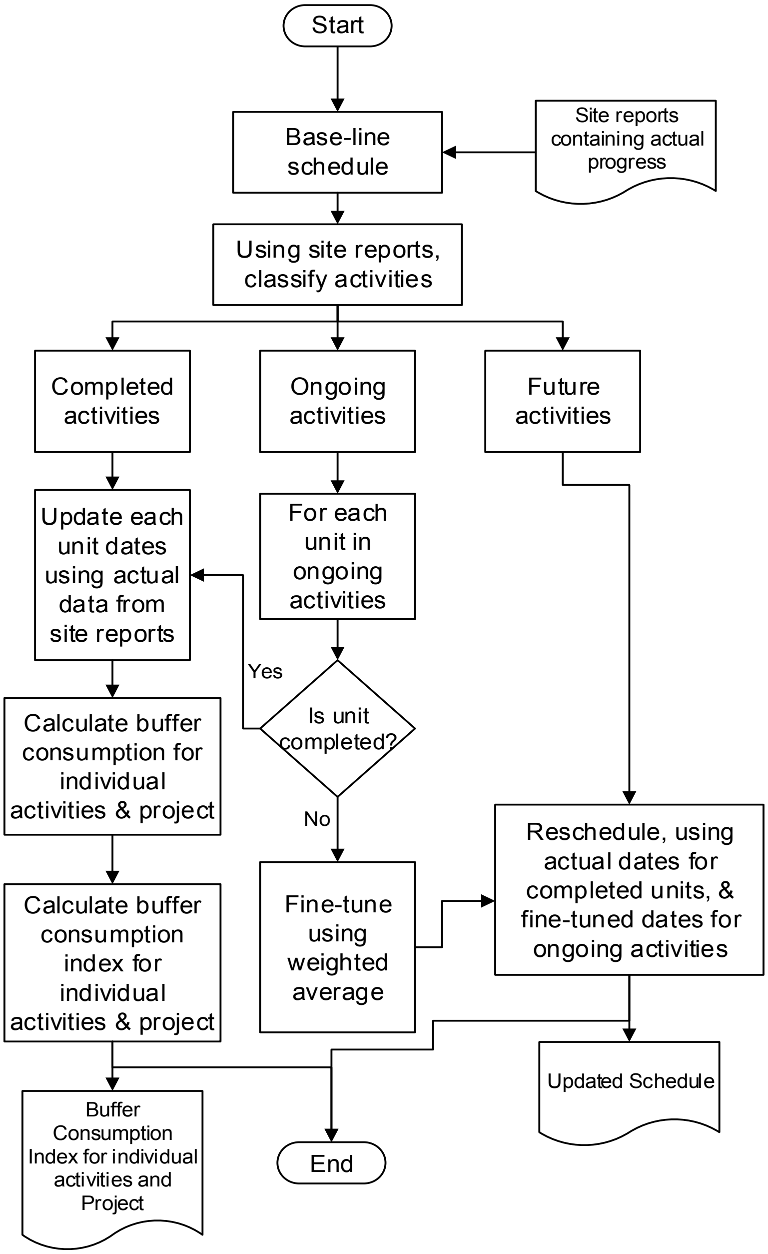


Figure 3.16: Flowchart for Monitoring and Dynamic Rescheduling

Monitoring buffer consumption closely through BCI helps identify deviations from baseline schedule early on. Users can set thresholds for buffer consumption rate, those thresholds can be used to trigger corrective actions. For example, a user can pre-define BCI for different stages of the project. At earlier project stages, users might not be alarmed if BCI is equal to or even slightly exceeds 1.0, as there is still time in the project and there is an opportunity to recover from delays and make up for the lost time. Whereas towards the end of the project, there is not enough time to recover from delays, hence a user might find it critical if BCI is over 0.9. Figure 3.17 shows a possible scheme that can be used by users to evaluate schedule performance using BCI and pre-set thresholds. The green area represents desirable BCI values, and red areas represent alarming values.

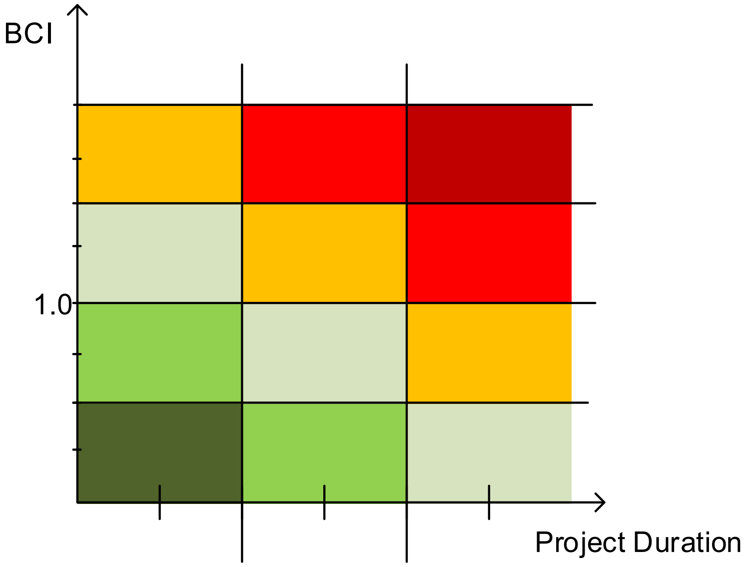


Figure 3.17: BCI Thresholds over Project Duration

3.4 **Acceleration Model**

This third and final model of the presented methodology is developed for schedule acceleration of repetitive construction projects. Project acceleration is a common yet complicated task performed by contractors in construction projects. In repetitive construction projects, contractors often face the challenge of having to accelerate their projects, and they face it with tools that are not suited for repetitive projects and that neglect many of the factors influencing acceleration plans. Project acceleration is usually performed as the need arises, without prior planning during the planning stage. It is performed under the pressure of a project running late and an already defined budget.

This model had two main objectives. The first was to present a model that is capable of addressing uncertainty, such a feature is needed but had not been addressed by any of the existing techniques. The second objective was to account for factors that are considered in the industry while prioritizing activities for acceleration. Recent studies prove that in industry prioritizing and queuing activities to accelerate depends on other criteria in addition to cost slope. Moselhi and Roofigari-Esfahan (2011) distributed a survey among 50 experienced contractors and construction managers in Canada and USA with a response rate of around 80%, accordingly they established other factors commonly taken into consideration by contractors while prioritizing activities for acceleration. These factors include resource availability, risk involved, complexity, logistics, sub-contractor related concerns, number of successors, cash flow constraints, weather and a few other factors.

For practicality and to avoid asking the user to collect and input a huge amount of data, the developed method collates the influential factors identified by Moselhi and Roofigari-Esfahan (2011) in a single criterion named contractor judgment. This allows the contractor to evaluate each activity through taking into consideration factors he sees relevant, in addition to the cost slope factor. This model works in an iterative approach towards achieving one of two main objectives, these are meeting a target project duration or a target project total cost. The proposed method is detailed in the following sections.

3.4.1 **Identifying Activities to Accelerate**

This is the first challenge faced in the acceleration process. Previous algorithms identified activities to accelerate based on relative activity alignment, which proved to correctly identify activities that when accelerated will shorten total project duration. In the proposed model, activities in a schedule are divided and studied as separate units, this gives the proposed model the ability to accommodate typical and non-typical activities. Although studying activities units separately implies performing more calculations, yet it allows more focused assignment of additional acceleration resources. The algorithm presented by Bakry et al. (2014) is utilized to identify activities to accelerate that would lead to project schedule compression. The least aligned unit is identified by calculating the difference in the moment of area trapped between successive activity lines around a virtual vertical access. When a unit has a lower rate than its successor it would have a bigger area trapped between itself and its successor, thus resulting in a bigger moment of area, and vice versa. The difference between a unit's moment of area and its predecessors moment of area is given the symbol Ω (Bakry et al. 2014). Equation 3.27 shows how to calculate Ω for each activity. Higher values of Ω indicate that this activity is less aligned with its successor and vice versa. Units with positive values of Ω are nominated for acceleration, while units with negative values of Ω will not reduce project duration if accelerated. Units with negative values of Ω are called converging activities, these are defined as activities progressing at a higher rate than their successor and their predecessor. Consequently these activities are likely to prolong project duration if accelerated as their start date is advanced and hence their successor's start date. These converging activities can shorten total project duration if they are slowed down (relaxed). Figure 2.10 illustrated the effect of relaxing converging activities.

$$\Omega_{(i)} = Area_{(i)} \text{ X } e_{(i)} - Area_{(i+1)} \text{ X } e_{(i+1)} \qquad \text{Equation 3.27}$$

Where:

$\Omega_{(i)}$ is the value reflecting the degree of misalignment of unit (i)

$Area_{(i)}$ is the area between unit (i) and (i -1),

$e_{(i)}$ is the eccentricity of the center of gravity to the center line of area (i) as shown in Figure 3.18.

A deeper look at the algorithm at hand reveals that studying each unit separately has a weakness. This approach would identify the criticality of an activity based only on the productivity of the assigned crew, regardless of the number of crews working on the same activity in other units. For example if 3 crews assigned to an activity each producing 1 unit per day, their total productivity is 3 units per day. Comparing each activity's rate locally (at each unit separately) and neglecting the global perspective would identify this activity to be more critical than an activity assigned to a single crew producing 2 units per day, although clearly the later activity progresses at a slower rate.

To address this issue the equations for calculating areas and their moment around the imaginary center line had to be modified to include also the number of crews, which enables correctly conveying the rate of an activity according to the productivity and number of crews assigned. Figure 3.18 together with the following equations below demonstrate how identifying the least aligned activity in a repetitive project is formulated. The activity with the largest value for Ω is the least aligned activity. Figure 3.18 below illustrates how least aligned activities are identified through calculating the value of Ω. It shows an illustration of three repetitive activities performed sequentially. For the first unit the virtual vertical centre line is drawn, Area$_{(i)}$ is multiplied by e$_{(i)}$ which is the distance between its center of gravity and the vertical center lines, similarly Area$_{(i+1)}$ and e$_{(i+1)}$ are multiplied. The difference between the two products is the value for Ω. In the illustrated figure activity$_{(i)}$ has a slower rate than activity$_{(i+1)}$, e$_{(i)}$ has a positive value and e$_{(i+1)}$ has a negative value, this will result in a positive value for Ω. The bigger the value of Ω is, the less aligned is activity$_{(i)}$ with activity$_{(i+1)}$.

$\text{Area}_{(i)} = \text{L.Side}_{(i)} + \text{R.Side}_{(i)} / 2$ Equation 3.28

$\text{L.Side}_{(i)} = S_{(i)} - S_{(i-1)}$ Equation 3.29

$\text{R.Side}_{(i)} = [F_{(i)} - F_{(i-1)}] - [D_{(i)}(n-1)/n] + [D_{(i-1)}(n'-1)/n']$ Equation 3.30

If L.Side$_{(i)}$ > R.Side$_{(i)}$

$C_{(i)}=(\text{L.Side}_{(i)}+2 \text{ X } \text{R.Side}_{(i)}) / [3(\text{L.Side}_{(i)}+\text{R.Side})_{(i)}]$ Equation 3.31

$e_{(i)} = C_{(i)} - 0.5$ Equation 3.32

If L.Side$_{(i)}$ > R.Side$_{(i)}$

$C_{(i)}=(\text{R.Side}_{(i)}+2 \text{ X } \text{L.Side}_{(i)}) / [3(\text{R.Side}_{(i)}+\text{L.Side}_{(i)})]$ Equation 3.33

$e_{(i)} = 0.5 - C_{(i)}$ Equation 3.34

$\Omega_{(i)} = \text{Area}_{(i)} \text{ X } e_{(i)} - \text{Area}_{(i+1)} \text{ X } e_{(i+1)}$ Equation 3.35

Where:

Area$_{(i)}$	is the area between activity (i) and (i -1)
S$_{(i)}$	is the start time of activity (i)
S$_{(i-1)}$	is the start time of activity (i-1)
F$_{(i)}$	is the end time of activity (i)
F$_{(i-1)}$	is the end time of activity (i-1)
D$_{(i)}$	is the duration of activity (i)
D$_{(i-1)}$	is the duration of activity (i-1)
n	is the number of crews assigned to activity (i)
n’	is the number of crews assigned to activity (i-1)
C$_{(i)}$	is the distance between the area’s edge to the area’s center of gravity
e$_{(i)}$	is the eccentricity of the center of gravity to the center line of area (i)
Ω$_{(i)}$	is the value reflecting the degree of misalignment of activity (i)

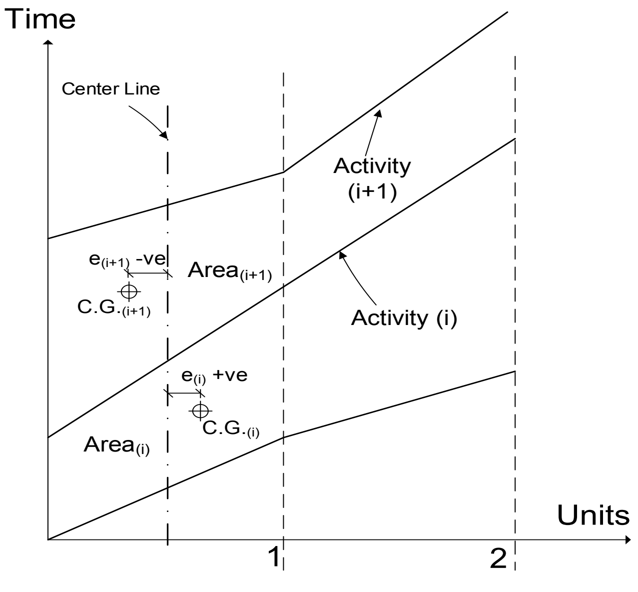


Figure 3.18: Identifying Least Aligned Activities

3.4.2 **Uncertainty in Additional Cost**

Project documentation primarily focuses on the original schedule and the final as built schedule, while project acceleration strategy and how it was reached is commonly left out. Hence, contractors do not usually have sufficient historical data to use while planning project acceleration. In view of these constraints and the unique conditions associated with accelerating projects, it was deemed necessary to account for uncertainties in performing the compression process. This model uses triangular fuzzy numbers as input, to allow users to model uncertainty in the additional acceleration costs, thus helping users make a better choice of acceleration plan for the project. For each unit or activity, a triangular fuzzy number is utilized to express the additional cost required to reduce a unit's duration by 1 unit of time. This number is defuzzified to using the center of area (COA) method (Shaheen et al, 2007), which represents the expected value (EV) using Equation 3.36:

$$\mathrm{EV} = \frac{\mathrm{a+b+c}}{3} \qquad \text{Equation 3.36}$$

This EV value for each activity is the cost slope. A priority ranking based on cost slope ($\mathrm{PCS_i}$) is assigned for each unit based on their respective cost slope. The unit with the least cost slope is assigned high priority of 1.0, the unit with the highest cost slope is assigned the least priority of 5.0, and the remaining units are assigned values by interpolation.

3.4.3 **Contractors' Judgment**

Contractors' judgment is an important criteria that cannot be overlooked when queuing activities for acceleration. It is used in addition to the cost slope. Contractors can use this criteria to indicate their favoring or disfavoring of an activity for queuing in view of influential factors affecting the choice of acceleration plan. These factors include resource availability, risk involved, complexity and logistics, sub-contractor related concerns, number of successors, cash flow constraints, weather and a few other factors (Moselhi and Roofigari-Esfahan 20011). For simplicity and to avoid the need for extensive input data from users, this model collates these factors in a single criterion named contractor's judgment. Contractors employ their experience to express their preferences by a priority ranking (PCJ_i) that they assign to each unit or activity based on a scale of 1.0 to 5.0. A score of 1.0 is assigned to a unit to express the contractors favoring for this unit to be accelerated, a score of 5.0 reflects the contractors disfavoring, while a score of 3.0 reflects the neutral case. Scores of 2.0 and 4.0 represent intermediate values. The chosen scale of 1.0 to 5.0 serves the job of marking and comparing the relative priorities of different units, accordingly using any other scale of 1 to 10 or 1 to 100 would result in the same order of activities as long as the same scale is used for cost slope and contractor's judgment. Understanding the involved influential factors and quantifying their impact to come up with a priority ranking for the contractor's judgment, although in a subjective approach, is still more comprehensive and realistic than prioritizing activities based on cost slope only.

3.4.4 **Schedule Acceleration**

After acceleration costs contractor's judgment have been calculated, each unit has two separate priority rankings, one representing cost slope priority (PCS_i) and one representing contractors' judgment priority (PCJ_i). A joint priority is needed to be produced. This joint priority will be the final ranking criteria that would guide a contractor to where to assign additional acceleration resources. Relative weights are used to set the comparative importance of the two ranking criteria, cost slope and contractors' judgment. These relative weights are used to allow the user to customize the prioritization process according to his specific needs. A bigger weight for the cost slope criteria will make the produced joint priority for an activity more dependent on its cost slope priority and vice versa. For example if a user wishes to build his acceleration plan based equally on the cost slope and the contractor's judgment, he would assign both the weight of 0.5. While if he wishes his decision to be more relying on cost than contractor judgment he would assign a weigh of 0.6 for cost slope and a weight of 0.4 for contractor judgment. A weight of 1.0 for the cost slope criteria and 0.0 for contractor judgment would generate the least cost acceleration plan. As the user gains experience with this technique he will settle on the weight values that suit his needs better. The joint priority is calculated as per Equation 3.37.

$$\mathrm{P_i} = (\mathrm{PCS_i} \times \mathrm{WCS}) + (\mathrm{PCJ_i} \times \mathrm{WCJ}) \qquad \text{Equation 3.37}$$

Where:

P_i	is the joint priority for the i^{th} unit,
PCS_i	is the priority assigned to unit i based on cost slope,

PCJ_i is the priority assigned to unit i based on contractors’ judgment,

WCS is the relative weight assigned to cost slope

WCJ is the relative weight assigned to the contractor’s judgment

After joint priorities have been calculated, now the actual acceleration starts. It is performed through incrementally assigning acceleration resources to the unit with the highest joint priority. For repetitive construction projects, several acceleration strategies were extracted from literature and included in the proposed method. These are (1) working overtime; (2) working double shifts; (3) working weekends and (4) employing more productive crews, while for converging activities strategies for relaxation are (5) using less productive resources or (6) introducing intentional work breaks (Hassanein and Moselhi 2005). The unit’s duration is reduced to the new accelerated duration and the rest of the schedule durations and project costs are recalculated accordingly. The developed method is applicable during the execution phase of the project, i.e. after contract signing and commencement of construction on jobsite. As such, normal cost and normal duration of project activities are considered to have crisp values as would be stipulated in contract documents. The project’s fuzzy total cost is calculated using Equation 3.38 below:

$$\mathrm{FTC} = \mathrm{DC} + \mathrm{IC} + \sum_{i=1}^{n}[a_{ac}, b_{ac}, c_{ac}]_i \qquad \text{Equation 3.38}$$

Where:

FTC is the project’s fuzzy total cost,

DC is project’s direct cost,

IC is project’s indirect cost,

a_{ac}, b_{ac}, c_{ac} are the three values representing the fuzzy acceleration cost of unit (i),

n is the total number of accelerated units.

After each unit is accelerated, the model recalculates of the schedule and the new project total cost and duration are plotted. These recalculations include recalculating **Ω**, as activities relative alignment changes when they are accelerated. The above procedure is repeated in an iterative manner until the targeted cost or duration are achieved. Figure 3.19 shows the detailed flowchart of the Acceleration Model.

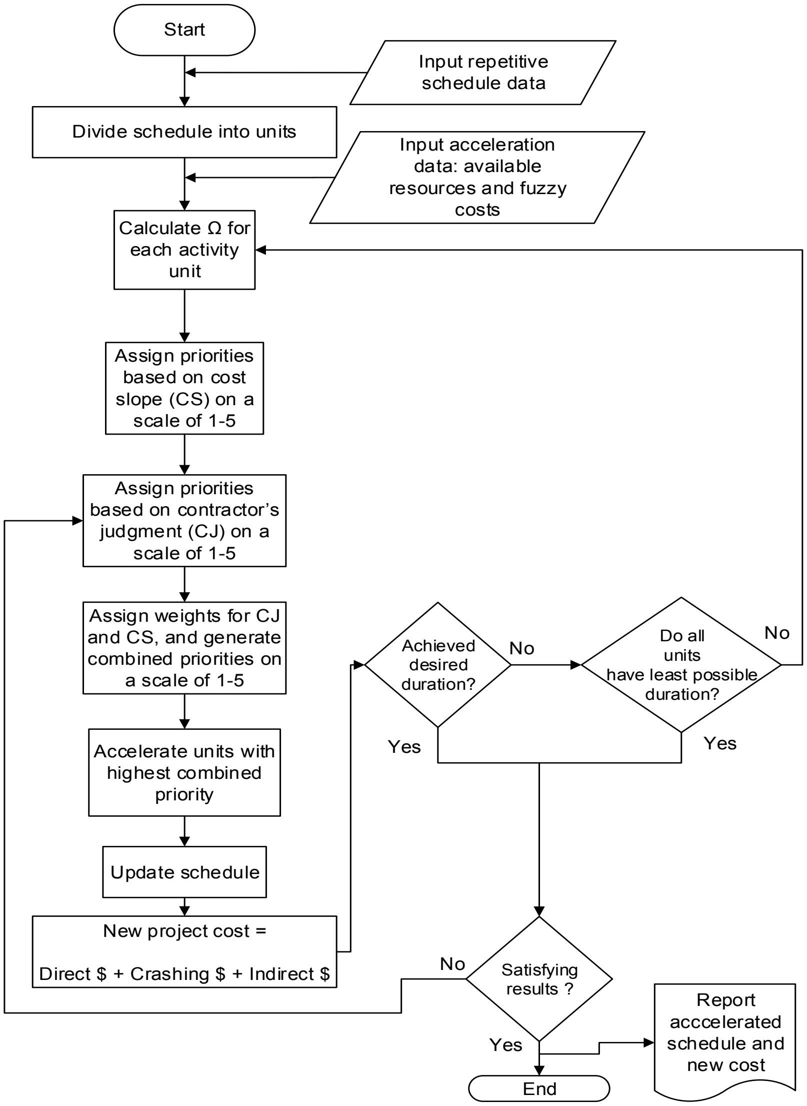


Figure 3.19: Flowchart for Acceleration Model

Chapter 4: Implementation of Schedule Optimization

4.1 Introduction

This chapter explains how the schedule optimization under uncertainty was developed into a software prototype. The dynamic programming utilizing fuzzy input and producing fuzzy output requires a large amount of calculations for calculating durations, dates, and costs and continuous evaluation of objective functions, which formed the motivation for automating this part of the methodology. The optimization was modeled using object oriented programming. In object oriented programming a complex problem is decomposed and modeled as objects. Objects can sometimes represent physical objects, such as a construction crew, or non-physical objects such as a path of optimum decisions. Each object contain data describing the object, and methods governing the behavior of the object. For example an object representing a concrete poring crew would contain data such as the crew's productivity and cost, and would contain as well methods allowing the object to calculate the needed duration for a work quantity and the start and end dates (Clark 2006). Similar objects can be grouped in classes. For example objects representing different crews would be grouped in a class, as they contain similar fields of data and similar methods.

After the problem is broken down and represented as objects, which are grouped into classes, object oriented programming offers various concepts to facilitate modelling. Abstraction is a concept that creates a blueprint for an object, defining its responsibilities and permitted actions. Encapsulation is another key feature of object oriented programming, it allows concealing the data in an object, and limiting access to it through the methods contained in the object, thus providing a more regulated exchange of data between different objects. The polymorphism concept is what allows similar objects to respond differently to the same command. For example asking an activity object to set its starting date, the first activity object would utilize the project start date, while other activities would utilized their predecessors end date. Finally, the concept of inheritance is another useful feature of object oriented programming, it allows changes introduced to a class to be passed on to all objects belonging to that class, thus making working with objects easier and faster (Clark 2006).

4.2 Design

The design of the object oriented model passed through the typical tasks of software design. It started by a listing of the system requirements which included all the features and capabilities that are needed in the software. The next step was developing the use cases. Use cases describe the software functionality from the point of view of an external user, it shows different possible interactions between the software and other entities. It also lists the necessary preconditions and post conditions of each case. After use cases were developed they are changed into sequence diagrams. Sequence diagrams documents the sequence of interactions of the model classes. An overview of the input and output of the software prototype is illustrated in Figure 4.1. The optimization procedure was modelled using three main classes. A Project class, an Activity Class and a Crew class.

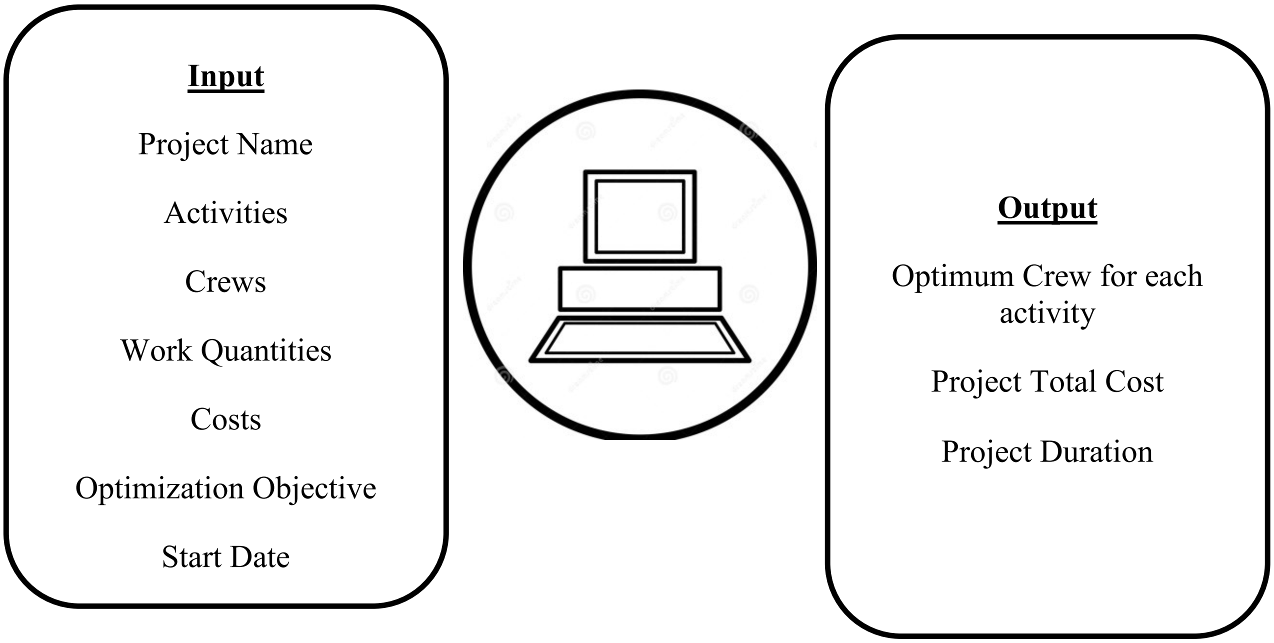


Figure 4.1: Input and Output Overview

4.2.1 Project Class

This class holds the general details of the project. As seen in Table 4.1, this class is responsible for holding the list of activities forming the project, number of repetitive units, project start date, optimization objective and project indirect cost.

Table 4.2 shows the main functions this class carries out, these are functions that add or remove activities, launch the optimization procedure, record optimum crew formation and project duration.

Table 4.1: Main Data Members of Project Class

Data	**Type**
_activities	List<Activity>
_startingDate	DateTime
_optimumPathEvaluationMethod	ByCost, or ByDuration
EnumOptimumPathEvaluationMethod	Enum
_projectIndirectCostPerDay	Float
NumberOfUnits	Integer

Table 4.2: Main Function Members of Project Class

Method	**Returns**
AddActivity()	void
RemoveActivity()	void
RunDynamicProgrammingEngine()	void
CreateOptimumPaths()	Void
GetTotalDuration()	FuzzyNumber

4.2.2 Activity Class

This activity class represents activities forming a project. The class is designed once, and then an instance of this class is generated to represent each activity in the project. Each activity holds a list for quantities, with the number of entries matching the number of repetitive units. An activity also holds a list of crews available for this activity. Other than its Id and Name, each activity also has a link to attach it to its parent project.

Table 4.3: Main Data Members of Activity Class

Data	**Type**
_quantities	List<QuantityUnit>
_crews	List<Crew>
_parentProject	Project
_materialCost	FuzzyNumber
Id	String
Name	String

The main methods included in an activity class are methods to add or remove a crew for this activity, a method to add quantities of work, and a method to calculate the material cost for that activity. Unlike labour and equipment costs, material cost is the same for an activity regardless of the chosen crew, which is why material cost is added to the Activity class and not the Crew class.

Table 4.4: Main Function Members of Activity Class

Method	Returns
AddCrew()	Void
RemoveCrew()	Void
CreateQuantities()	Void
ActivityMaterialCost()	FuzzyNumber

4.2.3 Crew Class

This class is designed to represent the crews in the project. Whenever a new crew is added to the list of crews available for an activity, an instance of this class is created. It carries information describing the crew, such as its name, the activity it can work on, its productivity and cost. Each crew has a Boolean variable named _canBeDropped, this variable is originally set to True for each crew, once a crew is identified as optimum for that activity the value for that variable is changed to False.

Table 4.5: Main Data Members of Crew Class

Data	Type
_parentActivity;	Activity
_startingDates	List <FuzzyNumber>
_canBeDropped	bool
Name	String
Productivity	FuzzyNumber
Cost	CostUnit
_optimumPredecessorByDuration;	Crew
_optimumPredecessorByCost;	Crew
maxDurationDifference	FuzzyNumber
duration	DurationUnit
TotalDurations	List<FuzzyNumber>
DurationsPerQuantities	List<FuzzyNumber>

The variable names DurationPerQuantity is responsible for dividing quantities of work by productivities to each unit's duration. Units' durations are then cumulated to get the

TotalDurations for that activity. Shifting the total durations allows getting the – StartingDates of the activity and maintains resource work continuity. The main functions in the Crew Class are responsible for calculating starting dates and identifying optimum predecessors either by cost or by duration.

Table 4.6: Main Function Members of Crew Class

Method	**Returns**
CalculateOptimumPredecessor()	Crew
CalculateStartingDates()	FuzzyNumber
GetOptimumPredecessorCrewByLeastCost()	Crew
GetOptimumPredecessorCrewByLeastDuration()	Crew

4.2.4 Fuzzy Numbers

A special class is created to facilitate the computational implementation of this model. This class defines a custom made data type called fuzzy number, it consists of three variables of type Float. It also contains the basic arithmetic operations for adding, subtracting, multiplying and dividing fuzzy numbers. Instances of this class are used to represent productivities, costs, durations and quantities. Figure 4.2 shows the base class FuzzyNumber and the instances derived from it.

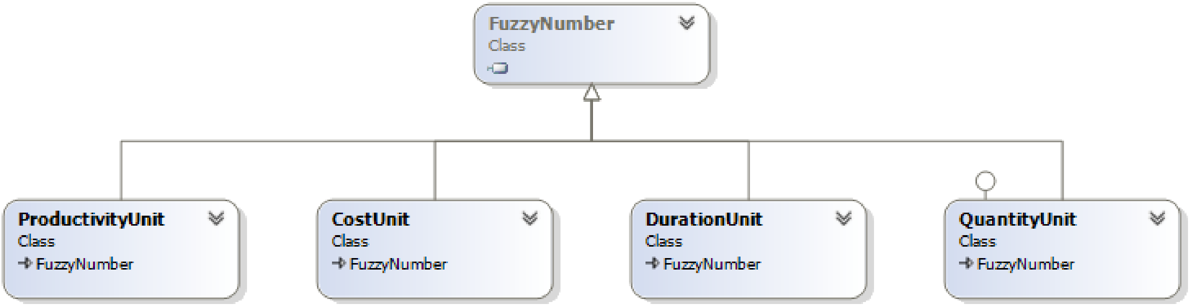
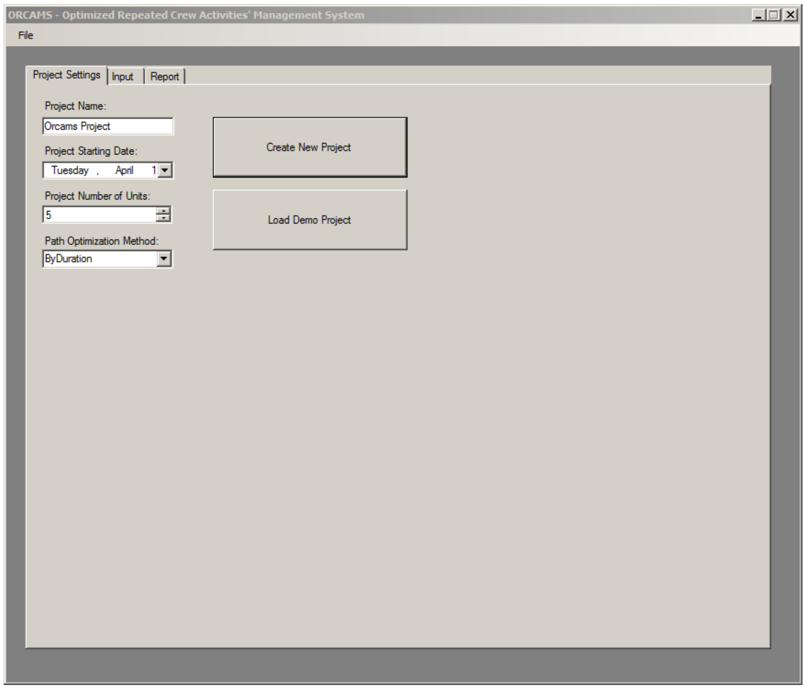


Figure 4.2: Instances Derived from FuzzyNumber Class

4.3 **OSRP**

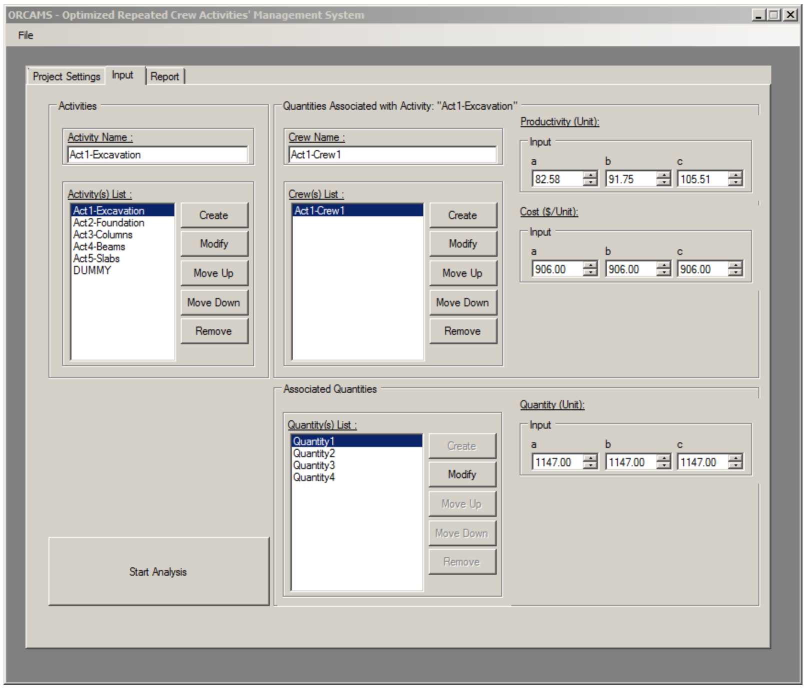
OSRP is the name given to the developed prototype. It stands for Optimized Scheduler of Repetitive Projects. The prototype was coded using C# as a programming language, it comes as a standalone Microsoft Windows application that can be run on Windows XP or later versions. The interaction with the software is through a user friendly interface comprising 2 input screens and one output screen. The first input screen contains the basic input related to the project, such as project name, indirect cost, number of repetitive units and the required optimization objective. Figure 4.3 shows the input screen. After the user completes basic input, the second input screen is launched, it allows entering the details of the project. Toggling between different screens is easy through selecting tabs at the top of the window. The second input screen, illustrated in Figure 4.4, lets the user input different activities performed in the project. For each of those activities the user inputs a list of available crews and quantities of work for each unit. For each of the input crews the user identifies its productivity and cost. Crews' productivities and costs and quantities of work for each unit are all entered as fuzzy numbers, through the smaller text boxes to the right of the screen.

After completing the input, the "Start Analysis" button at the bottom left corner of the second screen launches the optimization application. After the run is complete the user switches to the third and final tab, the "Report" tab, to see the output of the run. The output report includes project name, number of repetitive units and the optimization objective the user had chosen. The rest of the report is a list of each activity in the project and the optimum crew chosen for that activity. At the end of the report the total project duration and cost are listed in fuzzy numbers.



Project Name: Orcams Project
Project Starting Date: Tuesday , April 1
Project Number of Units: 5
Path Optimization Method: ByDuration

Figure 4.3: Project Input Screen



Activity Name : Act1-Excavation

Activity(s) List :
Act1-Excavation
Act2-Foundation
Act3-Columns
Act4-Beams
Act5-Slabs
DUMMY

Quantities Associated with Activity: "Act1-Excavation"

Crew Name : Act1-Crew1

Crew(s) List :
Act1-Crew1

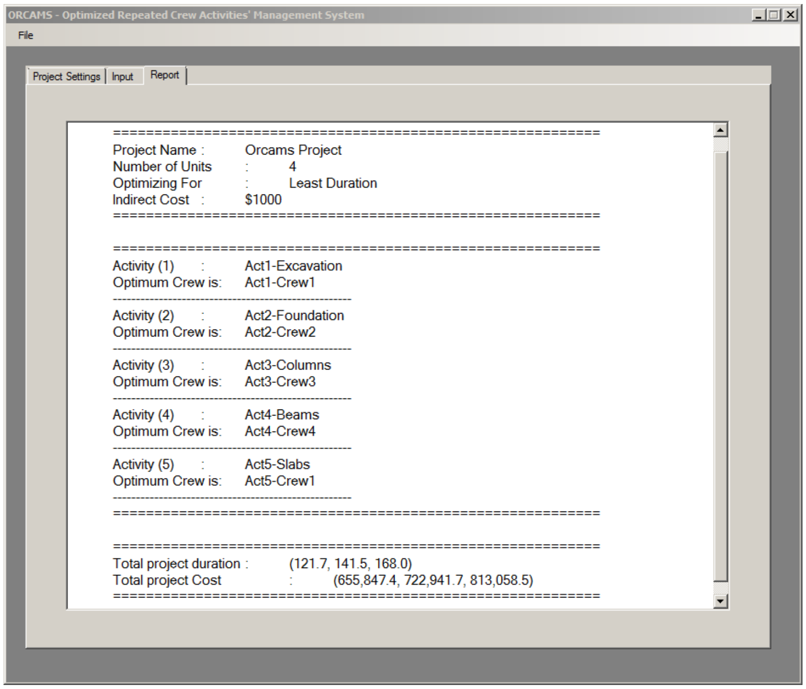
Productivity (Unit):
Input
a 82.58 b 91.75 c 105.51

Cost ($/Unit):
Input
a 906.00 b 906.00 c 906.00

Quantity(s) List :
Quantity1
Quantity2
Quantity3
Quantity4

Quantity (Unit):
Input
a 1147.00 b 1147.00 c 1147.00

Figure 4.4: Activities and Crews Input Screen



```
============================================================
Project Name :          Orcams Project
Number of Units         :       4
Optimizing For          :       Least Duration
Indirect Cost   :       $1000
============================================================

============================================================
Activity (1)    :       Act1-Excavation
Optimum Crew is:        Act1-Crew1
------------------------------
Activity (2)    :       Act2-Foundation
Optimum Crew is:        Act2-Crew2
------------------------------
Activity (3)    :       Act3-Columns
Optimum Crew is:        Act3-Crew3
------------------------------
Activity (4)    :       Act4-Beams
Optimum Crew is:        Act4-Crew4
------------------------------
Activity (5)    :       Act5-Slabs
Optimum Crew is:        Act5-Crew1
------------------------------
============================================================

============================================================
Total project duration :        (121.7, 141.5, 168.0)
Total project Cost              :       (655,847.4, 722,941.7, 813,058.5)
============================================================
```

Figure 4.5: Output Report

Chapter 5: Case Studies

5.1 Introduction

This chapter presents the analyzed case studies. Several case studies were drawn from literature and analyzed during different stages of the research. The purpose was to experiment and showcase different features of different models in the methodology, and to demonstrate the benefits and limitations of the developed models. Two different case studies are explained in details in this chapter. A first case study was analyzed to evaluate the performance of the three built models, and then a second case study was analyzed using the Acceleration Model only. The reason behind using those two case studies is trying to use case studies that have been used for similar previous methodologies to allow comparing results whenever possible.

This chapter is divided into three main sections following this brief introduction section. The first section presents the case study drawn from literature and analyzed to evaluate the performance of the Optimized Scheduling and Buffering Model. It examined the capabilities of the model in optimizing for least cost and least duration objectives. It then further extended to demonstrate schedule defuzzification and buffer sizing and insertion. The following section presents the two case studies drawn from literature and analyzed to evaluate the performance of the Acceleration Model. The final section of this chapter presents the output analysis of the case studies and discusses the findings.

5.2 Optimized Scheduling and Buffering

This section explains the case study drawn from literature and analyzed to demonstrate and evaluate the performance of the Optimized Scheduling and Buffering Model. This case study is for a hypothetical repetitive construction project. It was initially presented by Selinger (1980), and utilized later by many researchers with a few modifications for testing different models and algorithms (Russell and Caselton, 1988, Moselhi and El-Rayes, 1993, El-Rayes and Moselhi, 2001, Hyari and El- Rayes, 2004, Nassar, 2005, Hyari and El-Rayes, 2006 and Liu and Wang, 2007). The described repetitive project is a three span reinforced concrete bridge. The bridge is divided into four segments (units), each comprising five repetitive activities; those activities are Excavation, Foundation, Columns, Beams and Slabs. Logical relations between activities are finish to start with no lag time. Figure 5.1 shows an abstract illustration of the four bridge units and their activities, while Table 5.1 shows the activities quantities in m^3 for the original case study. It can be seen that the project activities are non-typical repetitive activities, as work quantities for each activity vary from a unit to another.

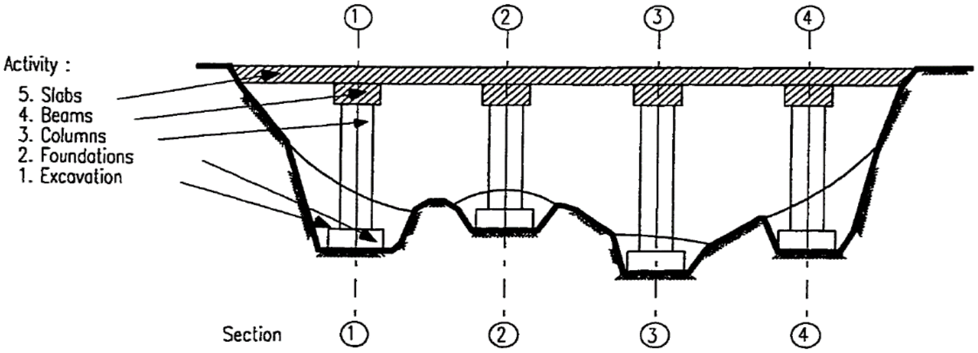


Figure 5.1: Three Span RC Bridge (Selinger 1980)

Table 5.1: Original Activity Quantities

Activity	**Quantity (m^3)**			
	Section 1	**Section 2**	**Section 3**	**Section 4**
Excavation	1147	1434	994	1529
Foundation	1032	1077	943	896
Columns	104	86	129	100
Beams	85	92	101	80
Slabs	0	138	114	145

For each of the activities a different number of crews are available to select from for construction, each characterized by a different cost and productivity. Table 5.2 shows the number of available crews for each of the project activities. Choosing crews having different relative rates for each activity would results in a unique crew formation for the project, which would result in the possible execution of the project in a different duration and budget. Table 5.3 shows the original crews productivity. The project's indirect cost is 1,000$/day, while the components of the direct cost are summarized in Table 5.4.

Table 5.2: Available Crews for Each Activity

Activity	**Excavation**	**Foundation**	**Columns**	**Beams**	**Slabs**
Available number of Crews	1	3	3	4	2

Table 5.3: Original Crews Productivity

Activity	Crew	Crew Productivity m^3/day
Excavation	1	91.75
Foundation	1	89.77
	2	71.81
	3	53.86
Columns	1	5.73
	2	6.88
	3	8.03
Beams	1	9.90
	2	8.49
	3	7.07
	4	5.66
Slabs	1	8.73
	2	7.76

Table 5.4: Project Direct Cost Components

Activity	Crew	Labour Cost (\$/day)	Equipment Cost (\$/day)	Material Cost (\$/m^3)
Excavation	1	566	340	-
Foundation	1	3,804	874	92
	2	2,853	655	
	3	1,902	436	
Columns	1	1,875	285	479
	2	2,438	371	
	3	3,000	456	
Beams	1	3,931	315	195
	2	3,238	259	
	3	2,544	204	
	4	1,850	148	
Slabs	1	2,230	177	186
	2	1,878	149	

Although the above case study was analyzed more than once before, the relevant results are those of El-Rayes (1997). El-Rayes used the above case to test his methodology for optimizing repetitive projects using dynamic programming. His aim was to find the crew formation that would complete the project at the least cost. His results were compared to current analysis results to evaluate this model's features. To evaluate the performance of the Optimized Scheduling and Buffering Model the above case study was analyzed three times. The first run was using the above mentioned data without introducing any changes, to find the least cost schedule to allow comparing the results to those of El-Rayes. The second run was also aiming at finding the least cost schedule but while accounting for uncertainty. The third and final run also accounted for uncertainty but aimed to find the least duration schedule.

5.2.1 **Least Cost Optimization**

This section contains the details and results of the first two runs. The first run was using deterministic input to find the crew formation that would yield the least cost schedule, while the second run had the same goal but it modeled uncertainty in different input parameters as will be detailed.

5.2.1.1 Verification Run

The purpose the first run is to verify the model. This run tested the model's ability to correctly complete the optimization procedure using fuzzy numbers, and to trust that the equations and algorithmic implementation are all verified. This run, similar to El-Rayes (1997), aimed at finding the optimum crew formation that would yield least cost schedule. The deterministic input utilized by El-Rayes (1997) was mapped into triangular fuzzy numbers. For Example a crew productivity of 91.8m^3/day was input as (91.8, 91.8, 91.8) m^3/day. The available number of crews for various activities can produce 72 different crew formations to execute the project. The solution started by listing all possible crew formations for the first 2 activities, and calculated their respective durations and costs. For each of the crews available for the Foundation activity (second activity), the optimum predecessor was identified, based on the value of the objective function (Equation 3.7 used for least total cost) calculated for the two activities. In this specific case the first activity Excavation had only 1 available crew, given the name E1, so this was the optimum predecessor for all Foundation activity crews. Gradual expansion takes place and the Columns activity was included, which had 3 available crews C_1, C_2 and C_3; these 3 crews were matched in formations with all previous activities' crews EF_1, EF_2 and EF_3, which resulted in 9 possible crew formations. The duration and cost of each formation were calculated and the optimum predecessor for each of the 3 Columns crews was identified separately for the three activities using the value of the least total cost objective function. Out of the 9 possible formations, the 3 optimum formations for the first 3 activities having the least total costs were EF_3C_1, EF_3C_2, and EF_3C_3. These 3 formations were then included in the next expansion of the problem which included the Beams activity, while the other 6 alternatives will were eliminated. The computational process continued until all activities were investigated and the optimum crew formation was identified. Table 5.5 shows the details of the run made to verify the model. The first column on the left shows the activity and the second column shows the available crews for that activity. The third column shows the predecessor crews being matched up with the current crew in a crew formation. The columns following that are the early start dates of each of the four sections and the finish date for the last section. The direct cost for each crew formation was calculated by adding its own direct cost to its predecessors' direct cost. The indirect cost is not shown in the schedule due to space limitations, it was calculated as 1,000 $/day X total project duration. The total cost is the sum of the indirect and direct costs. As the run was in deterministic numbers, all costs and durations were represented by three equal numbers. Local optimum predecessor was chosen based on the formation producing the least total cost. The run identified the optimum crew formation to be E1 F3 C1 B4 S2 yielding a least total cost of $1,460,203 and a corresponding duration of 143 days. Figure 5.2 shows the generated schedule.

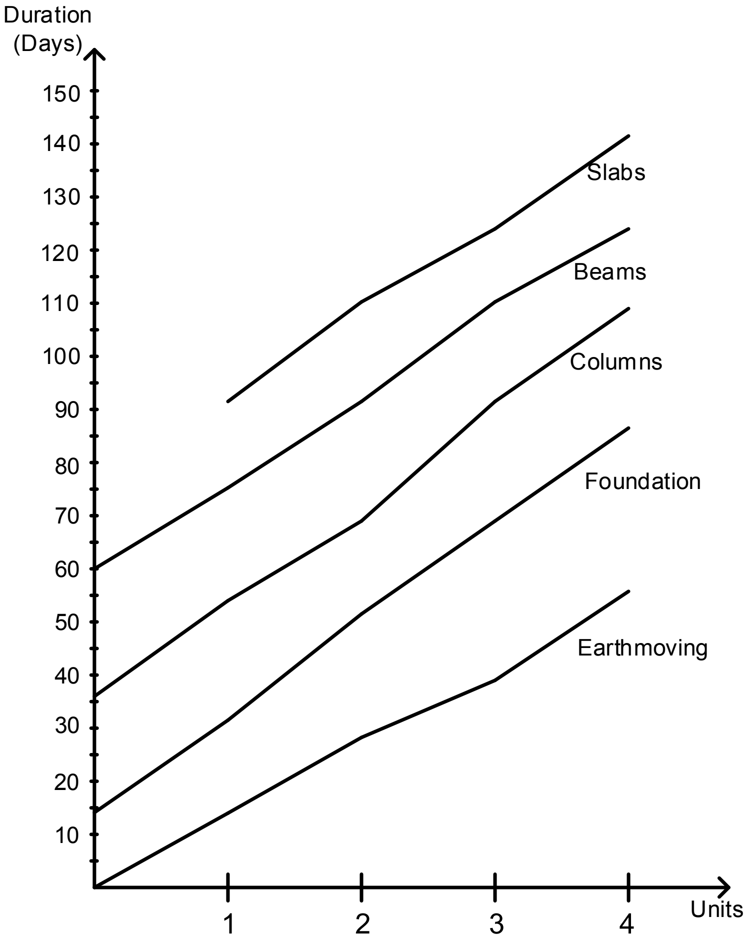


Figure 5.2: Least Cost Schedule Generated with Deterministic Input

Table 5.5: Least Cost Optimization (Verification run)

Activity	Crew	Predecessor crews	Early Start (Days)												Early Finish (Days)			Own direct Cost (1,000$)			Cum. Pred. direct Cost (1,000 $)			Total Cost (1,000 $)			Local Optimal predecessor
			Section 1			Section 2			Section 3			Section 4															
			a	b	c	A	b	c	a	b	c	a	b	b	a	b	c	a	b	c	a	B	c	a	b	c	
Excavation	1	-	0	0	0	13	13	13	28	28	28	39	39	39	56	56	56	50	50	50	-	-	-	106	106	106	-
Foundation	1	E	22	22	22	33	33	33	45	45	45	56	56	56	66	66	66	569	569	569	50	50	50	685	685	685	1
	2	E	14	14	14	28	28	28	43	43	43	56	56	56	69	69	69	556	556	556	50	50	50	675	675	675	1
	3	E	13	13	13	32	32	32	52	52	52	69	69	69	86	86	86	535	535	535	50	50	50	671	671	671	1
Columns	1	EF1	33	33	33	51	51	51	66	66	66	89	89	89	106	106	106	359	359	359	619	619	619	1,084	1,084	1,084	
	1	EF2	28	28	28	46	46	46	61	61	61	84	84	84	101	101	101	359	359	359	606	606	606	1,066	1,066	1,066	3
	1	EF3	36	36	36	54	54	54	69	69	69	92	92	92	109	109	109	359	359	359	585	585	585	1,053	1,053	1,053	
	2	EF1	33	33	33	48	48	48	61	61	61	79	79	79	94	94	94	372	372	372	619	619	619	1,085	1,085	1,085	
	2	EF2	29	29	29	44	44	44	56	56	56	75	75	75	90	90	90	372	372	372	606	606	606	1,068	1,068	1,068	3
	2	EF3	42	42	42	57	57	57	69	69	69	88	88	88	103	103	103	372	372	372	585	585	585	1,059	1,059	1,059	
	3	EF1	33	33	33	46	46	46	57	57	57	73	73	73	85	85	85	381	381	381	619	619	619	1,086	1,086	1,086	
	3	EF2	33	33	33	45	45	45	56	56	56	72	72	72	85	85	85	381	381	381	606	606	606	1,072	1,072	1,072	3
	3	EF3	46	46	46	59	59	59	70	70	70	86	86	86	98	98	98	381	381	381	585	585	585	1,064	1,064	1,064	
Beams	1	EF3C1	81	81	81	90	90	90	99	99	99	109	109	109	117	117	117	223	223	223	944	944	944	1,284	1,284	1,284	
	1	EF3C2	74	74	74	83	83	83	92	92	92	102	102	102	111	111	111	223	223	223	957	957	957	1,291	1,291	1,291	
	1	EF3C3	70	70	70	79	79	79	88	88	88	98	98	98	106	106	106	223	223	223	966	966	966	1,296	1,296	1,296	1
	2	EF3C1	76	76	76	86	86	86	97	97	97	109	109	109	119	119	119	217	217	217	944	944	944	1,279	1,279	1,279	
	2	EF3C2	70	70	70	80	80	80	91	91	91	103	103	103	112	112	112	217	217	217	957	957	957	1,286	1,286	1,286	
	2	EF3C3	66	66	66	76	76	76	86	86	86	98	98	98	108	108	108	217	217	217	966	966	966	1,291	1,291	1,291	1
	3	EF3C1	70	70	70	82	82	82	95	95	95	109	109	109	120	120	120	209	209	209	944	944	944	1,273	1,273	1,273	
	3	EF3C2	63	63	63	75	75	75	88	88	88	103	103	103	114	114	114	209	209	209	957	957	957	1,280	1,280	1,280	
	3	EF3C3	61	61	61	73	73	73	86	86	86	100	100	100	111	111	111	209	209	209	966	966	966	1,286	1,286	1,286	1
	4	EF3C1	60	60	60	75	75	75	92	92	92	110	110	110	124	124	124	196	196	196	944	944	944	1,263	1,263	1,263	
	4	EF3C2	57	57	57	72	72	72	88	88	88	106	106	106	120	120	120	196	196	196	957	957	957	1,273	1,273	1,273	
	4	EF3C3	59	59	59	74	74	74	90	90	90	108	108	108	122	122	122	196	196	196	966	966	966	1,285	1,285	1,285	1

Table 5.5 Continued

Activity	Crew	Predecessor crews	Early Start (Days)												Early Finish (Days)			Own direct Cost (1,000$)			Cum. Pred. direct Cost (1,000 $)			Total Cost (1,000 $)			Local Optimal predecessor
			Section 1			Section 2			Section 3			Section 4															
			a	b	c	a	b	c	a	b	c	a	b	c	a	b	c	a	b	c	a	b	c	a	b	c	
Slabs	1	E1 F3 C1 B1	-	-	-	99	99	99	115	115	115	128	128	128	144	144	144	183	183	183	1,167	1,167	1,167	1,495	1,495	1,495	
	1	E1 F3 C1 B2	-	-	-	97	97	97	113	113	113	126	126	126	143	143	143	183	183	183	1,161	1,161	1,161	1,487	1,487	1,487	
	1	E1 F3 C1 B3	-	-	-	95	95	95	111	111	111	124	124	124	140	140	140	183	183	183	1,153	1,153	1,153	1,476	1,476	1,476	
	1	E1 F3 C1 B4	-	-	-	95	95	95	111	111	111	124	124	124	140	140	140	183	183	183	1,140	1,140	1,140	1,463	1,463	1,463	4
	2	E1 F3 C1 B1	-	-	-	99	99	99	117	117	117	131	131	131	150	150	150	178	178	178	1,167	1,167	1,167	1,495	1,495	1,495	
	2	E1 F3 C1 B2	-	-	-	97	97	97	115	115	115	130	130	130	148	148	148	178	178	178	1,161	1,161	1,161	1,487	1,487	1,487	
	2	E1 F3 C1 B3	-	-	-	-	95	95	113	113	113	127	127	127	146	146	146	178	178	178	1,153	1,153	1,153	1,476	1,476	1,476	
	2	E1 F3 C1 B4	-	-	-	92	92	92	110	110	110	124	124	124	143	143	143	178	178	178	1,140	1,140	1,140	1,460	1,460	1,460	4

5.2.1.2 Run with Fuzzy Input

After the first run utilized deterministic input mapped into triangular fuzzy numbers to verify the model, this run aimed at evaluating the impact of accounting for uncertainty in different input variables. This run utilized fuzzy input and aimed at finding the crew formation that would yield the least cost schedule. The results of this run in terms of the identified optimum crew formation and the generated optimum schedule are compared to results of El-Rayes (1997) as he shared the same optimization objective and to those of the previous run. The first task was to change the deterministic project data into fuzzy data. Uncertainty was randomly chosen to effect crews' productivity for activities Excavation, Foundation and Slabs, and in the work quantities for the Columns activity. To change the deterministic original case data into triangular fuzzy numbers, the original deterministic number was used as the "b" value of the fuzzy number, then it was multiplied once by a factor less than 1.0 to get the "a" value, and once by a factor greater than 1.0 to get the "c" value. These factors were chosen randomly to create a triangular fuzzy number. Table 5.6 shows the factors randomly selected to fuzzify different input parameters. The "D" in the table represents the deterministic value in the original case study presented by El-Rayes (1997). These factors were applied to the deterministic figures to produce the fuzzy numbers that were used for the second run. The produced fuzzy numbers are listed in Table 5.7. The optimized scheduling model was run to find the crew formation that would generate the least cost schedule. Run details are summarized in Table 5.8. This run identified the optimum crew formation to be E1 F3 C1 B4 S2 yielding a fuzzy least total cost of ($1,338,562.9, $1,460,231.0, $1,630,342.8) and a corresponding fuzzy duration of (121.7, 142.9, 172.5) days. To allow plotting the generated least cost schedule, the durations were defuzzified and the schedule was regenerated using EV for each duration. Figure 5.3 shows the defuzzified least cost schedule having a total duration of 144 days, the figure also marks the original deterministic least cost schedule of El-Rayes (1997) having a duration of 143 days.

Table 5.6: Factors for Fuzzifying Original Case Study

Activity	Crew	Quantities (m^3)												Fuzzy Crew Output m^3/day		
		Section 1			Section 2			Section 3			Section 4			a	b	c
Excavation	1	D			D			D			D			0.9 × D	D	1.15 × D
Foundation	1	D			D			D			D			0.95 × D	D	1.15 × D
	2													0.85 × D	D	1.15 × D
	3													0.8 × D	D	1.25 × D
Columns	1	0.9 × D	D	1.25 × D	0.95 × D	D	1.2 × D	0.8 × D	D	1.2 × D	0.95 × D	D	1.25 × D	D		
	2													D		
	3													D		
Beams	1	D			D			D			D				D	
	2														D	
	3														D	
	4														D	
Slabs	1	D			D			D			D			0.9 × D	D	1.2 × D
	2													0.9 × D	D	1.25 × D

Table 5.7: Fuzzy Numbers after Applying Factors

Activity	Crew	Quantities (m^3)												Fuzzy Crew Output m^3/day		
		Section 1			Section 2			Section 3			Section 4			a	b	c
Excavation	1	1147			1434			994			1529			82.58	91.75	105.51
Foundation	1	1032			1077			943			896			85.28	89.77	103.24
	2													61.04	71.81	82.58
	3													43.09	53.86	67.33
Columns	1	93.6	104	130	81.7	86	103.2	103.2	129	154.8	95	100	125	5.73		
	2													6.88		
	3													8.03		
Beams	1	85			92			101			80			9.90		
	2													8.49		
	3													7.07		
	4													5.66		
Slabs	1	0			138			114			145			7.86	8.73	10.48
	2													6.98	7.76	9.70

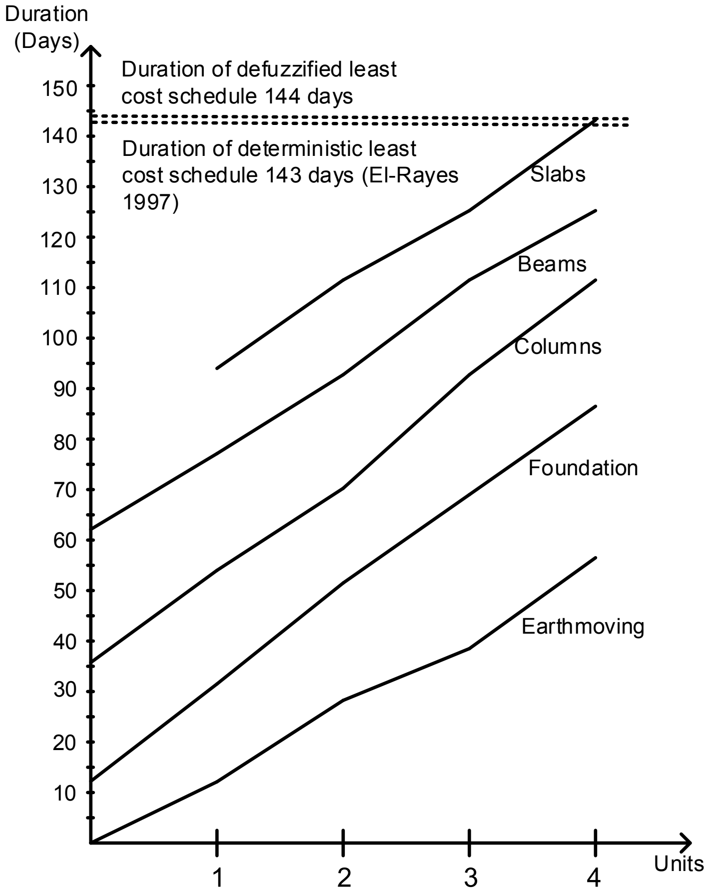


Figure 5.3: Defuzzified Least Cost Schedule

Table 5.8: Least Cost Optimization (Under Uncertainty)

Activity	**Crew**	**Predecessor crews**	**Early Start (Days)**												**Early Finish (Days)**			**Own direct Cost (1,000$)**			**Cum. Pred. direct Cost (1,000 $)**			**Total Cost (1,000 $)**			**Local Optimal predecessor**
			Section 1			Section 2			Section 3			Section 4															
			a	b	c	a	b	c	a	b	c	a	b	b	a	b	c	a	b	c	a	b	c	a	b	c	
Excavation	1	-	0	0	0	11	13	14	24	28	31	34	39	43	48	56	62	44	50	56	-	-	-	92	106	118	-
Foundation	1	E	20	22	26	29	33	38	40	45	51	49	56	62	58	66	72	542	569	580	44	50	56	644	685	708	1
	2	E	12	14	14	24	28	31	38	43	49	49	56	64	60	69	79	531	556	590	44	50	56	635	675	725	1
	3	E	11	13	14	26	32	38	42	52	63	56	69	85	70	86	106	500	535	577	44	50	56	614	671	739	1
Columns	1	EF1	29	33	38	46	51	61	60	66	79	78	89	106	95	106	128	320	359	439	586	619	636	1,000	1,084	1,203	3
	1	EF2	24	28	31	41	46	54	55	61	72	73	84	99	90	101	121	320	359	439	575	606	646	984	1,066	1,206	
	1	EF3	32	36	44	48	54	67	63	69	85	81	92	112	97	109	134	320	359	439	544	585	633	961	1,053	1,206	
	2	EF1	29	33	38	43	48	57	55	61	72	70	79	94	84	94	113	331	372	455	586	619	636	1,001	1,085	1,204	3
	2	EF2	25	28	31	38	43	50	50	56	65	65	74	88	79	89	106	331	372	455	575	606	646	985	1,067	1,207	
	2	EF3	31	42	51	44	57	70	56	69	85	71	88	107	85	103	125	331	372	455	544	585	633	961	1,059	1,214	
	3	EF1	29	33	38	41	46	54	51	57	67	64	73	86	76	85	102	340	381	467	586	619	636	1,002	1,086	1,204	3
	3	EF2	27	33	35	39	45	52	49	56	64	62	72	84	74	85	99	340	381	467	575	606	646	988	1,072	1,212	
	3	EF3	41	46	57	52	59	73	63	70	86	75	86	106	87	98	121	340	381	467	544	585	633	971	1,064	1,221	
Beams	1	EF3C1	69	81	106	78	90	115	87	99	124	97	109	134	105	117	142		223		864	944	1,073	1,192	1,284	1,438	1
	1	EF3C2	57	75	97	65	84	106	75	93	115	85	103	125	93	111	133		223		876	957	1,089	1,192	1,291	1,445	
	1	EF3C3	59	70	93	68	79	102	77	88	111	87	98	121	95	106	129		223		884	966	1,100	1,202	1,296	1,452	
	2	EF3C1	65	76	101	75	86	111	85	97	122	97	109	134	107	119	143		217		864	944	1,073	1,188	1,279	1,433	1
	2	EF3C2	52	70	93	62	80	103	73	91	114	85	103	125	94	112	135		217		876	957	1,089	1,187	1,286	1,441	
	2	EF3C3	54	65	88	64	75	98	75	86	109	87	98	121	96	107	130		217		884	966	1,100	1,197	1,290	1,447	
	3	EF3C1	58	70	94	70	82	106	83	95	119	97	109	134	109	120	145		209		864	944	1,073	1,181	1,273	1,426	1
	3	EF3C2	46	63	86	58	75	98	71	88	111	86	103	125	97	114	137		209		876	957	1,089	1,181	1,280	1,434	
	3	EF3C3	52	61	82	64	73	94	77	86	107	91	100	121	103	112	133		209		884	966	1,100	1,195	1,287	1,442	
	4	EF3C1	49	60	84	64	75	99	80	92	116	98	110	134	112	124	148		196		864	944	1,073	1,172	1,263	1,416	1
	4	EF3C2	44	57	76	59	72	91	76	88	108	93	106	125	108	120	140		196		876	957	1,089	1,179	1,273	1,424	
	4	EF3C3	52	59	75	67	74	90	83	90	106	101	108	124	115	122	138		196		884	966	1,100	1,195	1,284	1,434	

Table 5.8 Continued

Activity	**Crew**	**Predecessor crews**	**Early Start (Days)**												**Early Finish (Days)**			**Own direct Cost (1,000$)**			**Cum. Pred. direct Cost (1,000 $)**			**Total Cost (1,000 $)**			**Local Optimal predecessor**
			Section 1			Section 2			Section 3			Section 4															
			a	b	c	a	b	c	a	b	c	a	b	c	a	b	c	a	b	c	a	b	c	a	b	c	
Slabs	1	E1 F3 C1 B1	-	-	-	87	99	124	100	115	142	111	128	156	125	144	175	165	183	195	1,087	1,167	1,296	1,377	1,495	1,666	4
	1	E1 F3 C1 B2	-	-	-	85	97	122	98	113	140	109	126	154	123	142	173	165	183	195	1,081	1,161	1,290	1,369	1,487	1,658	
	1	E1 F3 C1 B3	-	-	-	85	95	119	98	111	137	109	124	151	123	140	170	165	183	195	1,073	1,153	1,282	1,361	1,476	1,646	
	1	E1 F3 C1 B4	-	-	-	88	95	116	101	111	134	112	124	148	125	140	167	165	183	195	1,060	1,140	1,269	1,350	1,463	1,631	
	2	E1 F3 C1 B1	-	-	-	87	99	124	101	117	144	113	131	160	128	150	181	157	178	189	1,087	1,167	1,296	1,372	1,495	1,666	4
	2	E1 F3 C1 B2	-	-	-	85	97	122	99	115	142	111	129	158	126	148	179	157	178	189	1,081	1,161	1,290	1,364	1,487	1,658	
	2	E1 F3 C1 B3	-	-	-	83	95	119	97	113	139	109	127	155	124	146	176	157	178	189	1,073	1,153	1,282	1,354	1,476	1,646	
	2	E1 F3 C1 B4	-	-	-	86	92	116	100	110	135	112	124	152	127	143	173	157	178	189	1,060	1,140	1,269	1,343	1,461	1,630	

Table 5.9: Least Duration Optimization

Activity	Crew	Predecessor crews	Early Start (Days)												Early Finish (Days)			Own direct Cost (1,000$)			Cum. Pred. direct Cost (1,000 $)			Total Cost (1,000 $)			Local Optimal predecessor
			Section 1			Section 2			Section 3			Section 4															
			a	b	c	a	b	c	a	b	c	a	B	b	a	b	c	a	b	c	a	b	c	a	b	c	
Excavation	1	-	0	0	0	11	13	14	25	28	31	34	39	43	48	56	62	44	50	56	-	-	-	92	106	118	-
Foundation	1	E	20	22	26	29	33	38	40	45	51	49	56	62	58	66	72	542	569	580	44	50	56	644	685	708	1
	2	E	12	14	14	24	28	31	38	43	49	49	56	64	60	69	79	531	556	590	44	50	56	635	675	725	1
	3	E	11	13	14	26	32	38	42	52	63	56	69	85	70	86	106	500	535	577	44	50	56	614	671	739	1
Columns	1	EF1	29	33	38	46	51	61	60	66	79	78	89	106	95	106	128	312	359	439	586	619	636	993	1,084	1,203	2
	1	EF2	24	28	31	41	46	54	55	61	72	73	84	99	90	101	121	312	359	439	575	606	646	976	1,066	1,206	
	1	EF3	32	36	44	48	54	67	63	69	85	81	92	112	97	109	134	311	359	439	544	585	633	952	1,053	1,206	
	2	EF1	29	33	38	43	48	57	55	61	72	70	79	94	84	94	113	331	372	455	586	619	636	1,001	1,085	1,204	2
	2	EF2	25	28	31	38	43	50	50	56	65	65	74	88	79	89	106	331	372	455	575	606	646	985	1,067	1,207	
	2	EF3	31	42	51	44	57	70	56	69	85	71	88	107	85	103	125	331	372	455	544	585	633	961	1,059	1,214	
	3	EF1	29	33	38	41	46	54	51	57	67	64	73	86	76	85	102	340	381	467	586	619	636	1,002	1,086	1,204	2
	3	EF2	27	33	35	39	45	52	49	56	64	62	72	84	74	85	99	340	381	467	575	606	646	988	1,072	1,212	
	3	EF3	41	46	57	52	59	73	63	70	86	75	86	106	87	98	121	340	381	467	544	585	633	971	1,064	1,221	
Beams	1	EF2C1	55	66	81	64	74	90	73	84	99	83	94	109	91	102	117		223		886	965	1,085	1,201	1,290	1,426	3
	1	EF2C2	51	61	78	60	70	87	69	79	96	79	89	106	87	97	114		223		906	978	1,101	1,217	1,299	1,439	
	1	EF2C3	46	57	71	55	66	80	64	75	89	74	85	99	82	93	107		223		914	988	1,113	1,220	1,304	1,443	
	2	EF2C1	57	68	88	67	78	98	78	89	109	90	101	121	99	110	130		217		886	965	1,085	1,202	1,293	1,433	3
	2	EF2C2	46	56	73	56	66	83	67	77	94	79	89	106	88	98	115		217		906	978	1,101	1,212	1,294	1,434	
	2	EF2C3	41	52	66	51	62	76	62	73	87	74	85	99	83	94	108		217		914	988	1,113	1,215	1,299	1,438	
	3	EF2C1	51	62	82	63	74	94	76	87	107	90	101	121	102	113	133		209		886	965	1,085	1,196	1,287	1,427	3
	3	EF2C2	40	50	67	52	62	79	65	75	92	79	89	106	91	101	118		209		906	978	1,101	1,206	1,288	1,428	
	3	EF2C3	39	47	60	51	59	72	64	72	85	78	87	99	90	98	111		209		914	988	1,113	1,213	1,294	1,432	
	4	EF2C1	42	52	72	57	67	87	73	84	103	91	102	121	105	116	135		196		886	965	1,085	1,187	1,277	1,417	3
	4	EF2C2	38	48	56	53	63	71	69	79	88	87	97	106	101	111	120		196		906	978	1,101	1,204	1,286	1,417	
	4	EF2C3	39	45	52	54	60	67	70	76	84	88	94	102	102	108	116		196		914	988	1,113	1,213	1,292	1,424	

Table 5.9 Continued

Activity	Crew	Predecessor crews	Early Start (Days)												Early Finish (Days)			Own direct Cost (1,000$)			Cum. Pred. direct Cost (1,000 $)			Total Cost (1,000 $)			Local Optimal predecessor
			Section 1			Section 2			Section 3			Section 4															
			a	b	c	a	b	c	a	b	c	a	B	c	a	b	c	a	b	c	a	b	c	a	b	c	
Slabs	1	E1 F2 C3 B1	-	-	-	64	75	89	77	91	107	88	104	121	102	120	140	1,138	1,211	1,336	1,405	1,515	1,671	1,404	1,514	1,671	3
	1	E1 F2 C3 B2	-	-	-	62	73	87	75	89	105	86	102	119	100	118	138	1,132	1,205	1,330	1,397	1,507	1,663	1,397	1,506	1,663	
	1	E1 F2 C3 B3	-	-	-	66	72	85	79	88	103	90	101	117	104	118	136	1,123	1,196	1,322	1,392	1,497	1,653	1,392	1,497	1,653	
	1	E1 F2 C3 B4	-	-	-	75	79	84	88	95	101	99	108	116	113	124	134	1,111	1,184	1,309	1,388	1,491	1,639	1,388	1,491	1,639	
	2	E1 F2 C3 B1	-	-	-	64	75	89	78	93	109	90	107	125	105	126	146	1,138	1,211	1,336	1,399	1,515	1,671	1,399	1,514	1,671	3
	2	E1 F2 C3 B2	-	-	-	62	73	87	76	91	107	88	105	123	103	124	144	1,132	1,205	1,330	1,391	1,506	1,663	1,391	1,506	1,663	
	2	E1 F2 C3 B3	-	-	-	64	72	85	78	90	105	90	105	121	105	123	142	1,123	1,196	1,322	1,385	1,497	1,652	1,385	1,497	1,652	
	2	E1 F2 C3 B4	-	-	-	76	76	84	90	94	103	102	108	120	117	127	141	1,111	1,184	1,309	1,384	1,488	1,638	1,385	1,489	1,638	

5.2.2 **Least Duration Optimization**

This third run aimed at finding the least duration schedule while accounting for uncertainty. This run was planned to test the impact of changing the objective from least cost to least duration schedule. Same fuzzy variables from Table 5.7 were used as input for this run. As different crews were matched together and scheduled, optimum predecessors were identified using the least duration objective function (Equation 3.8) instead of the least cost objective function used in previous runs. As this run was accounting for uncertainty, the identified least project duration and the corresponding total project cost were in fuzzy numbers.

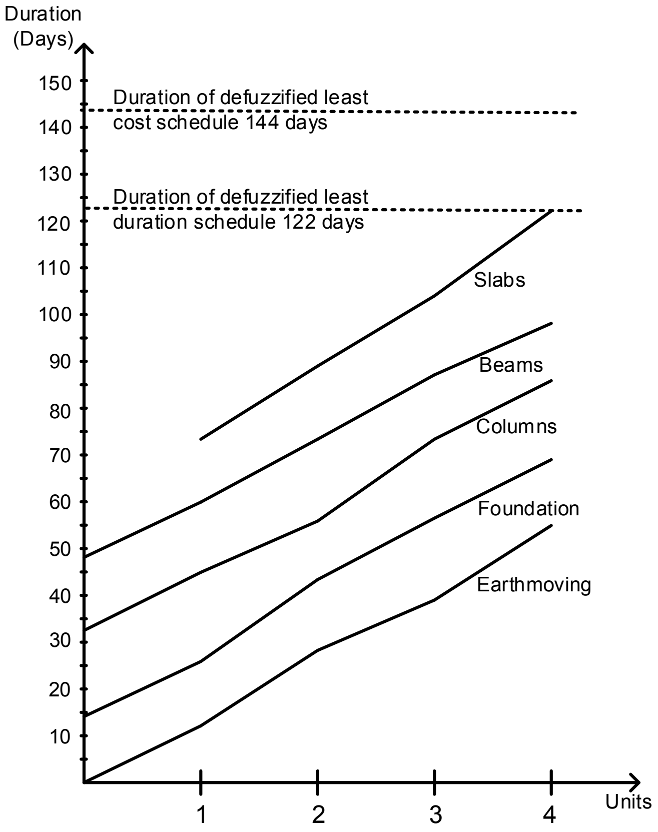


Figure 5.4: Defuzzified Least Duration Schedule

This run identified the optimum crew formation to be E1 F2 C3 B2 S1, generating a least duration schedule of (100, 118, 138) days and a total project cost of ($1,396,529.1, $1,506,358.8, $1,663,290.9). Run details are summarized in Table 5.9. The generated schedule was defuzzified, and the schedule was regenerated using Dur_{EV} for each duration. The defuzzified schedule is illustrated in Figure 5.4. To visualize the difference between different optimization objectives, the figure also shows the duration of the least cost schedule.

5.2.3 **Defuzzification and Buffering**

After the previous sections demonstrated the application of the optimized scheduling model, this section shows the application of the defuzzification and buffering functions of the developed method. The defuzzification and buffering was performed after the optimization procedure had been completed, producing the optimum crew formation and the fuzzy schedule. The objective of the defuzzification and buffering was to convert the fuzzy schedule into a deterministic one, and to size and insert time buffers to provide protection to the schedule against various anticipated delays. To utilize the same case study, the optimized fuzzy schedule produced in section 5.2.1.2 and summarized in Table 5.8 was used.

The defuzzification was performed first, where each fuzzy duration in the schedule was transformed into a deterministic duration (Dur_{EV}) using the expected value (EV) as per Equation 3.10. Then an agreement index (AI) of 0.9 was assumed for this project, reflecting the user desired confidence in the produced schedule. Accordingly for each duration a deterministic duration ($Dur_{AI0.9}$) was calculated using Equation 3.13. The difference between the two durations was the local buffer needed for each unit. Table 5.10 summarizes the calculated Dur_{EV} and $Dur_{AI0.9}$ for each unit. For a deterministic activity, such as the Beams in this case study, Dur_{EV} was the same as the original durations, and the buffer was equal to zero. After durations were calculated the schedule was regenerated. The first activity was scheduled continuously starting at day zero until all its units were completed. The remaining activities were scheduled on two stages, an initial scheduling, where each activity was scheduled continuously starting day zero. After that for each activity two shifts are calculated, the first shift to maintain work continuity and prevent clashing with previous activity, and the second shift to accommodate the intermediate buffers. The first shift is equal to the biggest negative difference between an activity's start at a unit and its predecessor's finish at the same unit as per Equation 3.18. To calculate the first shift of the Foundation activity, each units start date was compared to the same unit's end date of the Excavation activity. The biggest difference was found at the second unit (-12.4 days) as highlighted in Table 5.11. This was because the Foundation activity progresses at a slower rate compared to the Excavation activity, so the least duration between the two activities was after the first unit. Accordingly, the first shift was for the Foundation activity and was equal to (12.4 days). The second shift to accommodate the intermediate buffers was calculated at the unit where the biggest first shift was located, as this is the point of contact of the two activities and the point where the first impact of delays will be felt. The buffer inserted is equal to the summation of previous unit's buffers until the least duration between two successive activities as per 3.19. Accordingly for the Foundation activity the buffer was equal to only the local buffer of the first unit (0.8 days). Similarly for the Columns activity the least duration with its successor, the Foundation activity was at the third unit (36 days), and the previous activity's first three local buffers were aggregated resulting in a (7.2) days buffer. Same calculations are performed for the Beams activity. For the final activity, Slabs, as its predecessor activity was deterministic, there was no buffers and accordingly no second shift, only the first shift was calculated. After the end date of the Slabs activity, its local buffers are aggregated and added to its end date to get the total project end date. That final buffer insertion (4.2 days) provides protection to the total project duration against delays of the last activity. The calculated initial and final start and end dates and the aggregation and insertion of local buffers are summarized in Table 5.11. The local unit buffers of the last unit are aggregated and inserted after the finish date of the last unit of the last activity, to protect the project completion date from delays that could affect the last activity.

Table 5.10: DurEV and DurAI

Activity	Optimum Crew	Durations in Days											
		Unit 1			Unit 2			Unit 3			Unit 4		
		Dur_{EV}	$Dur_{AI=0.9}$	Buffer	Dur_{EV}	$Dur_{AI=0.9}$	Buffer	Dur_{EV}	$Dur_{AI=0.9}$	Buffer	Dur_{EV}	$Dur_{AI=0.9}$	Buffer
Exc.	E1	12.4	13.2	13.9	15.5	16.6	1.0	10.8	11.5	0.7	16.6	17.7	1.1
Found.	F3	19.5	21.9	2.4	20.3	22.9	2.5	17.8	20	2.2	16.9	19	2.1
Columns	C1	19.1	21	1.9	15.8	16.9	1.2	22.5	25	2.5	18.6	20.3	1.7
Beams	B4	15	15	0.0	16.3	16.3	0.0	17.8	17.8	0.0	14.1	14.1	0.0
Slabs	S2				17.3	18.7	1.5	14.3	15.5	1.2	18.1	19.7	1.5

Table 5.11: Deterministic Schedule with Buffers

Activity	Unit 1 (days)								Unit 2 (days)							
	Dur_{EV}	Buffer	Initial Start	Initial Finish	First Shift	Second Shift	Start	Finish	Dur_{EV}	Buffer	Initial Start	Initial Finish	First Shift	Second Shift	Start	Finish
Excavation	12.4	0.8	-	-	-	-	0.0	12.4	15.5	1.0	-	-	-	-	12.4	27.9
Foundation	19.5	2.4	0.0	19.5	-12.4	0.8	13.2	32.7	20.3	2.5	19.5	39.8	-8.5	-	32.7	53.1
Columns	19.1	1.9	0.0	19.1	-32.7	-	43.2	62.3	15.8	1.2	19.1	34.8	-34.0	-	62.3	78.1
Beams	15	0.0	0.0	15.0	-62.3	-	77.4	92.4	16.3	0.0	15.0	31.3	-63.0	-	92.4	108.6
Slabs	-	-	-	-	-	-	-	-	17.3	1.5	0.0	17.3	-108.6	-	109.2	126.5

Table 5.11: Deterministic Schedule with Buffers (continued)

Activity	Unit 3 (days)								Unit 4 (days)								
	Dur_{EV}	Buffer	Initial Start	Initial Finish	First Shift	Second Shift	Start	Finish	Dur_{EV}	Buffer	Initial Start	Initial Finish	First Shift	Second Shift	Start	Finish	End Date
Excavation	10.8	0.7	-	-	-	-	27.9	38.7	16.6	1.1	-	-	-	-	38.7	55.3	
Foundation	17.8	2.2	39.8	57.6	1.1	-	53.1	70.9	16.9	2.1	57.6	74.5	2.3	-	70.9	87.8	
Columns	22.5	2.5	34.8	57.3	-36.0	7.2	78.1	100.6	18.6	1.7	57.3	75.9	-30.4	-	100.6	119.2	
Beams	17.8	0.0	31.3	49.1	-69.3	-	108.6	126.5	14.1	0.0	49.1	63.3	-70.1	7.3	126.5	140.6	
Slabs	14.3	1.2	17.3	31.5	-109.2	0.0	126.5	140.7	18.1	1.5	31.5	49.6	-109.1	-	140.7	158.9	163.1

Finally the total project cost was recalculated for the regenerated deterministic schedule. Dur_{EV} was used to calculate the direct cost of each activity, and the new total project duration after adding buffers was used to calculate the project indirect cost. Table 5.12 summarizes the total project cost components, while Figure 5.5 shows the regenerated schedule with the inserted buffers.

Table 5.12: Total Project Cost

Component	**Duration (Days)**	**Indirect Cost ($)**	**Material ($)**	**Labour and Equipment ($)**
Project	163.1	163,072.0		
Excavation	55.3			961.3
Foundation	74.5		363,216.0	174,234.4
Columns	76.0		208,444.8	164,041.9
Beams	48.2		69,810.0	96,370.0
Slabs	49.6		73,842.0	100,628.3
Subtotal		163,071.94	715,312.8	536,235.8
		Total Project Cost		1,414,620.6

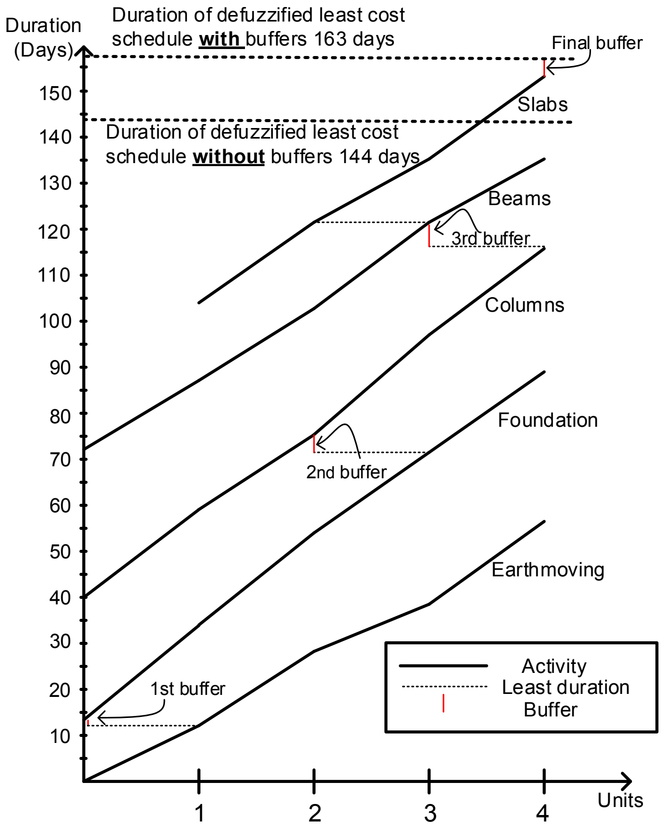


Figure 5.5: Defuzzified Least Cost Schedule with Buffers

5.3 Schedule Acceleration

5.3.1 Case Study I

This section explains the first case study drawn from literature and analyzed to evaluate the performance of the developed Acceleration Model and demonstrate its basic features. The case study, which was originally presented in El-Rayes (1997) and analyzed later by Hassanein and Moselhi (2005), consists of a 15 Km three-lane highway repetitive project, each Km consists of 5 sequential activities. These activities, in their order of precedence, are: (1) Cut and Chip Trees; (2) Grub and Remove Stumps; (3) Earthmoving; (4) Base; and (5) Paving. All precedence relations are finish to start, with no lag time. Each activity is divided into 15 segments of equal lengths, each is 1km long. This project includes typical and non-typical activities, as activities quantities vary from one segment to another. It also includes sequential and non-sequential activities, pas the Earthmoving activity starts at unit 4, then proceeds backwards till unit 1, then resumes again at unit 5 till unit 15. Available crews for each activity and their productivities are shown in Table 5.13, activities quantities and units sequence are shown in Table 5.14, while the base line schedule with activities direct costs is shown in Table 5.15.

Table 5.13: Crews Productivity

Activity	**Crew**	**Productivity (units/day)**
Cut and Chip Trees	1	3,000
	2	3,000
	3	3,000
Grub and Remove Stumps	1	4,000
	2	4,000
Earthwork	1	1,200
	2	800
Base	1	3,200
	2	3,200
	3	3,200
	4	3,200
Pave	1	4,000
	2	4,000
	3	4,000

Project's indirect cost is 4,000 $/day. Thus the total project cost mounts up to $1,878,300. The goal was to find the least cost acceleration plan. This was performed through the correct identification of units to accelerate and to prioritize them according to the relevant queuing criteria. For simplicity, only one acceleration strategy was considered in this example, which was adding overtime hours, with a maximum of 4 hours per crew per day. In the case that multiple acceleration strategies are available for an activity, each strategy’s cost slope and contractor’s judgment rankings are calculated separately and the one with the highest joint priority is chosen. Overtime cost is 300$/Hr for “cut and chip trees” crews, 600$/Hr for “grub and remove stumps” crews, 700 $/Hr for “excavation” crews, 400$/Hr for “base” crews, and finally 450$/Hr for “paving” crews. The initial schedule had a normal duration of 83 days. The baseline schedule is shown in Figure 5.6.

Table 5.14: Activities Quantities and Units Sequence

Activity	**Cut and Chip Trees**		**Grub and Remove Stumps**		**Excavation**		**Base**		**Paving**	
Unit	Quantity (m^2)	Sequence	Quantity (m^2)	Sequence	Quantity (m^3)	Sequence	Quantity (m^2)	Sequence	Quantity (m^2)	Sequence
1	12,000	1	12,000	1	6,000	4	3,200	1	3,200	1
2	12,000	2	12,000	2	6,000	3	3,200	2	3,200	2
3	18,000	3	18,000	3	6,000	2	3,200	3	3,200	3
4	12,000	4	12,000	4	7,000	1	3,200	4	3,200	4
5	18,000	5	18,000	5	8,600	5	3,200	5	3,200	5
6	30,000	6	30,000	6	7,000	6	3,200	6	3,200	6
7	36,000	7	36,000	7	6,500	7	3,200	7	3,200	7
8	30,000	8	30,000	8	6,000	8	3,200	8	3,200	8
9	24,000	9	24,000	9	6,000	9	3,200	9	3,200	9
10	24,000	10	24,000	10	6,000	10	3,200	10	3,200	10
11	18,000	11	18,000	11	6,000	11	3,200	11	3,200	11
12	12,000	12	12,000	12	6,000	12	3,200	12	3,200	12
13	12,000	13	12,000	13	6,000	13	3,200	13	3,200	13
14	12,000	14	12,000	14	6,000	14	3,200	14	3,200	14
15	12,000	15	12,000	15	6,000	15	3,200	15	3,200	15

Table 5.15: Baseline Schedule

Cut & Chip Trees						Grub Stumps					Earthmoving				
Unit	**Duration**	**Direct Cost**	**Crew #**	**Start**	**End**	**Duration**	**Direct Cost**	**Crew #**	**Start**	**End**	**Duration**	**Direct Cost**	**Crew #**	**Start**	**End**
1	4	7,800	1	0	4	3	12,000	1	4	7	8	32,000	2	20	28
2	4	7,800	2	0	4	3	12,000	2	4	7	5	20,000	1	17	22
3	6	11,700	3	0	6	5	20,000	1	7	12	8	32,000	2	12	20
4	4	7,800	1	4	8	3	12,000	2	8	11	6	24,000	1	11	17
5	6	11,700	2	4	10	5	20,000	2	11	16	8	32,000	1	22	30
6	10	19,500	3	6	16	8	32,000	1	16	24	9	36,000	2	28	37
7	12	23,400	1	8	20	9	36,000	2	20	29	6	24,000	1	30	36
8	10	19,500	2	10	20	8	32,000	1	24	32	5	20,000	1	36	41
9	8	15,600	1	20	28	6	24,000	2	29	35	8	32,000	2	37	45
10	8	15,600	2	20	28	6	24,000	1	32	38	5	20,000	1	41	46
11	6	11,700	4	24	30	5	20,000	2	35	40	8	32,000	2	45	53
12	4	7,800	1	28	32	3	12,000	1	38	41	5	20,000	1	46	51
13	4	7,800	2	28	32	3	12,000	2	40	43	5	20,000	1	51	56
14	4	7,800	4	30	34	3	12,000	1	41	44	8	32,000	2	53	61
15	4	7,800	1	32	36	3	12,000	2	43	46	5	20,000	1	56	61

Table 5.15: Baseline Schedule (Continued)

Base						Pave				
Unit	**Duration**	**Direct Cost**	**Crew #**	**Start**	**End**	**Duration**	**Direct Cost**	**Crew #**	**Start**	**End**
1	10	1	25000.0	28	38	8	20000.0	1	38	46
2	10	2	25000.0	26	36	8	20000.0	2	36	44
3	10	3	25000.0	33	43	8	20000.0	3	43	51
4	10	4	25000.0	31	41	8	20000.0	1	46	54
5	10	1	25000.0	38	48	8	20000.0	2	48	56
6	10	2	25000.0	36	46	8	20000.0	3	51	59
7	10	3	25000.0	43	53	8	20000.0	1	54	62
8	10	4	25000.0	41	51	8	20000.0	2	56	64
9	10	1	25000.0	48	58	8	20000.0	3	59	67
10	10	2	25000.0	46	56	8	20000.0	1	62	70
11	10	3	25000.0	53	63	8	20000.0	2	64	72
12	10	4	25000.0	51	61	8	20000.0	3	67	75
13	10	1	25000.0	58	68	8	20000.0	1	70	78
14	10	2	25000.0	56	66	8	20000.0	2	72	80
15	10	4	25000.0	61	71	8	20000.0	3	75	83

Applying the developed model to the case study aimed at testing the model's ability to identify activities to accelerate, and at evaluating the impact of the different queuing criteria on the generated acceleration plan. For that purpose three different scenarios were designed and analyzed. These scenarios represented different use cases a contractor can have during a project. These scenarios are described in the following subsections.

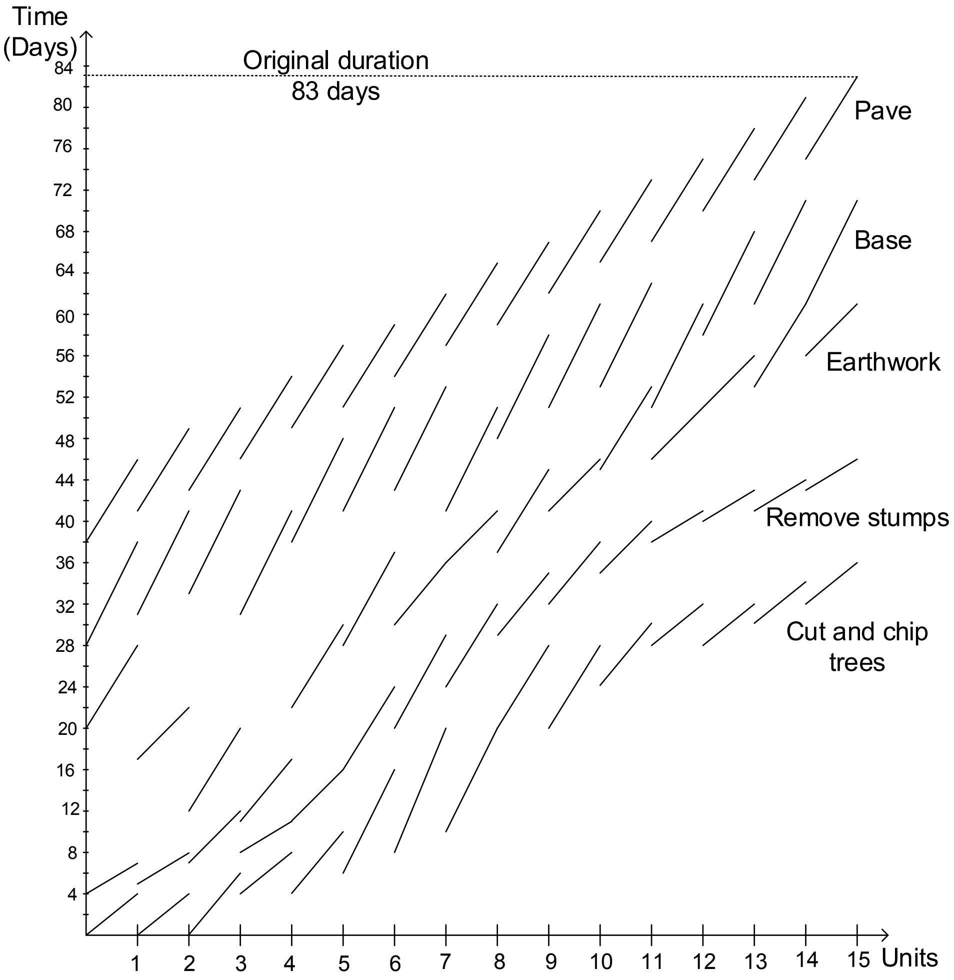


Figure 5.6: Original Schedule (Case Study I)

5.3.1.1 First Scenario

The first was the base case scenario, which aimed at testing the ability of the model to identify activities to accelerate, and to prioritize them to find the least cost acceleration plan. To serve this purpose, uncertainties associated with acceleration costs were neglected, and so was the contractor's judgment criteria. Deterministic acceleration costs were used to set the cost slope. Table 5.16 shows the cost of one overtime hour for each activity. According to each unit's quantity and its relevant crew's productivity, the number of hours needed to crash its duration by one day was calculated. The cost of those calculated hours was the cost slope for that activity at that unit. In this scenario, the acceleration cost of each unit was taken as a crisp number. The unit with the least cost slope was given the highest priority of 1.0, and the unit with the largest cost slope was given a priority of 5.0. The rest of the units were given a ranking between 1.0 and 5.0; proportional to the values of their respective cost slope. As contractor's judgment was not considered in this scenario, cost slope was the only ranking criteria for activities. The baseline schedule was input to the spread sheet application, which automatically calculated the value of Ω for each unit of each activity as per Equation 3.27. Units with positive value of Ω were considered for acceleration.

Table 5.16: Overtime Cost for Case Study I

Activity	Overtime Cost ($/hour)
Cut and Chip Trees	300
Grub and Remove Stumps	600
Earthmoving	700
Base	400
Pave	450

In an iterative manner, overtime hours were assigned to the unit with the highest priority (least cost slope). After each unit was accelerated, the spread sheet automatically updated schedule durations, values of Ω and total project cost accordingly, and recorded the new duration and its corresponding total project cost. After drawing the set of possible acceleration plans (Figure 5.12), the developed method located the least project total cost to be $1,878,000, and the matching duration was 79 days.

5.3.1.2 Second Scenario

After the base case scenario was analyzed, this second scenario aimed at incorporating the contractor's judgment as an additional criteria and assess its impact on the generated acceleration plan. Similarly, the aim was to locate the least costly acceleration plan. Additional needed input was the contractor's judgment ranking for each activity, and the weights of the cost slope and contractor's judgment criteria. The cost was set to be more important than contractor's judgment. These criteria's relative importance, expressed by their relative weights, was set to 0.6 and 0.4, respectively. The cost slope ranking was based on deterministic costs same as in the previous scenario and contractor's judgment was added. For simplicity, each activity was given the same contractor's judgment rank throughout all of its units. Contractor's judgment was set to rank 3.0 for activity Cut and Chip, set to 1.0 for the activity Grub Stumps, set to 5.0 for the activity Earthmoving, set to 2.0 for the activity Base, and finally was set to 4.0 for the activity Pave. After setting the weights and rankings the model started by calculating Ω and the joint priority for each unit. Acceleration resources were then assigned incrementally until the least cost acceleration plan was identified. During accelerating the project values of Ω continuously changed as relative alignment of successive units changed, however the joint priority remained the same. In case two units had the same joint priority, the unit with a higher Ω value was accelerated first. In case both had the same value for Ω, priority was given to the unit located later in the schedule. The method found the least project total cost to be \$1,892,600, and the matching duration is 79 days. (Figure 5.12).

5.3.1.3 Third Scenario

The third scenario aimed at assessing the impact of modelling uncertainty in the acceleration cost. The same case study was run, and similar to the previous scenario, the relative weights of cost slope and contractor's judgment were set to 0.6 and 0.4, respectively. The only change in this scenario was that additional acceleration costs were input as triangular fuzzy numbers to model uncertainty. Table 5.17 shows the acceleration cost for each activity modeled using triangular fuzzy numbers. The fuzzy additional cost was utilized to calculate the EV for the additional cost (Equation 3.36). The cost slope priority and contractor's judgment priority were assigned and the joint priority was calculated. Acceleration resources (over-time hours) were added iteratively to units with positive values for Ω. Finally the least total project cost was found to be (\$1,884,360, \$1,886,405, \$1,889,180) and the corresponding duration was 79 days. The deterministic total cost was calculated using the EV of the additional cost of each activity, which is the same EV equation used to calculate the activities cost slope. It was found to be \$1,890,648. The project's total cost was plotted in Figure 5.12.

Table 5.17: Fuzzy Overtime Cost for Case Study I

Activity	Overtime Cost ($/hour)		
	a	b	C
Cut and Chip Trees	275	300	350
Grub and Remove Stumps	560	595	640
Earthmoving	650	697.5	750
Base	370	400	440
Pave	400	455	500

5.3.2 **Case Study II**

The repetitive project drawn from literature and analyzed for the optimized scheduling model, is utilized again with the Acceleration Model. The baseline schedule used was the least cost optimized schedule, listed in Table 5.5, without the inserted buffers, having a total duration of 143 days and a total cost of $1,462, 137. Figure 5.7 shows the original schedule before acceleration. Similar to the procedure followed with the previous acceleration case study, a single acceleration strategy was made available, that is working overtime hours. The objective was also to assess the model's ability to identify activities to accelerate, and evaluate the impact of accounting for different queuing criteria on the produced acceleration plans. Three scenarios were run. The first scenario aimed at finding the least cost acceleration plan while accounting for deterministic acceleration costs. The second scenarios aimed at evaluating the impact of including contractor's judgment as an additional queuing criteria. Finally, the third scenario aimed at accounting for uncertainty in accelerations costs while producing the least cost acceleration plan.

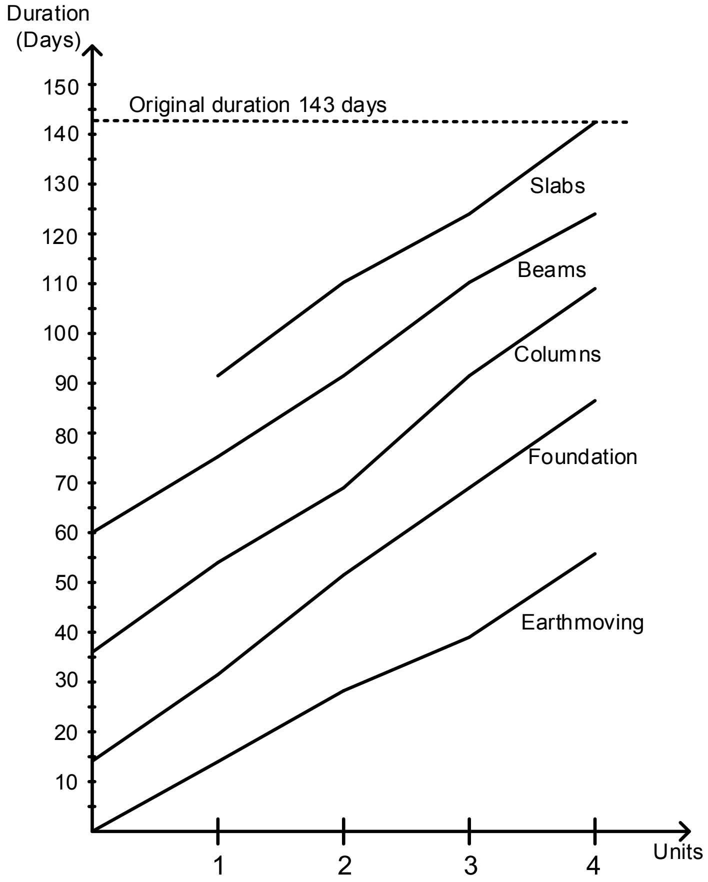


Figure 5.7: Original Schedule before Acceleration

5.3.2.1 First Scenario

This scenario aimed at finding the least cost acceleration plan without accounting for contractor's judgment or uncertainties in acceleration costs. The baseline schedule was input to the spreadsheet application. Then acceleration costs were input. As acceleration costs were not part of the original project data, they had to be assumed. The equipment and labour cost data for each activity which was provided initially in (\$/day) was divided by the number of working hours per day to get the cost in (\$/hour), this obtained number considered to be the regular hourly cost, was multiplied by a 2.5 factor, as overtime hours are more expensive than base hours. The final deterministic acceleration cost for each activity used for this scenario is summarized in Table 5.18.

Table 5.18: Overtime Cost for Case Study II

Activity	**Overtime Cost ($/Hr)**
Excavation	283.13
Foundation	730.63
Columns	675.00
Beams	624.38
Slabs	633.48

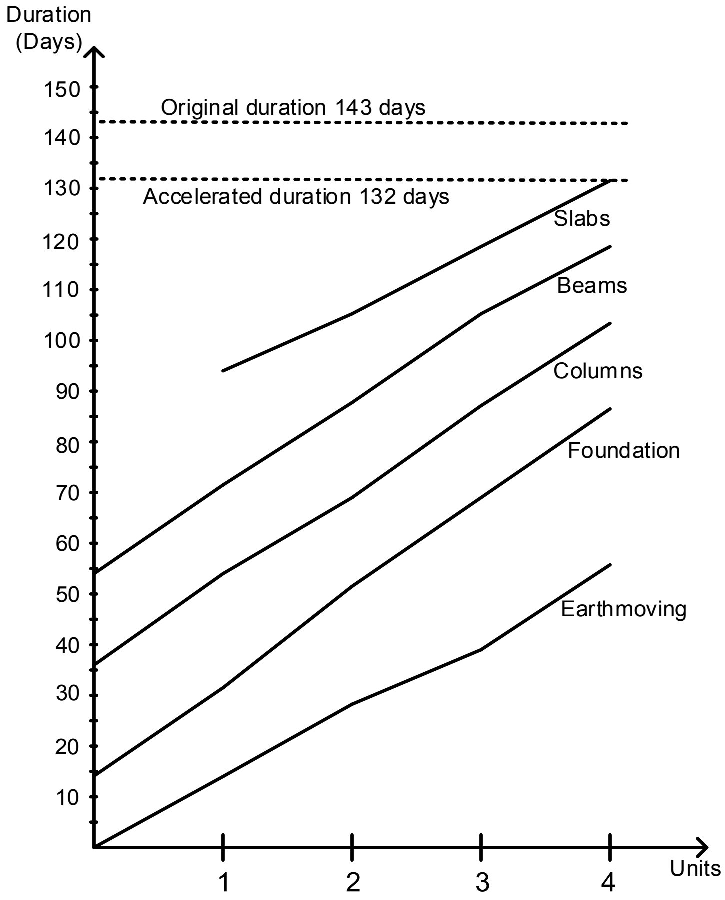


Figure 5.8: Scenario 1 Accelerated Schedule

The spread sheet identified the least aligned units through calculating **Ω**, and prioritized those units using cost slope ranking, based on the cost of the total number of overtime hours required to reduce the duration by 1 day. Based on this prioritization, overtime hours were assigned in an iterative manner, and the total project duration and cost were calculated and tabulated. The least cost acceleration plan was found to have a total cost of $1,458,639 and a corresponding duration of 132 days. Figure 5.8 shows the accelerated schedule.

5.3.2.2 Second Scenario

The second scenario aimed at identifying the least cost acceleration plan while accounting for contractor's judgment as an additional queuing criteria. Same data for the previous scenario was utilized. Contractor's judgment was set to be more important than cost slope, and consequently weights were assigned as 0.6 and 0.4 for contractor's judgment and cost slope respectively. Values assigned for each activity's contractors' judgment are listed in Table 5.19. The joint ranking based on cost slope and contractor's judgment was calculated and activities were queued for acceleration accordingly. The least cost acceleration plan was found to have a total project cost of $1,459,029 and a corresponding duration of 138 days. Figure 5.9 shows the accelerated schedule.

Table 5.19: Assigned Contractor's Judgment Values for Case Study II

Activity	Contractor's Judgment Ranking
Excavation	5
Foundation	4
Columns	2
Beams	1
Slabs	3

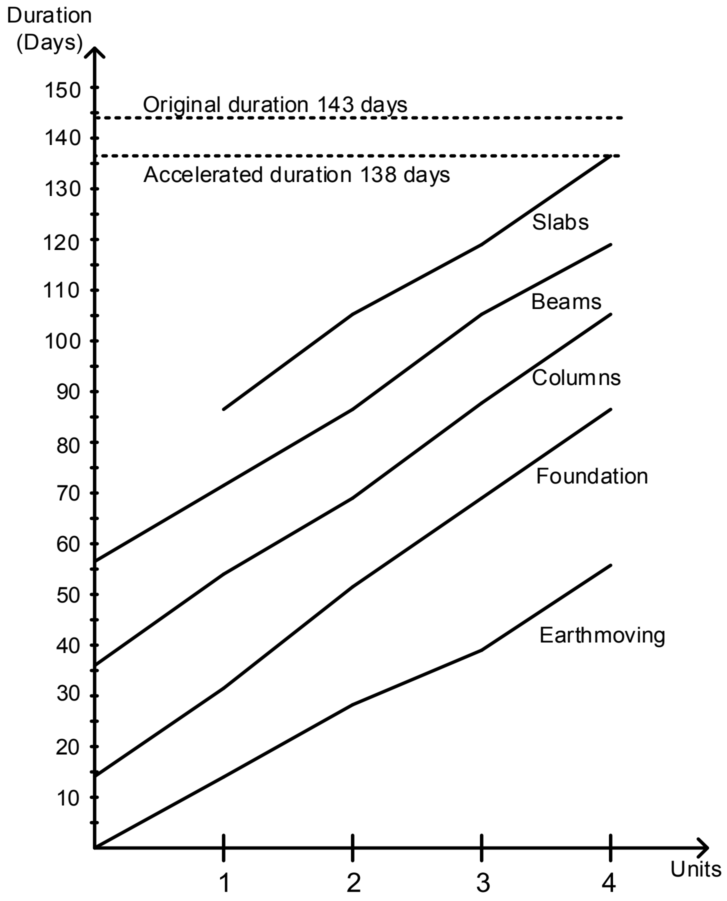


Figure 5.9: Scenario 2 Accelerated Schedule

5.3.2.3 Third Scenario

The third and final scenario was run to evaluate the model's ability to account for uncertainty in acceleration cost and the impact of such uncertainty on the generated acceleration plan. Accordingly before the run, acceleration cost had to be changed into triangular fuzzy numbers. Each activity's acceleration cost was multiplied once by a factor less than 1.0 and once by a factor greater than 1.0 to get the "a" and "c" values for the triangular fuzzy number. Those factors were randomly chosen as listed in Table 5.20. When those factors were applied to the cost of overtime hours, the resulting fuzzy cost of overtime hours were obtained, as listed in Table 5.21.

Table 5.20: Factors for Fuzzifying Acceleration Cost for Case Study II

Activity	Factor		
	a	b	c
Excavation	0.85	1.00	1.20
Foundation	1.00	1.00	1.00
Columns	0.90	1.00	1.20
Beams	0.80	1.00	1.10
Slabs	0.95	1.00	1.25

Table 5.21: Fuzzy Overtime Cost for Case Study II

Activity	Acceleration Cost ($/hour)		
	a	b	c
Excavation	240.7	283.1	339.8
Foundation	730.6	730.6	730.6
Columns	607.5	675.0	810.0
Beams	499.5	624.4	686.8
Slabs	601.8	633.4	791.8

The fuzzy acceleration cost was input to the spreadsheet, which calculated the EV of each activities acceleration cost, and queued activities for acceleration. Weights for contractor's judgment and cost slope criteria were kept the same as the previous scenario before running the acceleration procedure. The least cost acceleration plan was found to have a total cost of $1,459,074 and a corresponding duration of 138 days. The accelerated schedule is shown in Figure 5.10.

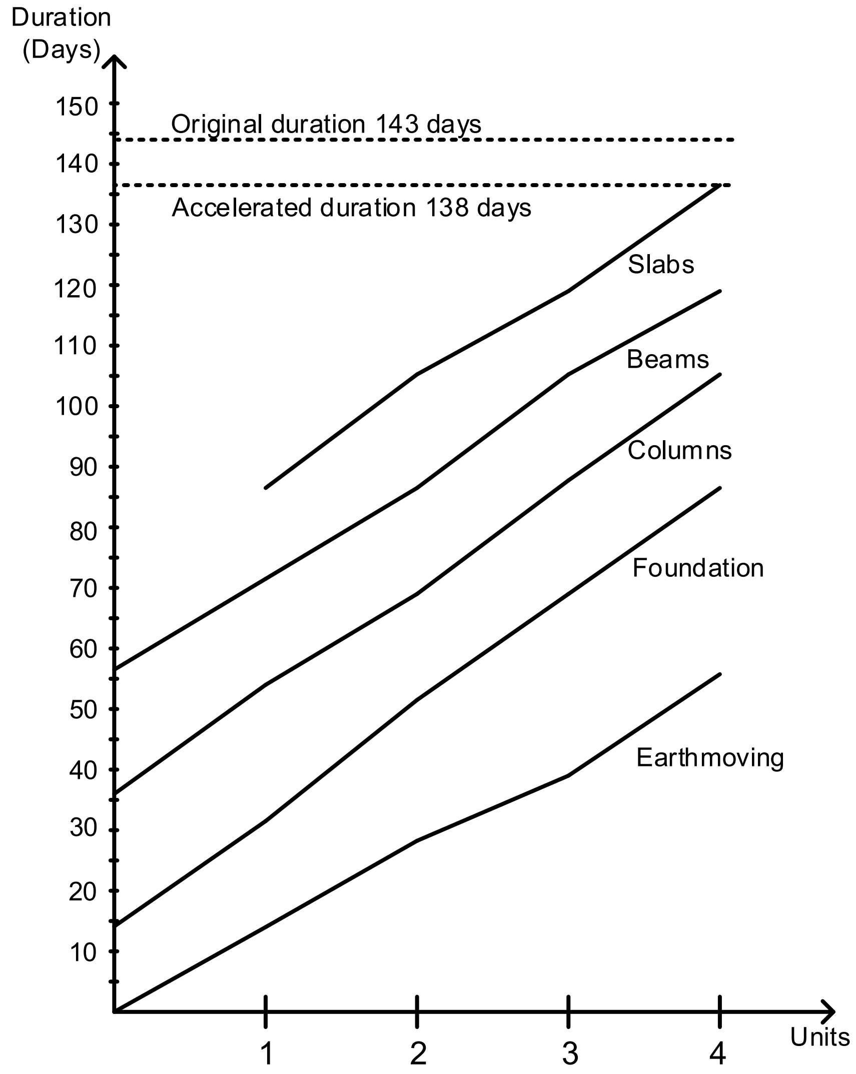


Figure 5.10: Scenario 3 Accelerated Schedule

5.4 Results Analysis and Findings

5.4.1 Optimized Scheduling and Buffering

The presented optimization and buffering model was successfully used to optimize the schedule in the described case study. Although many researchers utilized the same case study for testing different repetitive scheduling techniques, this case study was not used before with any technique accounting for uncertainty in any of the input parameters. The results generated by the presented model were compared to those of El-Rayes (1997). Although his work was addressing deterministic optimization, he similarly to this method adopted dynamic programming as an optimization tool. Different runs were made for the developed model. Initially the model was utilized to find the least cost schedule with deterministic numbers mapped into triangular fuzzy numbers. At the end of the run, optimum project cost was found to be $1,460,203 and the corresponding duration was 143 days. The original case analyzed by El-Rayes produced a least cost of $1,458,799 in a duration of 143 days. The difference between the costs of the two results is less than 1%, and the durations were an exact match. Also the crew formation yielding the least cost schedule was identified to be E1 F3 C1 B4 S2, which is the same as that identified by El-Rayes (1997).

The second run was to test the impact of accounting for uncertainty while finding the least possible cost for the project. After the run was completed the generated schedule had a fuzzy total cost of ($1,338,562.9, $1,460,231.0, $1,630,342.8). This total cost has an EV of $1,476,378.9, and the corresponding fuzzy duration was (121.7, 142.9, 172.5), which has an EV of 146 days. The crew formation yielding this result was E1 F3 C1 B4 S2. Later this optimized schedule was used for the defuzzification and buffering process. The defuzzified scheduled had a total duration of 163.1 days after inserting buffers, and the total project cost was $1,414,620.6. The final run optimized the schedule with a different objective, which was to find the least possible duration, also while accounting for various uncertainties. The least possible fuzzy duration turned out to be (100, 118, 138) days, which has an EV of 119 days. The corresponding cost was ($1,396,529.1,

$1,506,358.8, $1,663,290.9), with an EV of $1,522,060 and the utilized corresponding crew formation was E1 F2 C3 B2 S1.

Table 5.22 summarizes the output of the three runs. The first run generated the same results as the original optimized deterministic schedule by El-Rayes (1997). The exact same duration and crew formation were generated, and the difference in the identified least cost is less than 1%. This is attributed to different approximations in durations and costs along the optimization procedure. These results verifies the model's performance and confirms that all utilized equations are properly working. When plotted, the output of this run and the original schedule almost completely overlapped. The second run that accounted for uncertainties in various input parameters generated a slightly higher total cost and a slightly longer duration and utilizing a different crew formation. The modeled uncertainties changed the duration and cost of each crew, resulting in different local optimum predecessors, and eventually resulting in slightly different total project cost and duration.

When the results of that last run were defuzzified and buffers were sized and inserted, the resulting deterministic schedule cost and duration were $1,414,620.6 and 163.1 days respectively. The increase in the duration was due to the insertion of buffers which prolonged the project duration. As buffers are calculated based on an AI of 0.9 in this case study, then the individual duration of each activity and its local buffer should always be less than the biggest value in that activity's fuzzy duration. The same applies to the total project duration. Hence it is logical to find the total duration of the defuzzified project after inserting buffers less than the biggest number in the fuzzy total project duration (163.1 days and 172.5 days respectively). The reduction in the total project cost is due to the reduction in the direct costs of activities, as the EV of each activity is used for calculating direct costs, while the added buffers only increase the indirect cost of the project.

The last run aimed at finding the least duration schedule. The resulting duration was 119 days, which is less than other runs total durations. Also the corresponding cost was $1,522,060, which is higher than other runs costs. For the identified optimum crew formation, a different crew was selected for each activity in comparison to other runs, accept for the Excavation activity which had only one crew available. The selected crews were the ones having higher productivity and higher cost. The generated results are consistent as they produced a considerably shorter total project duration at a higher cost. Figure 5.11 shows the relative results of different runs.

Table 5.22: Optimization Results Summary

Case	**Optimization Objective**	**Fuzzy Input**	**Duration**				**Cost (1,000$)**				**Crew Formation**
Original schedule	Least cost	No	143				1,459				E1 F3 C1 B4 S2
Verification run	Least cost	No	143				1,460				E1 F3 C1 B4 S2
			a	b	c	EV	a	b	c	EV	
Run with fuzzy input (least cost)	Least cost	Yes	122	143	173	146	1,338.6	1,460.2	1,630.3	1,476.4	E1 F3 C1 B4 S2
Run with fuzzy input (least duration)	Least duration	Yes	100	118	138	119	1,396.5	1,506.4	1,663.3	1,522.1	E1 F2 C3 B2 S1
After defuzzification and buffering			163.1				1,415				

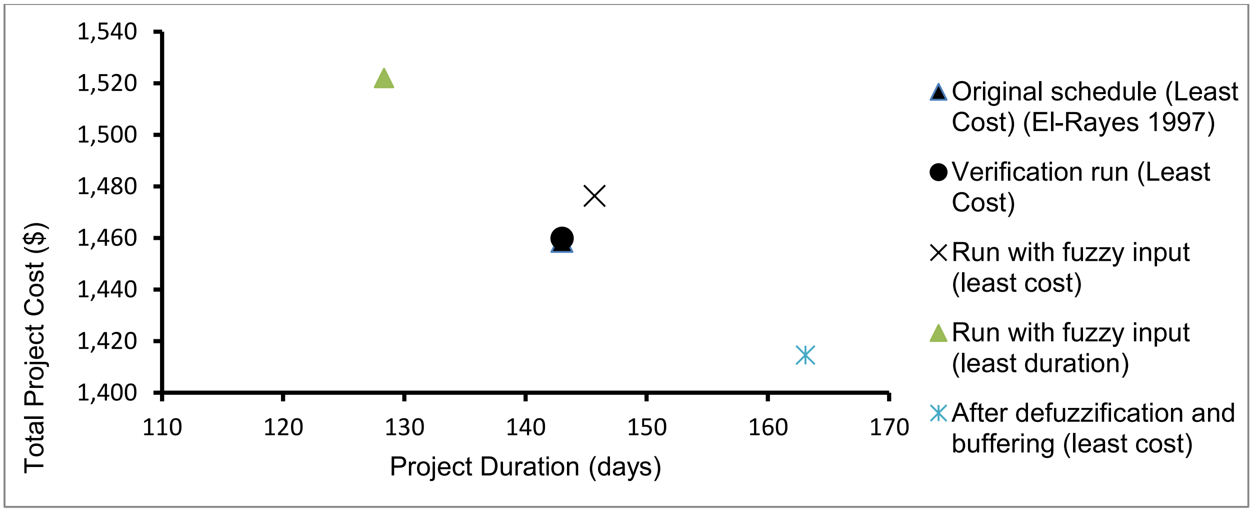


Figure 5.11: Optimization Results Summary

5.4.2 **Schedule Acceleration**

The developed Acceleration Model was run to analyze two different case studies, each comprising three different scenarios. Each scenario aimed at evaluating the impact of a different feature of the developed model. In the first case study, the model was first run to perform traditional acceleration, it didn't account for uncertainty in the acceleration cost or contractor's judgment. The aim was to test the model's ability to identify activities to accelerate. The least cost acceleration plan was found to have a cost of $1,878,000 and a corresponding duration of 79 days. Although the reduction in duration was small in comparison to the original schedule, but this is justified by the small size of the project. The second run incorporated contractor' judgment as an additional criteria for prioritizing activities for acceleration. The identified least cost was more than the one identified in the first scenario, this is due to incorporating an additional criteria that shifted the selection of activities to accelerate away from the activities with the least cost slope.

Accounting for such a criteria resulted in a least cost acceleration plan having a duration of 78 days and a corresponding cost of $1,892,600. The third and final scenario accounted for uncertainty in the acceleration cost, in addition to the contractor's judgment criteria. The least cost acceleration plan was identified to cost $1,890,648 and to have a duration of 79 days. The contractor's judgment values were utilized the same as the previous scenario and the fuzzy additional cost was designed to have an EV almost the same as the deterministic cost of the previous scenario, the results came almost similar with a little deviation. Table 5.23 summarizes the results of the run scenarios, while Figure 5.12 shows how the least cost was identified for each of the scenarios.

Table 5.23: Results of Different Acceleration Scenarios (for Case Study I)

Case	**Uncertainty in Acceleration Cost**	**Contractor Judgment**	**Cost ($)**	**Duration (days)**
Original Schedule	-	-	1,878,300	83
First Scenario	No	No	1,878,000	79
Second Scenario	No	Yes	1,892,600	78
Third Scenario	Yes	Yes	1,890,648	79

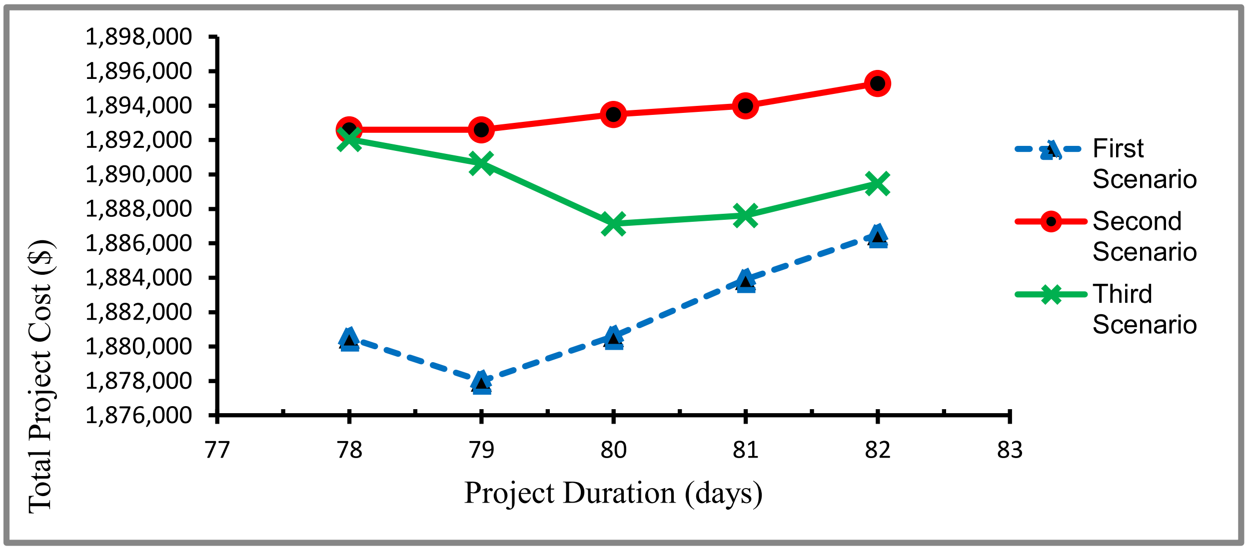


Figure 5.12: Locating Least Cost Plan for Different Scenarios (for Case Study I)

Observing the listed outcome of the generated acceleration plans, it was noticed that all plans costs are higher than the initial project schedule. Also the total number of assigned overtime hours was bigger than expected. Reviewing runs details revealed the reason. The studied project involved many crews for each activity, with a total of 15 crews working on the 5 project activities (as illustrated in Figure 5.6). When overtime hours are assigned for a crew working on a number of units, those units are completed in shorter durations, but this might not always be reflected on that activity's finish time, as there are other crews working on other units that have not been accelerated yet. To reduce the activities total duration, many overtime hours had to be assigned to accelerate the needed durations for all involved crews. This consequently increased the added direct cost, overcoming the corresponding reduction in indirect cost. This observation drew the attention to the fact that this model manages to calculate the additional resources needed (overtime-hours in this case) to decrease the duration needed to complete work on a unit by 1 unit of time, which does not necessarily translate to a 1 unit of time reduction in the duration of the activity and hence the project.

To further evaluate the performance of the model, the second case study was analyzed. The second case study (utilized for testing the optimization model) is also a repetitive project, but after the optimization procedure has been completed, the project was performed by a single crew for each activity, which is the optimum crew. Accordingly any 1 unit of time reduction in any unit's duration, would result in a 1 unit of time reduction in the activity's duration. And if that activity had a positive value for **Ω**, then it would be less aligned with its successor, thus is more likely to lead to a 1 unit of time reduction in the total project duration. Similarly three case studies have been analyzed, each scenario serving a different purpose as the first case study. In the three scenarios, assigning acceleration resources continued until no further reduction in duration was achievable. When plotted, each of the produced time cost relations of the three scenarios had a shape closer to the traditional u-shape curve characterizing project acceleration. Where a reduction in cost and duration was recorded as overtime hours were iteratively assigned, until the least cost had been recorded. Further duration reductions were possible but were associated with an increase in total project cost.

The first scenario that accounted for neither contractor's judgment nor uncertainty in acceleration cost recorded the least total project cost of $1,458,639 at a duration of 132 days. In the second scenario, including contractor's judgment as an additional queuing criteria changed the least cost slope prioritization followed during the first scenario, hence resulted in a higher least project cost of $1,459,029 at a 138 days duration. For the third and final scenario which accounted for uncertainties in acceleration cost the results came almost identical to the second scenario. This is because in both scenario contractor's judgment criteria had a weight of 0.6 against a weight of 0.4 for cost slope. So accounting for uncertainties in the acceleration cost did not result in a significant change in activities prioritization. Using EV of each activity's acceleration cost to calculate the project's total cost had a very little deviation from the second scenario. The third scenario's least cost acceleration plan had a total cost of $1,459,074 and a corresponding duration of 138 days. Scenarios results are summarized in Table 5.24 and plotted in Figure 5.13. Figure 5.14 shows the least cost acceleration plan's duration and corresponding cost for the three scenarios and the original schedule's duration and cost.

Table 5.24: Results of Different Acceleration Scenarios (for Case Study II)

Case	Uncertainty in Acceleration Cost	Contractor Judgment	Cost ($)	Duration (days)
Original Schedule	-	-	1,462,137	143
First Scenario	No	No	1,458,639	132
Second Scenario	No	Yes	1,459,029	138
Third Scenario	Yes	Yes	1,459,074	138

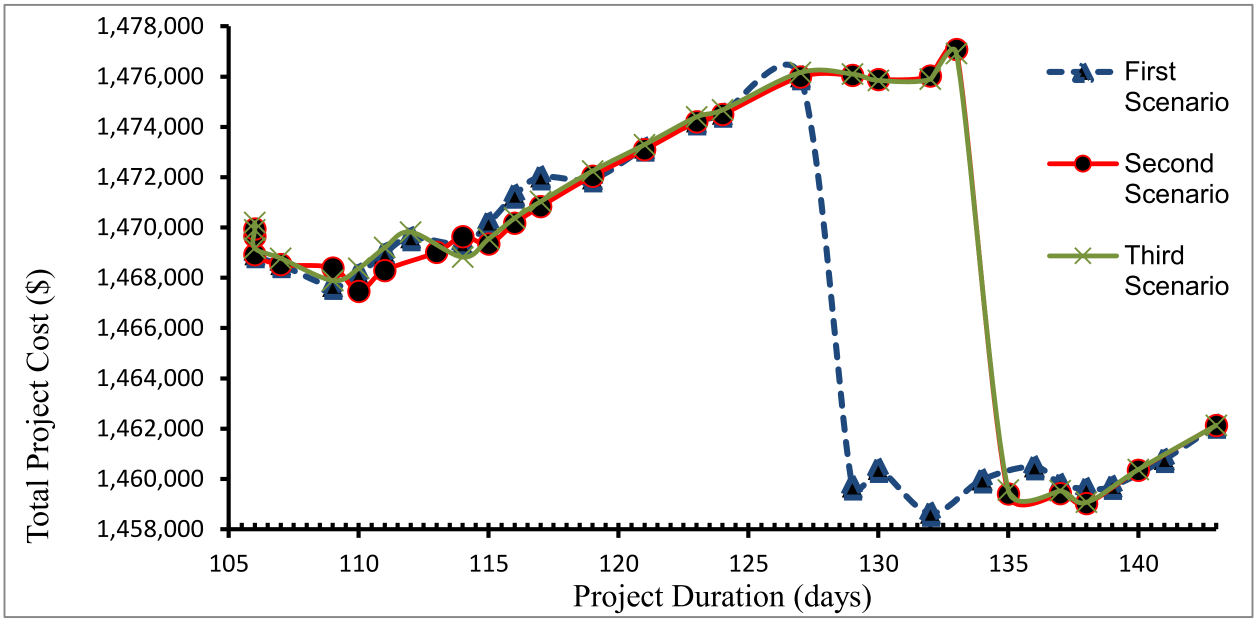


Figure 5.13: Locating Least Cost Plan for Different Scenarios (for Case Study II)

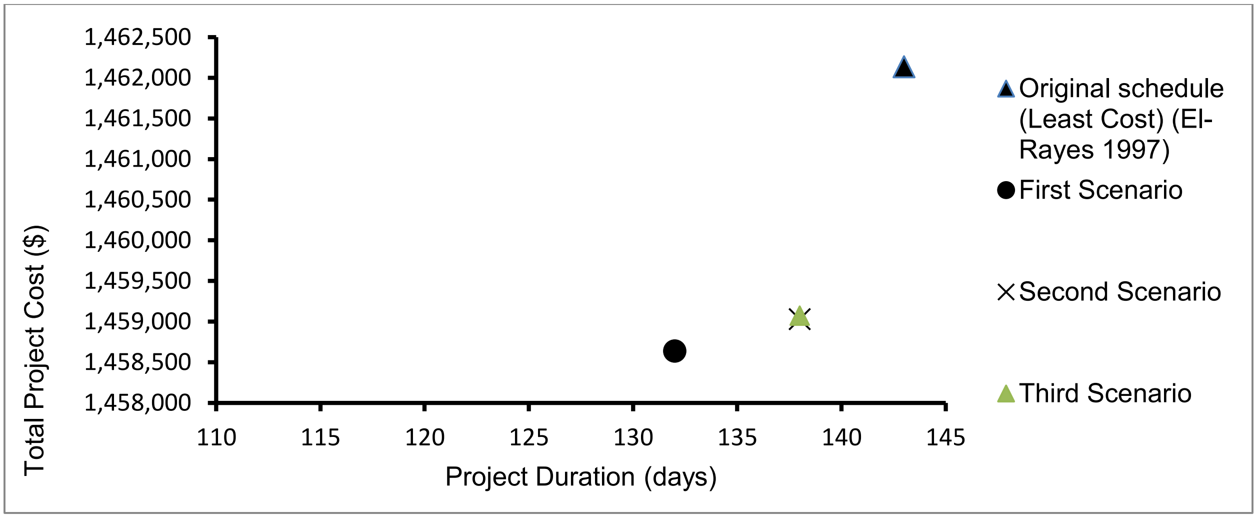


Figure 5.14: Results of 3 Scenarios (Case Study II)

Figure 5.15 shows the accelerated schedule generated in the first scenario and the original schedule, while Figure 5.16 shows the same but for the schedule generated in the second and third scenarios.

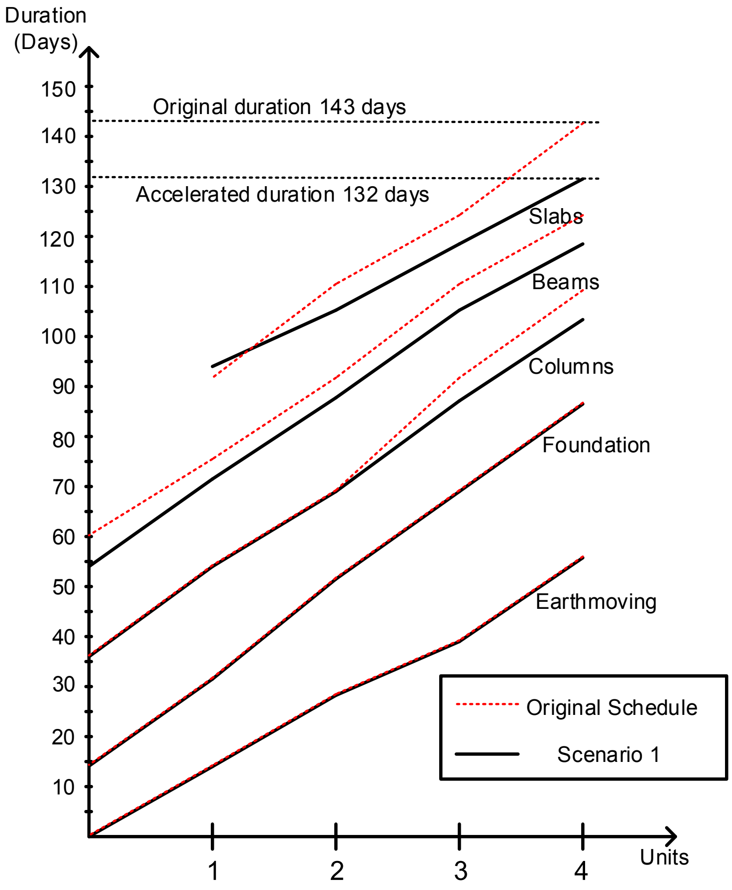


Figure 5.15: Original Schedule Vs Scenario 1 (Case Study II)

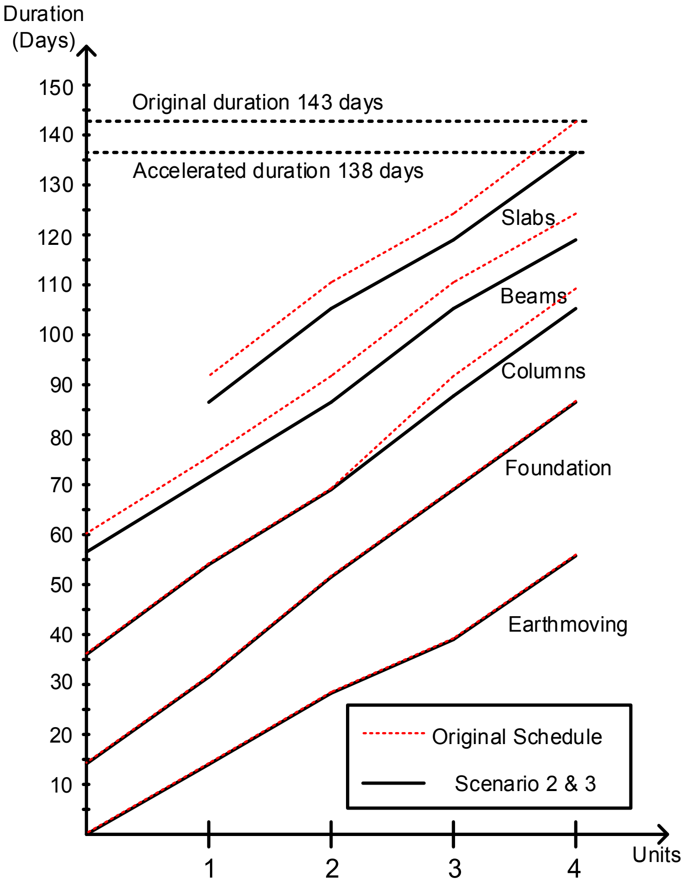


Figure 5.16: Original Schedule Vs Scenario 2 & 3 (Case Study II)

An overview of the acceleration plans generated for the 3 scenarios detailed above in the two case studies reveals a number of facts. The first scenario that overlooked contractor's judgment and addressed the cost slope in a deterministic approach located the least project's total cost among the 3 scenarios. In the second scenario, the prioritization of units for acceleration differed than the first scenario as the contractor's judgment was taken into consideration. Accordingly the identified least cost was more than that identified in the first scenario. Furthermore, when uncertainty was also considered, the acceleration plan changed accordingly, and the identified least total cost was still more than the first cost based scenario and at a different duration too. These results show that the selection of the acceleration plan differs significantly when uncertainties in additional cost and/or contractor's judgment are taken into consideration.

The above scenarios were analyzed using the developed spread sheet application. The application is designed using Microsoft Excel® 2010 Macro-Enabled Worksheet. The computer used has a Core (TM) i5-2400 CPU at 3.1 GHz processor and 8.00 GB of installed memory. The running time for each of the scenarios was below 4 minutes.

Chapter 6: Conclusion

6.1 Summary and Concluding Remarks

Throughout the stages of this research several lessons have been learnt and several conclusions have been drawn. Starting with the review of the existing literature which revealed a number of findings, these are 1) the general lack of tools and techniques tailored for optimized scheduling of repetitive construction projects under uncertainty, 2) the difficulty in using tools and techniques designed for traditional projects for managing repetitive projects, 3) the lack of a comprehensive buffering approach capable of addressing main sources of uncertainty without relying on relevant historical data, 4) the suitability of dynamic programming for the optimized scheduling of this category of projects, 5) the inability of the existing techniques to account for uncertainties in acceleration costs and other influential factors, and finally 6) the unsuitability of the followed approach of activity based acceleration in comparison to unit based acceleration.

To circumvent the limitations of current practices three integrated models are developed. The first model Optimized Scheduling and Buffering is designed for the preconstruction phase. It presents a fuzzy dynamic programming optimization that is capable of modelling uncertainties in different input parameters, it identifies the optimum crew formation that is capable of generating the least cost or the least duration schedule. The analyzed case study revealed the usability of fuzzy set theory as means of modelling uncertainty in various input parameters, such as work quantities, productivities and costs.

Through comparing results to a previous deterministic model, this model's ability to perform dynamic programming utilizing fuzzy input was verified. This model identified the exact same optimum crew formation generating the least cost schedule, with the same schedule duration and less than 1% deviation in total project cost. Also the different scenarios analyzed showed the responsiveness of the model to the change in the objective functions. This was demonstrated when the optimization objective was changed to least duration and the model responded by selecting crews having higher productivities and higher costs.

The model continues after that with the defuzzification of the schedule, where the schedule is converted into a deterministic schedule. The analyzed case study demonstrated the ease of utilizing the expected values (EV) of the optimized fuzzy schedule to assess the amount of uncertainty affecting individual activities. It also showed the ability of the Agreement Index (AI) to capture the user's desired confidence level in the schedule and size and insert time buffers accordingly. The analysis of the generated schedules reflects the effect of the insertion of time buffers, as they increased total project duration, but had a less corresponding total cost. The end product of this first model is an optimized deterministic schedule that is protected with time buffers against various anticipated delays.

The second model is the Monitoring and Dynamic Rescheduling Model is to be used during the construction phase. This model serves the purpose of capturing actual onsite progress and using it for updating the completed part of the schedule. It presents a novel approach for learning from the project and utilizing actual progress on site to reduce uncertainty in the remaining part of the schedule. A developed index named buffer consumption index (BCI) is utilized to track buffer consumption and compare it to that planned as an indication to project progress. As the buffers are inserted at least duration between successive activities, they were found to overlap on the controlling activity path, which indicates the importance of monitoring BCI as a pointer to project progress. This model significantly increases the opportunity of controlling the project, as it offers continuous schedule refinement in a more proactive approach and detailed monitoring of delays and their impact.

The third and final model is the Acceleration Model, it is to be used when the need arises to speed up the delivery of a project. It identifies units to accelerate using a modified unit based acceleration algorithm and queues those units for acceleration while accounting not only for uncertainties in acceleration costs, but also for contractor’s judgment as an additional queuing criteria. The unit based acceleration allows a more focused assignment of acceleration resources, in comparison to activity based acceleration. It also makes this model suited for non-typical repetitive projects. Through offering contractor's judgment as additional queuing criteria, this model is more practical as it is closer to what is practically followed in the industry. Through offering different acceleration strategies, this model generates different alternatives for accelerating project execution and identifies least cost acceleration plans. The analyzed case studies experimented with different scenarios, to evaluate different features of this model. It was concluded that the contractor’s judgment is an important criteria that plays a significant role in generating the least cost acceleration plan, as it changes the priorities of units to accelerate. The case study also showed that uncertainty in the acceleration cost has its impact on the acceleration plan.

6.2 Research Contributions

The three integrated models of the developed methodology provide flexible and comprehensive tools for optimized scheduling of repetitive construction projects under uncertainty. During the research and as the methodology was being developed, several original contributions have been made:

- Developed optimized scheduling and buffering model capable of optimizing schedules for least project duration or least total project cost, while offering users enough flexibility to model uncertainties in different input parameters.
- Developed buffering approach that provides a systematic tool for building buffers to protect the project schedule against anticipated delays. The buffer sizing introduces the utilization of the agreement index factor (AI) to build buffers in order to meet owner's desired confidence level in the generated schedule.
- Developed dynamic rescheduling model that capitalizes on the repetitive nature of the project. Through the utilization of onsite data capturing, this model adjusts uncertainty estimates for the remaining part of the project.
- Developed the buffer consumption index (BCI), which forms a new approach for tracking project progress through comparing buffer consumption to date to that planned.

- Developed a schedule acceleration model that comprises unit-based acceleration algorithm capable of finding least cost acceleration plans through accelerating units instead of activities.
- Developed queuing criteria for schedule acceleration that allows accounting for uncertainty in acceleration cost and for contractor's judgment while prioritizing units for acceleration.

6.3 Research Limitations

The main limitations are summarized as follows:

- This methodology utilized only triangular fuzzy numbers to model uncertainties in different input parameters.
- The schedule optimization model is a single objective model that addresses either cost or duration, but not both.
- The evaluation of the buffer consumption index (BCI) as a pointer to project performance is presented in a subjective approach, without exploring different thresholds for different project activities.
- The acceleration model investigates the alignment of each unit separately, which necessitates performing a large amount of calculations. In case of a bigger schedule with a larger number of strategies for acceleration, the developed spread sheet application needs to be converted into a standalone application with significantly better computational capabilities.

- The acceleration model does not offer changing job logic as a strategy to accelerate project delivery.
- The acceleration model was trying to find the least cost acceleration strategy without taking into consideration the practicality or consistency in adding acceleration resources.

6.4 Opportunities for Future Work

Based on the research conducted the following is recommended for future work. The recommendations are presented in two categories, enhancements to the presented methodology and extensions to it.

6.4.1 Improving Presented Methodology:

- Some features included in other scheduling techniques could be incorporated in the developed methodology, examples of which are learning curve effect, loss of productivity while assigning more acceleration resources due to over-crowdedness, and having limitations for crews' availability periods during the project.
- Buffer consumption index (BCI) is presented as a measure for project progress, further exploration of such index could include associating certain BCI values with the automatic triggering of corrective actions, and detailed comparisons to Earned Value analysis.
- Adding an additional objective for least risk optimization could be looked into.

6.4.2 **Extending Presented Methodology:**

- Using fuzzy rules, where a user can evaluate sources of uncertainty linguistically and a corresponding buffer would be calculated accordingly.
- Considering multi-objective optimization that is capable of addressing uncertainties.
- Using discrete event simulation in the case of the availability of relevant historical data could be a more objective approach for buffer sizing.

References

Al Sarraj, Z. M. (1990). “Formal development of Line‐of‐Balance technique.” *Journal of Construction Engineering and Management,* 116(4), 689-704.

Albulak, M. Z. (1986). “Line‐of‐Balance scheduling in pavement construction.” *Journal of Construction Engineering and Management,* 112(3), 411-424.

Ammar, M. (2010). “Optimization of project time-cost trade-off problem with discounted cash flows.” *Journal of Construction Management and Economics,* 137(1), 65–71.

Arditi, D., and Albulak, M. (1979). “Comparison of network analysis with line-of-balance in a linear repetitive construction project.” *Proceedings of the 6th INTERNET Congress*, Garmish-Partenkirchen, Germany, 12-25.

Arditi, D., Tokdemir, O., and Suh, K., (2002). “Challenges in line-of-balance scheduling.” *Journal of Construction Engineering and Management,* 128(6), 545–56.

Ashley, D. B. (1980). “Simulation of repetitive-unit construction.” *Journal of the Construction Division,* 106(2), 185-194.

Ayyub, B. M. (1984). “Project scheduling using fuzzy set concepts.” *Journal of Construction Engineering and Management,* 110(2), 189-204.

Bakry, I., Moselhi, O. and Zayed, T. (2013). “Fuzzy Dynamic Programming for Optimized Scheduling of Repetitive Construction Projects”. *IFSA World Congress and NAFIPS Annual Meeting (IFSA/NAFIPS)*, 1172-1176. IEEE.

Bakry, I., Moselhi, O., & Zayed, T. (2014). “Optimized acceleration of repetitive construction projects.” *Journal of Automation in Construction,* 39, 145-151.

Carr, R. I. (1979). “Simulation of construction project duration.” *Journal of the Construction Division,* 105(2), 117-128.

Carr, R. I., and Meyer, W. L. (1974). “Planning construction of repetitive building units.” *Journal of the Construction Division,* 100(3), 403-412.

Chehayeb, N., and Abourizk, S., (1998). “Simulation-based scheduling with continuous activity relationships.” *Journal of Construction Engineering and Management.* 124(2), 107-115.

Chrzanowski Jr, E. N. (1986). “Application of linear scheduling.” *Journal of Construction Engineering and Management,* 112(4), 476-491.

Chung, T. (2007) “Simulation-based productivity modelling for tunnel construction operations.” *Ph.D. dissertation, Department of Civil and Environmental Engineering, University of Alberta, Edmonton, Alberta.*

Clark, D. R. (2006). *Beginning object-oriented programming with VB 2005: From novice to professional.* Springer.

Cyril Northcote, P. (1957). *Parkinson's law, and other studies in administration.* Houghton Mifflin.

Ding, C., Fujimura, S., Wei, X., and Wei, W. (2010). “Time-Cost Trade-off Problem under Uncertainty Incorporating Multi-Objective Genetic Algorithm and Fuzzy Sets Theory.” *Proceedings of the 17th Conference on Industrial Engineering and Engineering Management* (IE&EM), 29-31 Oct., 290-294.

El - Rayes, K. (1993). “Scheduling of repetitive projects with cost optimization.” *Journal of Construction Engineering and Management,* 119(4), 681-697.

El-Rayes, K. A. (1997). “Optimized Scheduling for Repetitive Construction Projects.” *PhD Dissertation, Concordia University, Montreal, Quebec.*

El-Rayes, K., and Moselhi, O. (2001). “Optimizing resource utilization for repetitive construction projects.” *Journal of Construction Engineering and Management,* 127(1), 18-27.

El-Rayes, K., and Kandil, A. (2005). “Time-cost-quality trade-off analysis for highway construction.” *Journal of Construction Engineering and Management*, 131(4), 477-486.

Eshtehardian, E., Abbasnia, R., and Afshar, A. (2008 a). “Optimization of uncertain construction time-cost trade-off problem.” *Proceedings of the 1st International Conference on Construction in Developing Countries (ICCIDC–I) “Advancing and Integrating Construction Education, Research & Practice”* August 4-5, Karachi,, Pakistan. 192- 200.

Eshtehardian, E., Abbasnia, R., and Afshar, A. (2008 b). "Time–cost optimization: using GA and fuzzy sets theory for uncertainties in cost." *Journal of Construction Management and Economics,* 26(7), 679–691.

Ezeldin, A. S., and Soliman, A. (2009). "Hybrid time-cost optimization of nonserial repetitive construction projects." *Journal of Construction Engineering and Management*, 135(1), 42-55.

Fallah, M., Ashtiani, B., and Aryanezhad, M. B. (2010). "Critical chain project scheduling: utilizing uncertainty for buffer sizing." *International Journal of Research and Reviews in Applied Science,* 3(3).

Farag, M., Gehbauer, F., and Bhatla, A. "An integration of a buffer assessment model based on fuzzy logic with LPS™ for improving highway construction process." *Proceeding of the 18th Conference of the International Group of Lean Construction,* Haifa, Palestine, 264-273.

Georgy, M. E. (2008). "Evolutionary resource scheduler for linear projects." *Journal of Automation in Construction*, 17(5), 573-583.

Goldratt, E. M. (1997). *Critical chain.* North River Press Great Barrington, MA.

Grey, J. R. (2007). "Buffer Techniques for Stochastic Resource Constrained Project Scheduling with Stochastic Task Insertions Problems.*" PhD Dissertation, University of Central Florida, Orlando, Florida.*

Hajjar, D., and AbouRizk, S. (2002). "Unified modelling methodology for construction simulation." *Journal of Construction Engineering and Management*, 128(2), 174-185.

Halpin, D.W. (1973). "An investigation of the use of simulation networks for modelling construction operations", *PhD Thesis, University of Illinois, Urbana-Champaign, Illinois.*

Harmelink, D.J., and Rowings, J. (1998). "Linear scheduling model: development of controlling activity path." *Journal of Construction Engineering and Management,* 124(4) 263–268.

Hassanein, A. (2003). "Planning and scheduling highway construction using GIS and dynamic programming." *PhD Dissertation, Concordia University, Montreal, Quebec.*

Hassanein, A., and Moselhi, O. (2005). “Accelerating linear projects.” *Journal of Construction Management & Economics,* 23(4), 377-385.

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(4), 382-391.

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(4), 333-342.

Hegazy, T. and Kamarah, E. (2008). ”Efficient Repetitive Scheduling for High-Rise Construction.” *Journal of Construction Engineering and Management,* 134(4), 253–264.

Hegazy, T., and Kassab, M. (2003). “Resource optimization using combined simulation and genetic algorithms.” *Journal of Construction Engineering and Management*, 129(6), 698-705.

Hegazy, T., and Wassef, N. (2001). “Cost optimization in projects with repetitive nonserial activities.” *Journal of Construction Engineering and Management*, 127(3), 183-191.

Henderickson, C., and Au, T. (1989). *Project management for construction*. Prentice-Hall, Englewood Cliffs, N.J.

Herroelen, W., and Leus, R. (2001). “On the merits and pitfalls of critical chain scheduling.” *Journal of Operation Management.* 19(5), 559-577.

Herroelen, W., Leus, R., and Demeulemeester, E. (2002). “Critical chain project scheduling-Do not oversimplify.” *Project Management Journal*. 33(4), 46-60

Hiyassat, M. (2001). “Applying modified minimum moment method to multiple resource leveling.” *Journal of Construction Engineering and Management*, 127(3), 192–8.

Hoel, K., and Taylor, S. G. (1999). “Quantifying buffers for project schedules.” *Production and Inventory Management Journal,* 40, 43-47.

Huang, R. Y., and Sun, K. S. (2009). “A GA optimization model for workgroup-based repetitive scheduling (WoRSM).” *Advances in Engineering Software*, 40(3), 212-228.

Hyari, K., and El-Rayes, K. (2006). “Optimal planning and scheduling for repetitive construction projects.” *Journal of Management in Engineering,* 22(1), 11-19.

Hyari, K. H., El-Rayes, K. and El-Mashaleh, M. (2009) "Automated trade-off between time and cost in planning repetitive construction projects." *Journal of Construction Management and Economics*, 27(8), 749-761

Ipsilandis, P. G. (2007). "Multiobjective linear programming model for scheduling linear repetitive projects." *Journal of Construction Engineering and Management*, 133(6), 417-424.

Kavanagh, D. P. (1985). "SIREN: A repetitive construction simulation model." *Journal of Construction Engineering and Management,* 111(3), 308-323.

Kaufmann, A., and Gupta, M. (1991). *Introduction to fuzzy arithmetic: Theory and application*. NY: Van Nostrand Reinhold.

Liu, W., Flood, I., and Issa, R. R. A. (2005). "Simulation and optimization of linear construction projects." *Proceedings of the 2005 ASCE International Conference on Computing in Civil Engineering*, July 12-15 Cancun, Mexico.

Liu, S. and Wang, C. (2007). "Optimization model for resource assignment problems of linear construction projects." *Journal of Automation in Construction,* 16(4), 460–73.

Long, L. D., and Ohsato, A. (2007). "Fuzzy Activity Network Method for Project Scheduling Under Resource Constraints." *Journal of Advanced Computational Intelligence and Intelligent Informatics*, *11*(8), 914-921.

Long, L. D., and Ohsato, A. (2008). "Fuzzy critical chain method for project scheduling under resource constraints and uncertainty." *International Journal of Project Management*, 26(6), 688-698.

Long, L. D., and Ohsato, A. (2009). "A genetic algorithm-based method for scheduling repetitive construction projects." *Journal of Automation in Construction*, 18(4), 499-511.

Lorterapong, P. (1995). "Fuzzy Project-Network Scheduling Under Resource Constraints.*" PhD Dissertation, Concordia University, Montreal, Quebec.*

Lorterapong, P., and Moselhi, O. (1996). "Project-network analysis using fuzzy sets theory." *Journal of Construction Engineering and Management*, 122(4), 308-318.

Lumsden, P. (1968). *The line-of-balance method.* Pergamon Press Limited.

Lutz, J. D. (1990). "Planning of linear construction projects using simulation and line of balance." *PhD Dissertation, Purdue University, Lafayette, Indiana*.

Lutz, J. D., and Halpin, D. W. (1992). Analyzing linear construction operations using simulation and line of balance. *Transportation Research Record*, 1351, 48-56.

Maravas, A. and Pantouvakis, J.P. (2011). "Fuzzy repetitive scheduling method for projects with repeating activities." *Journal of Construction Engineering and* Management, 137(7), 561–564.

Min, Z., and Rongqiu, C. (2008). "Buffer sized technique in critical chain management: A fuzzy approach." *Proceedings of the 4th International Conference on Wireless Communications, Networking and Mobile Computing. WiCOM'08.* Dalian, China, 1-4.

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

Moselhi, O., and El-Rayes, K. (1993). "Scheduling of repetitive projects with cost optimization." *Journal of Construction Engineering and Management*, 119(4), 681-697.

Moselhi, O., and Hassanein, A. (2003). "Optimized scheduling of linear projects." *Journal of Construction Engineering and Management*, 129(6), 664-673.

Moselhi, O., and Alshibani, A., (2010). "Cost and Experience Based Method for Project Acceleration." *Proceedings of the 6th International Structural Engineering and Construction Conference (ISEC-6) Zürich, Switzerland*, 21-26 June.

Moselhi, O. and Roofigari-Esfahan, N. (2012) "Compression of Project Schedules using the Analytical Hierarchy Process." *Journal of Multi-Criteria Decision Analysis.* 19, 67–78

Nassar, K. (2005). "Evolutionary optimization of resource allocation in repetitive construction schedules." *Journal of Information Technology in Construction*, 10, 265-273.

Okada, S., and Gen, M. (1994). "Fuzzy shortest path problem." *Journal of Computers & industrial engineering,* 27(1), 465-468.

Pagnoni, A. (1990). *Project engineering: computer oriented planning and operational decision making.* Spring, Berlin and Heidelberg, Germany.

Perera, S. (1982). “Network planning of projects comprising repetitive activities.” *Proceeding of IAHS Conference on Impact of Economy and Technology*, Vienna, Austria, Nov. 15-18,927-985.

Perera, S. (1983). “Resource sharing in linear construction.” *Journal of Construction Engineering and Management*, 109(1), 102-111.

Polat, G., Buyuksaracoglu, Y., and Damci, A. (2009). “Scheduling asphalt highway construction operations using the combination of line-of-balance and discrete event simulation techniques.” *Proceedings of the International Conference in Industrial Engineering and Engineering Management*, December 8-11, Hong Kong, 1126-1130.

Raz, T., Barnes, R., and Dvir, D. (2003). “A critical look at critical chain project management.” *Project Management Journal,* 34(4), 24-32.

Reda, R. M. (1990). “RPM: Repetitive project modelling.” *Journal of Construction Engineering and Management*, 116(2), 316-330.

Rezaie, K., Manouchehrabadi, B., and Shirkouhi, S. N. (2009). “Duration estimation, a new approach in critical chain scheduling.” *Proceedings of the 3rd Asia International Conference on Modelling & Simulation, AMS'09,* May 25-29, Bali, 481-484.

Robinson, H., and Richards, R. (2010). “Critical chain project management: Motivation & overview.” *Aerospace Conference (IEEE), March 6-13, Big Sky, Montana,* 1-10.

Russell, A. D., and Caselton, W. F. (1988). “Extensions to linear scheduling optimization.” *Journal of Construction Engineering and Management*, 114(1), 36-52.

Russell, M. M., Hsiang, S. M., Liu, M., and Wambeke, B. (2014). “Causes of Time Buffer and Duration Variation in Construction Project Tasks: Comparison of Perception to Reality.” *Journal of Construction Engineering and Management*.

Siemens, N. (1971). "A simple CPM time-cost trade-off algorithm." *Management Science*, 17(6), 354–363.

Selinger, S. (1980). “Construction planning for linear projects.” *Journal of the Construction Division,* 106(CO2), 195-205.

Senior-Brown, B. A. (1993). “A study of the planning and integrated cyclic analysis of serial system operations.” *Ph.D. Thesis, Purdue University, West Lafayette, Indiana*.

Senior, B.A., and Halpin, D.W. (1994). “Integrated approach to repetitive construction planning.” *Proceedings of the 1st Annual Conference on Computing in Civil Engineering,* June 20-22, Washington, D.C., 25-32.

Shaheen, A., Robinson, A., and AbouRizk, S. (2007). “Fuzzy numbers in cost range estimating.” *Journal of Construction and Engineering and Management*, 133(3), 325-334.

Shankar, N. R., Raju, M., Srikanth, G., and Bindu, P. H. (2011). “Time, cost and quality trade-off analysis in construction of projects.” *Journal of Contemporary Engineering Sciences,* 4(6) 289-299.

Shi, Q., and Gong, T. (2009). “An improved project buffer sizing approach to critical chain management under resources constraints and fuzzy uncertainty.” *Proceedings of the 8th International Conference on Artificial Intelligence and Computational Intelligence,* Nov 7, Shanghai, China 486-490.

Shou, Y., and Yao, K. (2000). “Estimation of project buffers in critical chain project management.” *Proceedings of the 1st International Conference on the Management of Innovation and Technology,* Nov 12-15, Singapore, 162-167.

Srisuwanrat, C. (2009). “The sequence step algorithm: A simulation-based scheduling algorithm for repetitive projects with probabilistic activity durations.” *PhD Dissertation, Michigan University, Michigan, USA.*

Srisuwanrat, C., Ioannou, P. G., and Tsimhoni, O. (2008). “Simulation and optimization for construction repetitive projects using promodel and SimRunner.*” Proceedings of the Winter Simulation conference*, Dec 7-10, Austin, Texas, 2402-2412.

Tukel, O. I., Rom, W. O., and Eksioglu, S. D. (2006). “An investigation of buffer sizing techniques in critical chain scheduling.” *European Journal of Operational Research,* 172(2), 401-416.

Vorster, M. C., and Bafna, T. (1992). Discussion of "Formal development of line-of-balance technique". *Journal of Construction Engineering and Management,* 118(1), 210-211.

Wan, Y. (1994). “Resource allocation for a stochastic CPM-type network through perturbation analysis.” *European Journal of Operational Research*, 79(2), 239–248.

Wei Feng, C., Liu, L., and Burns, S. (2000). “Stochastic construction time-cost trade off.” *Journal of Computing in Civil Engineering*, 14(2), 117-126.

Wilson, J.R. (1994). “Simulation of learning development in repetitive construction.” *Journal of Construction Engineering and Management,* 120(4), 753-773.

Wong, W. C. M. (1993). “New generation of planning structures.” *Journal of Construction Engineering and Management,* 119(2), 196-214.

Yang, I. (2002). “Repetitive project planner: Resource-driven scheduling for repetitive construction projects.” *PhD Dissertation, University of Michigan, Michigan.*

Yang, I., and Chang, C. Y. (2005). “Stochastic resource-constrained scheduling for repetitive construction projects with uncertain supply of resources and funding*.” International Journal of Project Management*, 23(7), 546-553.

Zadeh, L. A. (1965). Fuzzy sets. *Information and Control, 8*(3), 338-353.

Zayed, T., Amer, M., and Pan, J. (2008). “Assessing risk and uncertainty inherent in Chinese highway projects using AHP.” *International Journal of Project Management*, 26(4), 408-419.

Zhao, Z. Y., You, W. Y., and Lv, Q. L. (2008). “Applications of fuzzy critical chain method in project scheduling.” *Proceedings of the 4th Conference in Natural Computation,* Oct 18-20, Jinan, China, 473-477.

Zhang, X., Cui, N., Bie, L., and Zuo, F. (2011). “Buffer sizing method with considering effect of various uncertainties based on fuzzy comprehensive evaluation method in critical chain scheduling.” *Proceedings of the 2nd International Conference on Emergency Management and Management Sciences,* Aug 8-10, Beijing, China, 566-569.

Zheng, D., Thomas Ng, S., and Kumaraswamy, M. (2004). “Applying a Genetic Algorithm-Based Multi-objective Approach for Time-Cost Optimization.” *Journal of Construction Engineering and management*, 130(2), 168-176.