# STRUCTURING NPD PROCESSES: ADVANCEMENTS IN TEST SCHEDULING AND ACTIVITY SEQUENCING

QIAN YANJUN

NATIONAL UNIVERSITY OF SINGAPORE

2009

# STRUCTURING NPD PROCESSES: ADVANCEMENTS IN TEST SCHEDULING AND ACTIVITY SEQUENCING

QIAN YANJUN

(*M.Mgt., Xian Jiaotong University, China*)

A THESIS SUBMITTED

FOR THE DEGREE OF DOCTOR OF PHILOSOPHY

DEPARTMENT OF INDUSTRIAL & SYSTEMS ENGINEERING

NATIONAL UNIVERSITY OF SINGAPORE

2009

# ACKNOWLEDGEMENTS

First of all, I would like to express my deep and sincere gratitude to my supervisor, Professor Goh Thong Ngee, for his patience and seasoned guidance of my research, and for his important support throughout this work. His wide knowledge and logical way of thinking have been of great value for me. His understanding and encouraging have provided a good basis for the present thesis. I would also like to thank Professor Xie Min for his guidance, constructive comments and suggestions on my research. His enthusiasm in research and hard-working has greatly motivated me throughout this work.

I wish to thank Associate Professor Tan Kay Chuan and Dr. Wikrom Jaruphongsa who served on my oral examination committee and provided me helpful comments on my thesis research. I would like to thank all the other faculty members in the Department of Industrial and Systems Engineering, from whom I have learnt a lot through coursework and discussions. I also wish to thank Ms. Ow Lai Chun and Mr. Lau Pak Kai for their excellent administrative support during my PhD study.

I must acknowledge the National University of Singapore for offering me a Research Scholarship. I wish to thank the members of Quality and Reliability Lab, for their friendship and kind help throughout my thesis research. I also wish to express my appreciation for the great assistance received from our case study companies.

Last but not least, thanks my husband Lin Jun, my parents and my parents-in-law, for their unflagging love and support during my PhD study. They have lost a lot due to my research abroad. Without their encouragement and understanding it would have been impossible for me to finish this work.

# TABLE OF CONTENTS

# SUMMARY

Efficient New Product Development (NPD) processes are critical to the success of many modern corporations. Motivated by needs of companies and research gaps identified, this thesis focuses on two key decision problems for structuring NPD processes: test scheduling and activity sequencing, and consists of two parts.

The first part views the NPD process as consisting of a series of development stages and deals with the test scheduling problem. Past studies, which are developed to determine the optimal scheduling of tests, often focused on single-stage testing of sequential NPD process. Meanwhile, overlapping has become a common mode of product development. We therefore present two analytical models for the optimal scheduling of tests in overlapped NPD process.

When the testing set-up time is relatively small, the analytical model in Chapter 3 can help management decide when to stop testing at each stage, and when to start downstream development (e.g. mold fabrication). The model in Chapter 3 also yields several useful insights. When the testing set-up time is long, the analytical model in Chapter 4 can help decision makers determine the optimal number of tests needed at each stage, together with the optimal overlapping policies. The impact of different model parameters on the optimal solution is also discussed, which can help the management adjust testing and overlapping strategies for NPD processes with different characteristics. These two analytical models are illustrated with two case studies in consumer electronics companies.

A development stage may be further broken down into smaller activities. Since there are no clear precedence constraints among activities, another key and challenge issue is how to plan the time and sequence of activities, which is the focus of the second part of this thesis. Formal network-based techniques, such as CPM and PERT, cannot effectively model cyclic information flows and iteration, limiting their capability of planning NPD processes. To address this shortfall, one popular approach is Design Structure Matrix (DSM), which has spawned many research efforts on sequencing design activities with the objective of minimizing feedbacks. However, the problem is NP-complete. To solve large problems, we follow previous decomposition methods and present two new approaches.

In Chapter 5, we first propose two simple rules for feedback reduction through activity exchange. After that, a new decomposition approach is presented for solving large DSM sequencing problem. We have also applied the proposed solution strategy to three real data sets, and show that compared to the solutions presented in previous studies, applying our approach results in better solutions with smaller feedbacks. In Chapter 6, we further establish rules of block-activity exchange and block-block exchange, for feedback reduction. We find that based on the fold operation, a block has similar properties to a single activity. Based on these findings, a novel decomposition approach is presented. One advantage of this approach is that it can solve the sub-problems in parallel. Finally, in some situations, activity dependencies may not be precisely estimated, we therefore present a fuzzy approach to DSM sequencing problem. The methodology is applied to the powertrain development, and is shown that it can help managers better manage NPD processes with uncertainty.

# LIST OF TABLES

# LIST OF FIGURES

# NOMENCLATURE

| | |
|---|---|
| AHP | Analytic Hierarchy Process |
| CE | Concurrent Engineering |
| CPM | Critical Path Method |
| DeMAID | Design Manager's Aid for Intelligent Decomposition |
| DSM | Design Structure Matrix |
| EOQ | Economic Order Quantity |
| GA | Genetic Algorithm |
| GERT | Graphical Evaluation and Review Technique |
| LIP | Linear Integer Program |
| NDSM | Numerical Design Structure Matrix |
| NHPP | Non-Homogeneous Poisson Process |
| NP | Nondeterministic Polynomial time |
| NPD | New Product Development |
| PERT | Program Evaluation and Review Technique |
| QFD | Quality Function Deployment |
| QIP | Quadratic Integer Program |

# CHAPTER 1

# INTRODUCTION

Efficient New Product Development (NPD) processes are critical to the success of many modern corporations. To model and structure NPD processes, decisions are often made about the testing strategies for project monitoring and control, the degree of overlapping, and the planned timing and sequence of design activities. Motivated by needs of companies and research gaps identified, this thesis contributes to some methodological issues for scheduling tests in overlapped product development and for sequencing design activities with iteration loops. In this introductory chapter, we first show the necessity for modeling and structuring NPD processes in Section 1.1, followed by the research gaps proposed in Section 1.2. In Section 1.3, we discuss the scope and objectives of our study. Finally, the structure of this thesis is presented in Section 1.4.

## 1.1 Need for Modeling and Structuring NPD Processes

An NPD process is a formal template through which a company can repetitively convert ideas into new products (Cooper, 1994; Browning, 2009). Such a process defines and describes the required steps and resources for driving new product projects from ideas to launch (Rosenau et al., 1996; Biazzo, 2009). Facing intense competition, rapidly evolving technologies, changing customer needs, and shorter product life cycles, many firms need to develop lower cost, higher quality products at a rapid pace (Cooper, 2001; Mitchell and Nault, 2007). An efficient NPD process is essential to achieve these goals, and thus is critical to the success of many modern corporations (Rosenau and Githens, 2005; Bhaskaran and Krishnan, 2009).

However, structuring the NPD process is challenging. Part of the difficulty is due to the following characteristics of the NPD process:

(1) Complex interaction among activities. A typical NPD process can be divided into a series of development stages. A development stage may further be broken down into smaller activities. Unlike the manufacturing process, the NPD process often involves a number of decision-making activities, for example, the design of an automobile may involve thousands of engineers making millions of design decisions (Eppinger et al., 1994). Moreover, none of these activities are performed in isolation; instead, each design choice may affect many other design parameters (Eppinger et al., 1994; Pich et al., 2002; Pektas and Pultar, 2006).

(2) Iteration. Iteration, the repetition of design activities, is a fundamental characteristic of the NPD process (Black and Repenning, 2001; Gil, 2009). There are two main reasons why iteration is commonly occurred in an NPD process. First, the outputs of activities, such as engineering drawings, specifications and bill of materials, are often unstable and inaccurate, and need to be reworked when downstream activities detect some faults in the original design (Gil et al., 2004; Terwiesch and Xu, 2008). Second, downstream activities may be repeated when modified information is passed along from upstream activities (Smith and Eppinger, 1997b; Loch and Terwiesch, 2005; Love et al., 2009).

(3) Conflicting product development performance. Generally, there are three measures of product development performance: completion time, development cost, and product quality (Cohen et al., 1996; Pollack-Johnson and Liberatore, 2006; Ulrich

and Eppinger, 2008). However, these metrics are often conflict with each other. For example, changing the development policies may reduce project completion time, but may at the cost of lower product quality and/or higher development cost (Harter et al. 2000; Harter and Slaughter, 2003; Wu et al., 2009).

In recent years, product development undergoes new trends such as distributed product development, cross-functional teams, and overlapping product development stages (Nambisan, 2002; Gerwin and Barrowman, 2002; Zhou et al., 2005; Novak and Stern, 2008). These new trends further increase the uncertainty and complexity of NPD processes (O'Sullivan, 2003; Bhuiyan et al., 2006; Kang and Hong, 2009). Therefore, efficient and effective models are needed to represent above essential characteristics and new trends of NPD processes so as to systematically analyze the effect of development policies on the product development performance, then improve and optimize the product development performance.

## 1.2 Research Gaps

NPD process modeling has received considerable attention over the last 15 years from both the academic community and practitioners (MacCormack et al., 2001; Roemer and Ahmadi, 2004; Levardy and Browning, 2009). To model and structure NPD processes, decisions are often made about the testing strategies for project monitoring and control, the degree of overlapping, and the planned timing and sequence of design activities (Krishnan and Ulrich, 2001; Browning and Ramasesh, 2007). In the following subsections, we will briefly introduce these decision problems, some existing models and research gaps.

### 1.2.1 Test Scheduling

A typical NPD process can be viewed as consisting of four consecutive stages: (1) concept development, (2) product design, (3) process design, and (4) pilot production (Cooper, 2001; Chakravarty, 2003). For most product development projects, the initial outputs of these stages inevitably contain design problems, such as mismatches with customer needs or technical design faults (Thomke and Bell, 2001; Gil et al., 2008). Testing, which is a primary way to detect and resolve these problems, is central to product development (Loch et al., 2001; Thomke, 2007; Erat and Kavadias, 2008).

It is known that undetected design problems of an upstream stage (e.g. concept development) will accumulate and proliferate to downstream stage (e.g. product design). Thus, the outputs of an upstream stage need to be tested extensively before releasing them to downstream stage. Inadequate testing would allow design problems to propagate, and finally deteriorate the product quality. On the other hand, testing also incurs time and cost. Too much testing at one stage would impede the project's progress and increase development costs. Thus, how to optimally schedule various tests along the NPD process so as to maximize product development performance is an important decision problem.

Some models have been developed to determine the optimal scheduling of tests and/or reviews for product development projects (e.g. Ha and Porteus, 1995; Thomke and Bell, 2001; Xie and Yang, 2001; Dai et al., 2003; Pham and Zhang, 2003; Serich, 2005; Erat and Kavadias, 2008; Yang et al., 2008; Bartels and Zimmermann, 2009). These models have clearly shed light on the analysis of test scheduling problem. However, they focus on the testing policies at one development stage and do not take into account the multi-stage nature of testing process.

An important contribution in modeling the multi-stage testing is due to Cooper (1980, 1993a, 1993b, 1993c). Their system dynamics models were further extended by several independent researchers, such as Ford and Sterman (1998), Williams et al. (2003), Love et al. (2008), and Lin et al. (2008). While these simulation models have greatly advanced our understanding of the multi-stage testing process, they cannot find good testing policies quickly and effectively, which is a limitation of most simulation models (Cho and Eppinger, 2005; Lancaster and Ozbayrak, 2007).

Ahmadi and Wang (1999), as well as Kogan and Raz (2002), built analytical models and explicitly examined how to optimally schedule tests for multiple development stages. The former assumed that all development stages are carried out in fully sequential, while the latter assumed that all stages start and finish simultaneously. However, in practice, the development stages are often overlapped (i.e. in partial parallel) rather than fully sequential or parallel (Krishnan, 1996; Mitchell and Nault, 2007; Gerk and Qassim, 2008). As far as we know, no analytical model exists for scheduling tests in overlapped NPD process.

### 1.2.2 Overlapping Policies

Overlapping, which refers to the partial parallel execution of development stages, has become a common mode of product development over the last decade as a result of increasing importance of time-to-market (Smith and Reinertsen, 1998; Loch and Terwiesch, 2005; Yan and Xu, 2007). As illustrated in Figure 1.1, in sequential process, the second stage starts only after completion of the first stage, while in overlapped process, the second stage starts earlier with preliminary information of the first stage. Thus, project completion time may be reduced, see e.g. Smith and Reinertsen (1998), Sobek et al. (1999) for successful cases. However, studies also show that overlapping is not applicable to all NPD projects (see e.g. Terwiesch and Loch, 1999; Gil et al., 2008). Because overlapping requires that downstream stages start on preliminary information, rework is often necessary to accommodate upstream design changes (Krishnan et al., 1997; Loch and Terwiesch, 1998; Roemer et al., 2000; Gerk and Qassim, 2008). If the uncertainty or the dependency between development stages is high, most of downstream tasks done on upstream preliminary information need to be reworked, which makes overlapping unfavorable (Krishnan et al., 1997; Helms, 2002; Lin et al., 2010). Thus, analytical investigation of the trade-offs involved is needed.

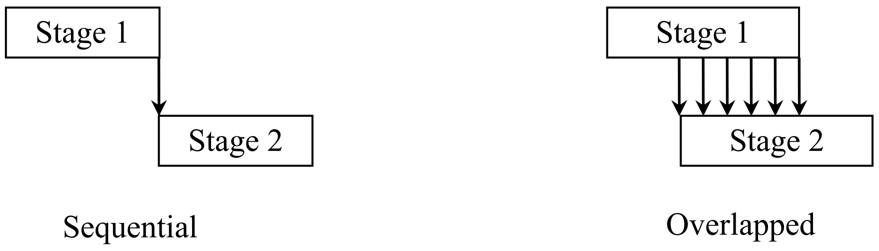


**Figure 1.1 Sequential and overlapped NPD processes**

Many independent researchers have examined this key trade-off and derived optimal overlapping levels for projects with different characteristics (e.g. Krishnan et al., 1997; Loch and Terwiesch, 1998; Roemer et al., 2000; Chakravarty, 2001; Joglekar et al., 2001; Wang and Yan, 2005; Gerk and Qassim, 2008; Lin et al., 2009). These studies are insightful in many respects. However, all of them assume that testing policies are predetermined. Analytical models are needed to combine these two decisions (i.e. test scheduling and overlapping levels) into one modeling framework since they are interacted.

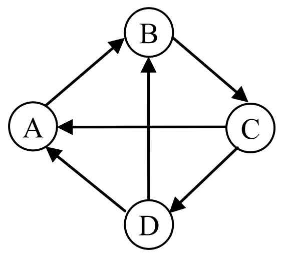
### 1.2.3 Sequencing Design Activities

When the NPD process is viewed as consisting of a series of development stages, it's no need to consider the sequencing problem since the execution sequence of development stages (such as the concept design and the product design) is known. However, when the NPD process is further broken into smaller activities, then, a key and challenging issue, i.e. the planned time and sequence of activities, arises because clear precedence constraints among design activities do not exist and are rarely known in advance (Eppinger et al., 1994; Ahmadi et al., 2001; Karniel and Reich, 2009).

As reported by many researchers (e.g. Eppinger et al. 1994; Rodrigues and Bowers, 1996; Anderson and Joglekar, 2005; Karniel and Reich, 2009), traditional network-based techniques, such as Critical Path Method (CPM) and Program Evaluation and Review Technique (PERT), cannot effectively model cyclic information flows among activities, as well as iteration, limiting their capability of planning for NPD processes. For instance, in the four-activity example shown in Figure 1.2(a), after completion of activity C, the process may iterate back to activity A when activity C discovers some design problems or incompatibility. Similarly, activities A and B may have to be reworked in light of the arrival of new information from activity D. This iterative process is common in most product development projects and PERT/CPM could not deal with such loops effectively.

To address this shortfall, one known method is Design Structure Matrix (DSM). As illustrated in Figure 1.2(b), DSM is a binary matrix representation of a project with elements denoting individual activities which are executed in the temporal order listed from top to bottom (Browning, 2001; Chen and Huang, 2007). Sub-diagonal marks represent information input from upstream activities to downstream, and super-diagonal marks denote feedbacks from downstream activities to upstream (Yassine et al., 1999a; Lancaster and Cheng, 2008). As such, DSM provides a concise way in describing and investigating information dependencies among design activities, as well as iteration (Cho and Eppinger, 2005; Abdelsalam and Bao, 2007).

The DSM approach was first introduced by Steward (1981). Eppinger et al. (1994) extended Steward's work by explicitly including numerical measures of activity dependencies. Figure 1.2(c) shows an example of Numerical DSM (NDSM), where the off-diagonal numbers represents the degree of information dependencies among activities. Since then, many researchers have reported the successful application of DSM/NDSM in managing NPD projects (see e.g. Eppinger, 2001; Clarkson et al., 2004; MacCormack et al., 2006; Sosa, 2008; Voss and Hsuan, 2009). Reviews of DSM approach can be found in Browning (2001), Karniel and Reich (2009).



(a) Graph Representation

| | A | B | C | D |
|---|---|---|---|---|
| A | ■ | | × | × |
| B | × | ■ | | × |
| C | | × | ■ | |
| D | | | × | ■ |

(b) DSM Representation

| | A | B | C | D |
|---|---|---|---|---|
| A | ■ | | 0.2 | 0.3 |
| B | 0.8 | ■ | | 0.4 |
| C | | 0.9 | ■ | |
| D | | | 0.5 | ■ |

(c) NDSM Representation

**Figure 1.2 Iterative NPD process: four-activity example**

It is known that iteration is a major driver for lengthy and costly product development (Smith and Eppinger, 1997b; Ahmadi et al., 2001; Love et al., 2009). To structure NPD processes, the DSM approach suggests to re-sequencing the activities such that iterative behaviors are minimized in the matrix. Over the years, a number of studies have examined how to sequence design activities in a DSM. As reported by Meier et al. (2007), and Lancaster and Cheng (2008), in most of previous studies, the objective of sequencing is to minimize feedbacks[1]. Except for minimizing feedbacks,

[1] For easier explanation, we will refer to such problem as DSM sequencing problem.

some other considerations have also been incorporated in sequencing design activities (see e.g. Altus et al., 1996; Smith and Eppinger, 1997a; Abdelsalam and Bao, 2006).

Several independent researchers have reported that DSM sequencing problem is NP-complete (McCulley and Bloebaum, 1996; Ahmadi et al., 2001; Meier et al., 2007). To solve large-scale sequencing problems, one stream of literature focused on developing meta-heuristic methods, such as Genetic Algorithm (Altus et al., 1996; Whitfield et al., 2003; Meier et al., 2007), Simulated Annealing (Abdelsalam and Bao, 2006, 2007), and Evolutionary Algorithm (Lancaster and Cheng, 2008). Another stream of literature focused on decomposition based methods. More specifically, the overall problem is first decomposed into smaller sub-problems which are easier to solve, and then the sub-problem solutions are merged to a solution of the overall problem. Examples of such studies include McCulley and Bloebaum (1996), Rogers (1996, 1999), Ahmadi et al. (2001).

## 1.3 Research Scope and Objectives

Depending on their newness to the company and marketplace, product innovations can be incremental or radical (Eppinger et al., 1994; Grupp and Maital, 2001; Hauser et al., 2006). Radical innovation often requires developing products with an entirely new technology and/or with an entirely new set of performance features, e.g. certain smart-chip devices (Leifer et al. 2000; Zhou et al. 2005). On the other hand, an extension or improvement of existing products is termed as incremental product innovation. This thesis focuses mainly on incremental product innovation. We also focus product development projects which are economically feasible, in other words, the decision has been made to design and implement the projects. Finally, motivated by needs of companies and research gaps identified, we devote our attention to two key decision problems for structuring NPD processes: test scheduling and activity sequencing. More specifically, we present some analytical models for the optimal scheduling of tests in overlapped NPD process, and propose some approaches for solving large-scale DSM sequencing problem.

### 1.3.1 Optimal Scheduling of Tests in Overlapped NPD Process

Testing is central to product development (Loch et al., 2001; Erat and Kavadias, 2008). Past studies, which are developed to determine the optimal scheduling of tests, often focused on single-stage testing of sequential NPD process. Meanwhile, overlapping has become a common mode of product development (Terwiesch et al., 2002; Yassine et al., 2008; Roemer and Ahmadi, 2010). We therefore present two analytical models for the optimal scheduling of tests in overlapped NPD process.

Let us use a practical example to illustrate the problem studied. As shown in Figure 1.3, the refrigerator development process generally consists of four stages: concept creation, industrial design, detail design, and mold fabrication. Following these stages, four types of tests are carried out. Concept tests use CAD model to test customers' reaction to the proposed new product. Industrial design tests build digital mockups to verify the feasibility of the industrial design. Detail design tests construct engineering prototypes to verify that the design can function, and finally system tests produce concrete refrigerators to improve the overall performance of the product.

Then, how much budget should be allocated to testing the design at each development stage? When should we stop testing? In overlapped process, downstream stages (e.g. mold fabrication) can start at any time after the initial upstream design is available and before the completion of upstream tests (e.g. detail design tests). Then, what is the optimal start time of downstream stages (e.g. mold fabrication)? If overlapping is applied, how should we adjust the testing strategies?

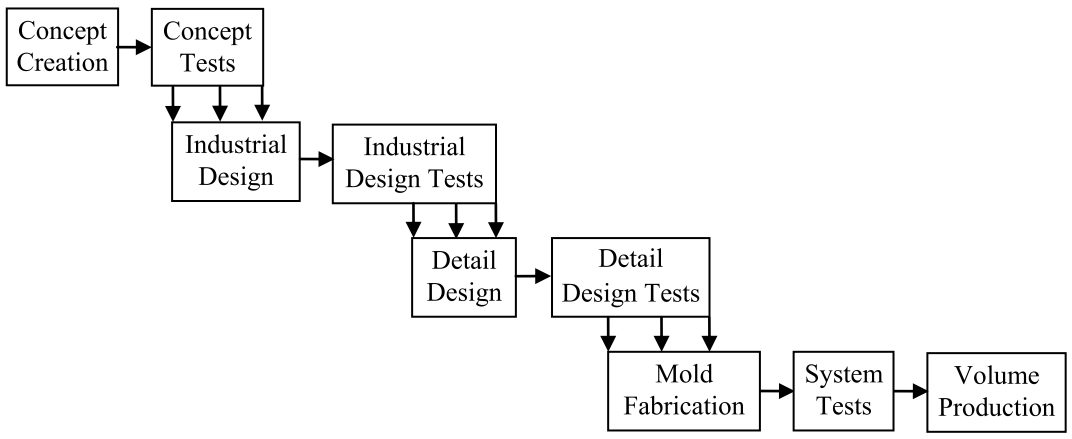


**Figure 1.3 Refrigerator development process**

Our analytical models can be used to answer these questions which are of concern to design managers. According to literature review and field study, testing may be modeled as a continuous Non-Homogeneous Poisson Process (NHPP) (e.g. Serich, 2005; Lin et al., 2008; Love et al., 2008), or a discrete cyclic process (e.g. Ha and Porteus, 1995; Dahan and Mendelson, 2001; Erat and Kavadias, 2008). In this thesis, the continuous and discrete testing processes are examined separately, since the models and policies for these processes are different.

### 1.3.2 Approaches for DSM Sequencing Problem

To structure NPD processes, another key and challenging decision faced by the management is how to plan the sequence of design activities with iteration loops (Krishnan and Ulrich, 2001; Anderson and Joglekar, 2005). In recent years, there has been a growing interest in applying DSM for planning design activities (Browning and Ramasesh, 2007; Sharman and Yassine, 2007; Karniel and Reich, 2009). One important objective of planning is to find an activity sequence with minimum feedbacks. Since the problem is NP-complete, there is almost no chance to develop solution algorithms with a polynomial-time complexity (Li and Sun, 2006).

To solve large DSM sequencing problem, we follow previous decomposition methods (e.g. McCulley and Bloebaum, 1996; Rogers, 1996, 1999; Ahmadi et al., 2001), and extend this line of research by developing some rules for feedback reduction, and by proposing two new decomposition approaches. The proposed approaches outperform previous ones in their flexibility and well construction of sub-problems. We have also applied the proposed approaches to three real data sets, and show that applying our approaches result in better solutions with smaller feedbacks.

In some real world situations, the information dependencies among activities may be difficult to estimate accurately (Chen et al., 2004; Luh et al., 2009). To address this issue, we resort to fuzzy set theory to represent uncertain activity dependencies and present a fuzzy approach to DSM sequencing problem. To demonstrate its utility, the proposed approach has been applied to a data set published in Eppinger (2001).

## 1.4 Structure of the Thesis

As shown in Figure 1.4, this thesis focuses on two decision problems for structuring NPD processes: test scheduling and activity sequencing, and consists of eight chapters:

**Chapter 1: Introduction** presents the research motivation, research gaps, research scope and objectives, and finally the overall structure of this thesis.

**Chapter 2: Literature Review** provides a review of relevant literature. Based on the decision problems considered, we categorize previous literature into three groups:

test scheduling, overlapping policies, and project scheduling.

**Chapter 3: Optimal Testing Strategies in Overlapped Design Process** treats testing as a continuous NHPP, and presents an analytical model for scheduling tests in overlapped process. Analysis of the model yields several useful insights, which can be used to improve NPD processes where the testing set-up time is relatively small. The methodology is validated with a case study at a handset design company.

**Chapter 4: Scheduling Tests in N-stage Overlapped Design Process** deals with discrete cyclic testing process, and develops a model for determining optimal number of tests needed at each stage, together with the optimal overlapping policies, in N-stage overlapped process. The model yields several useful insights, which can be used to structure NPD processes where the testing set-up time is long. The methodology was applied to a refrigerator development at a consumer electronics company.

While Chapter 3 to 4 deal with the test scheduling problem, Chapter 5 to 7 are concerned with the activity sequencing problem.

**Chapter 5: A Decomposition Approach for Sequencing Design Activities** first introduces a 0-1 quadratic integer program for DSM sequencing problem. After that, we establish two simple rules for feedback reduction, and show that small-scale sequencing problem can be solved by a Branch-and-Bound method. A heuristic decomposition procedure is then presented to extend the Branch-and-Bound method to solve large-scale problems. To demonstrate its utility, the proposed solution strategy has been applied to three real data sets, and benchmarked with the solutions presented in previous studies.

**Chapter 6: A Novel Approach to Large-scale DSM Sequencing Problem**

further deals with DSM sequencing problem. Based on the results proved, a novel approach is presented for solving large-scale problems. Comparison of application results between the approach in this chapter and the one in Chapter 5 is also presented.

**Chapter 7: A fuzzy Approach to DSM Sequencing Problem** applies some fuzzy set theory to represent imprecise activity dependencies and presents a fuzzy approach to DSM sequencing problem. To illustrate its utility, the proposed approach is applied to the powertrain development at General Motors (Eppinger, 2001).

**Chapter 8: Conclusions and Future Study** gives a conclusion of this thesis and some possible future research topics.

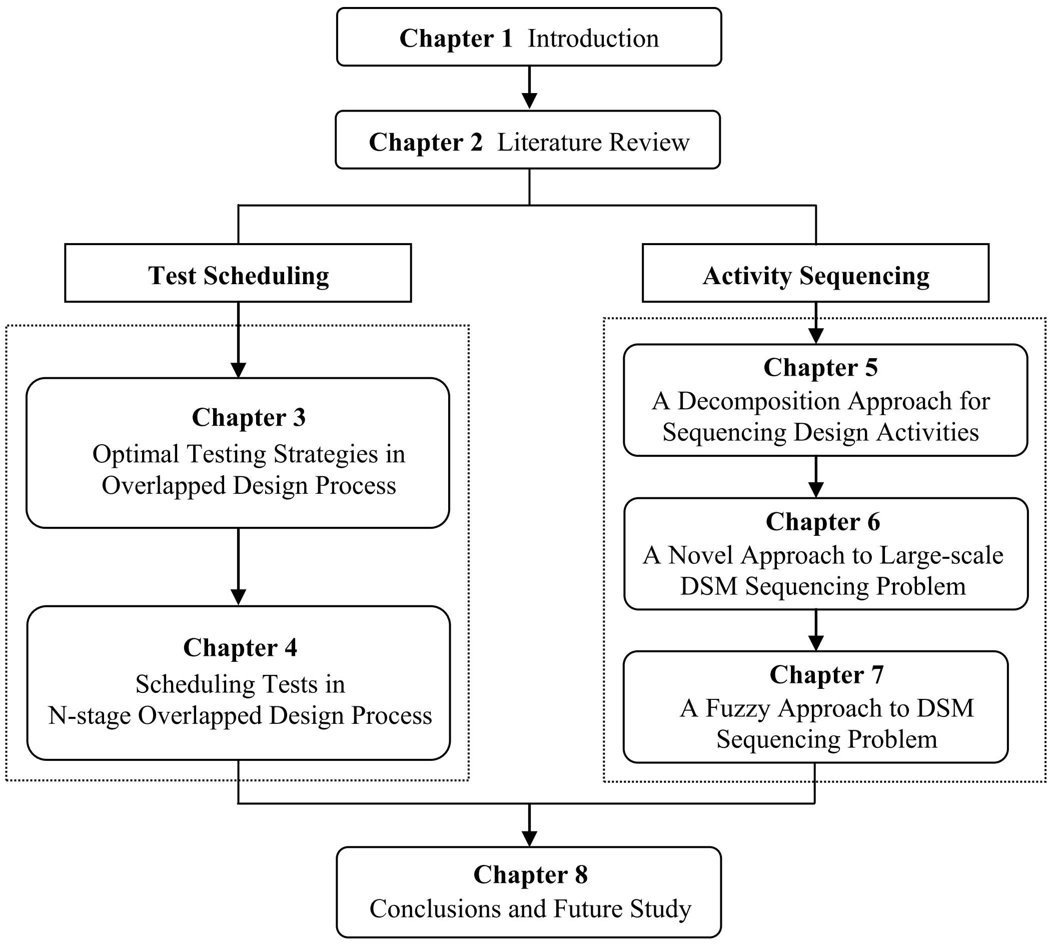

Chapter 1 Introduction
Chapter 2 Literature Review
Test Scheduling
Activity Sequencing
Chapter 3 Optimal Testing Strategies in Overlapped Design Process
Chapter 4 Scheduling Tests in N-stage Overlapped Design Process
Chapter 5 A Decomposition Approach for Sequencing Design Activities
Chapter 6 A Novel Approach to Large-scale DSM Sequencing Problem
Chapter 7 A Fuzzy Approach to DSM Sequencing Problem
Chapter 8 Conclusions and Future Study

**Figure 1.4 Structure of the thesis**

# CHAPTER 2

# LITERATURE REVIEW

NPD process modeling has received considerable attention over the last 15 years from both the academic community and practitioners (Roemer and Ahmadi, 2004; Shane and Ulrich, 2004; Chao et al., 2009). To model and structure NPD processes, decisions are often made about the test scheduling for project monitoring and control, the degree of overlapping and mechanisms for coordination, and the planned timing and sequence of design activities (Krishnan and Ulrich, 2001; Browning and Ramasesh, 2007). In this chapter, an extensive review of the relevant literature is presented. Based on the decisions considered, we categorize previous literature into three groups. Section 2.1 reviews the literature on test scheduling. Section 2.2 discusses previous studies on overlapping policies. Section 2.3 presents a review on different methods on project scheduling. Finally, Section 2.4 summarizes the concluding comments.

## 2.1 Test Scheduling

### 2.1.1 Empirical Studies

The importance of testing for successful NPD has been emphasized by many researchers. First, testing usually accounts for the majority of project completion time and development cost. For example, Shooman (1983), as well as Cusumano and Selby (1995), showed that testing activities can account for nearly half of total development effort. Thomke (2003) reported that project teams spent nearly 50% of their time on testing and related analysis. Second, for most product development projects, the initial outputs inevitably contain design problems, such as mismatches with customer needs, technical design faults, or issues regarding manufacturability and maintainability of the product (Thomke and Bell, 2001; Dahan and Hauser, 2002; Gil et al., 2008). Testing, which is a primary way to detect and resolve these problems, is central to product development (Loch et al., 2001; Thomke, 2007).

As reported by Loch et al. (2001), because testing is so central to NPD, a growing number of researchers have started to study testing strategies or test scheduling problem. Recent qualitative and empirical studies focused on the effect of "Front-Loading" on product development performance. Front-Loading refers to the recent emerging testing methodologies which allow an earlier detection of potential engineering problems. For example, Thomke (1998) studied the costs and benefits of such advanced testing methods as rapid prototyping and computer simulation. Dahan and Srinivasan (2000) observed that compared with the traditional paper-and-pencil testing methods, internet-based tests are more effective in measuring market potential, and lower in cost. Thomke and Fujimoto (2000) reported that the use of computer simulation tests allowed the Toyota Motor Corporation solving about 80% of all problems by stage two (overall of eight development stages), and thus resulted in about 30-40% reduction in development costs and lead time.

### 2.1.2 Test Scheduling Problem

A typical NPD process often involves a series of development stages, followed by testing activities performed to detect and remove design problems in each stage's outputs. It is known that undetected design problems of an upstream stage (e.g. concept development) will accumulate and proliferate to downstream stage (e.g.

product design). Thus, the outputs of an upstream stage need to be tested extensively before releasing them to downstream stage. Inadequate testing would allow design problems to propagate, and finally deteriorate the product quality. On the other hand, testing also incurs time and cost. Too much testing at one stage would impede the project's progress and increase development costs. Thus, how to optimally schedule various tests along the NPD process so as to maximize product development performance is an important decision problem (Krishnan and Ulrich, 2001; Thomke and Bell, 2001; Qian et al., 2009).

Some mathematical models have been developed to determine the optimal scheduling of tests and/or reviews for product development projects. We categorize them into two groups. The first group of studies, which is discussed in Section 2.1.2.1, focused on test scheduling problem at one development stage, while the second group of studies, which are discussed in Section 2.1.2.2, examined the test scheduling problem for multiple development stages.

#### 2.1.2.1 Mathematical Models for Single-stage Test Scheduling

Ha and Porteus (1995) studied the costs and benefits of design reviews for two overlapped design phases. In their work, frequent reviews enabled earlier detection of upstream flaws and concurrent execution of downstream phase, but would require additional time spent on the reviews. Given these trade-offs, they developed a model to decide the optimal timing and frequency of design reviews so as to minimize the project completion time. Their model was based on two main assumptions. First, no flaw would arise in the downstream phase. Second, the design reviews were perfect, in other words, each review could detect all the existing design flaws.

Dahan and Mendelson (2001) modeled the concept testing as a probabilistic search process and proposed an extreme-value model to determine the optimal number of tests and total budget for the concept development phase. Thomke and Bell (2001) developed a mathematical model to decide the optimal frequency, and fidelity of sequential testing activities. Their model was based on two main assumptions. First, the number of cumulated problems increased linearly with development time. Second, the cost of a test depended only on fidelity, where a test with higher fidelity would tend to uncover most currently detectable design problems. They showed optimal testing strategies should balance several tensions, including redesign cost, the cost of a test, and the correlation between sequential tests. A simple form of their model yielded an EOQ-like result: the optimal number of tests was the square root of the ratio of avoidable cost and the cost of a test.

Loch et al. (2001) developed a model to determine the optimal mix of parallel and serial testing strategies that would minimize the total testing costs. In their model, the design team gradually learned through sequential tests, and so sequential testing strategy would require smaller number of tests to be carried out than parallel testing strategy. However, sequential testing strategy had the disadvantage of proceeding more slowly than parallel testing. A dynamic programming model was then presented to address this trade-off. Recently, Erat and Kavadias (2008) extended the work of Loch et al. (2001) by considering the design space structure and the correlations among design performances.

Serich (2005) considered a three-phase project beginning with an optional prototyping phase, followed by a construction phase, and a rework phase. In their work, prototyping would reduce uncertainty and the resulting rework, but at the cost of additional time spent in prototyping. An analytical model was proposed to decide the optimal amount of time spent in prototype tests such that the overall project duration would be minimized.

Bartels and Zimmermann (2009) stated that in some industries, such as the automobile and the aircraft industry, the majority of testing costs were incurred by the final prototype testing stage since the construction of one experimental vehicle could be very expensive. At this testing stage, many different items of tests, such as the functional test and drop test, were conducted before launching the new product into market. They then introduced an approach to determine the optimal sequence of these tests such that the number of required experimental vehicles would be minimized.

Test scheduling problem has been studied extensively in software development literature, and a recent review can be found in Xie et al. (2007). For instances, Yamada et al. (1995) considered the optimal allocation of testing resources among software modules based on a NHPP. Hou et al. (1997) investigated the cost optimal release policy for software systems with scheduled delivery time under Hyper-Geometric distribution software reliability growth model with exponential or logistic learning factor. Xie and Yang (2001) investigated the problem of optimal allocating testing resources among software modules to maximize reliability of whole system. Dai et al. (2003) presented a genetic algorithm for multi-objective test resource allocation problem. Pham and Zhang (2003) developed an analytical model to determine the optimal testing stop rules so as to achieve the required reliability at minimal cost. Huang and Lyu (2005) studied the impact of software testing effort and efficiency on the cost for optimal release time. Tamura and Yamada (2006) examined optimal software release problems by using a flexible stochastic differential equation model based on the reusable rate in the system testing phase of the distributed development environment. Kapur et al. (2007) discussed the testing resource allocation problem among modules to maximize the total number of faults removed from software consisting of several independent modules. Yang et al. (2008) proposed a risk-control approach to examine the uncertainty in software cost and its impact on testing strategies and optimal software release time.

The above models have clearly shed light on the analysis of test scheduling problem. However, they focus on the testing policies at one development stage and do not take into account the multi-stage nature of testing process. It is known that the testing activities at different development stages are interacted and should be adjust coordinately. For example, in refrigerator development, engineers can do one round of prototype test at concept development stage, or many rounds of tests. Spending more time in prototype tests of concept development stage will reduce the potential problems in detail design. Therefore, the project completion time may be reduced. Then, how to balance the testing activities in concept development, detail design, and process design? It should be valuable to investigate it in detail.

#### 2.1.2.2 Mathematical Models for Multi-stage Test Scheduling

An important contribution in modeling multi-stage testing for product development projects is due to Cooper (1980, 1993a, 1993b, 1993c). Based on his experience as a consultant, he distinguished between the initial design of development stages and testing. In the initial design, development stages were performed at different but usually less than perfect quality. In other words, the initial outputs of development stages, such as the product specifications and bill of materials, contained design faults and would to be reworked when these design faults were identified by the following testing activities. Testing activities were not perfect and could not find all design faults. Therefore, design faults would propagate across development stages, and resulting in downstream tasks done on these design faults. Finally, when tests at the downstream stage identified these upstream design faults, not only the design faults need to be corrected, but also the relevant downstream tasks need to be reworked. Such a fault discovery delay could therefore substantially increase the cost of rectifying errors and project completion time. Except for the design faults, rework may also caused by customer changes. He then defined completion quality as the proportion of work being done which will not require rework, and testing quality as the percentage of design faults identified in the testing process. Based on these definitions, he simulated the major development stages of shipbuilding operation using system dynamics approach, and concluded that testing quality at earlier stages of project life increased testing cost, but reduced project completion time considerably and increased the probability of meeting the customer's specifications.

Ford and Sterman (1998, 2003a, 2003b), as well as Joglekar and Ford (2005), extended the works of cooper (1993a, 1993b, 1993c) by including process structure and resource allocation in their system dynamics models. Williams et al. (2003) presented a system dynamics model to structure the delay and disruption claims. Based on system dynamics approach, William (2005) analyzed a number of failed projects to explore why the common project-management discourse could give rise to failed projects. They found that for projects that were complex, uncertain, and time-limited, conventional methods might be inappropriate, and aspects of newer methodologies in which the project "emerges" rather than being fully preplanned would be more appropriate. More recently, Love et al. (2008) examined how and why design-induced rework occurred in a commercial construction project since design-induced rework could contribute up to 70% of the total amount of rework. In their work, the underlying behavioral dynamics that contributed to design errors, such as the experience and skill of engineers, schedule pressure and information technology, were modeled and simulated using system dynamics approach. In Love et al. (2009), they further present a system dynamics model to examine the underlying factors that contribute to omission errors in construction and engineering projects. Here omission errors are resulted from pathogens within a system (such as time pressure, understaffing, fatigue, and inexperience) that translate into error provoking conditions within the firm and project.

Lin et al. (2008) complemented previous system dynamics models by including overlapping in their model. They explicitly defined and modeled two types of rework: *Rework due to Development Errors*, which referred to rework or rectification of design errors, and *Rework Due to Corruption*, which referred to rework or rectification of relevant downstream tasks due to the change of tasks in an upstream stage. Based on these concepts, they proposed a Dynamic Development Process Model for managing overlapped iterative product development, and validated the model with an in-depth case study at a handset design company.

The above system dynamics simulation models have greatly advanced our understanding on the multi-stage testing process. Given a set of testing strategies, these models can be used to compare the solutions and identify which one is best. However, it is often impossible to tell how far the current solution is from optimality (Sterman, 2004; Cho and Eppinger, 2005). Moreover, for problems with continuous decision variables, it’s unlikely to get a good solution quickly and efficiently.

In literature on analytical approaches, Ahmadi and Wang (1999) explicitly modeled the multi-stage review process, and examined how to optimally schedule reviews and engineering resources along the design process so as to achieve the required level of process confidence at minimal development cost. While their work is useful for managing the sequential process, the solutions and insights they get may not be applicable to overlapped process where downstream stages start before the completion of upstream stages.

Kogan and Raz (2002) examined how to optimally schedule the inspection activities in an N-stage system so as to minimize the sum of inspection costs and penalty costs caused by undetected defects. An efficient algorithm was proposed to solve the problem. However, their work assumes that all stages start and finish simultaneously, which is less common in practice.

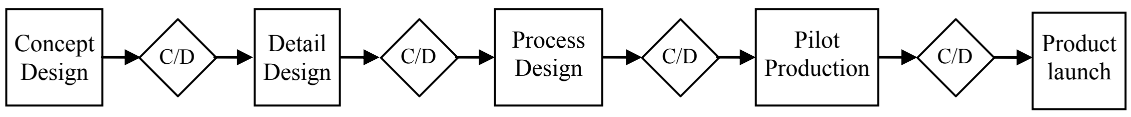
As far as we know, no analytical model exists for scheduling tests in overlapped NPD process. Meanwhile, overlapping development stages has become a common mode of product development (Terwiesch et al., 2002; Loch and Terwiesch, 2005; Yassine et al., 2008), and the testing strategies combined with overlapping policies may affect project performance differently compared with testing strategies in the sequential process. Therefore, it is meaningful and worthwhile to investigate the testing strategies in overlapped NPD process.

In modeling testing processes, one stream of existing literature (e.g., Cooper, 1993a, 1993b, 1993c; Yamada et al., 1995; Kogan and Raz, 2002; Pham and Zhang, 2003; Serich, 2005; Lin et al., 2008; Love et al., 2008; Love et al., 2009) modeled testing as a continuous NHPP process of discovering and solving design problems. It is justified that when design problems arise from many components or modules, the set-up time of a test is relatively small and can be ignored such that the rate of discovering and solving design faults is approximately continuous. On the other hand, the second group of studies postulated that each time a test takes place, a certain amount of set-up time (such as the time to get organized for the test, to construct prototypes and to prepare documents) is required, and modeled testing as a discrete cyclic process (see e.g. Ha and Porteus, 1995; Dahan and Mendelson, 2001; Loch et al., 2001; Erat and Kavadias, 2008). This stream of literature echoed previous empirical studies (e.g. Thomke, 1998; Thmoke and Fujimoto, 2000), which showed that the execution of testing often involved a three-step iterative cycle: (1) build virtual or physical prototypes that embody the key aspects of the design; (2) test the prototypes to identify design problems; and (3) modify the design to remove these design problems.

## 2.2 Overlapping Policies

A typical NPD process can be viewed as consisting of four consecutive stages: concept design, detail design, process design, and pilot production (Haberle et al., 2000; Chakravarty, 2003; Yan et al., 2003; Browning, 2009). Generally, concept design stage defines the product's concept, architecture and specifications based on market research of customer preferences. Detail design stage involves the determination of design parameters and detailed design of components. Process design stage constitutes the design of tools, facilities, equipment, and so on. Pilot production is the stage where the overall product design is realized as physical products with further testing implemented to improve the overall quality of the product.

As shown in Figure 2.1, traditional phase-milestone NPD processes are sequential, with check and decision points placed at the end of each stage (Cooper, 1994; MacCormack et al., 2001; Nair and Boulton, 2008). Moreover, the process is highly functionally segregated, in other words, different functions are responsible for different stages, and communication between the functions are only occurred at the end of each stage (Cooper, 1994; Bhuiyan, 2001; Carrillo and Franza, 2006). Clark and Fujimoto (1991) stated that such process would be appropriate "…when markets were relatively stable, product life cycles were long, and customers concerned most with technical performance." However, such traditional paradigm would increase friction among different function groups, and lead to bottlenecks in the flow of information through the NPD processes (Clark and Fujimoto, 1991; Swink et al., 1996; Browning and Health, 2009), which would further increase the project completion time and consume additional resources (Patrashkova-Volzdoska et al., 2003; Bhuiyan et al., 2004; Sosa et al., 2007a).



C/D: Checking & Decision

**Figure 2.1 Traditional phase-milestone NPD process**

Over the last two decades, intense competition, rapidly evolving technologies, changing customer needs, and shorter product life cycles force many firms to develop lower cost, higher quality products at a rapid pace (Eppinger et al., 1994; Wagner and Hoegl, 2006; Cooper and Edgett, 2008). Many corporations have responded to these challenges through using Concurrent Engineering (CE) approach. Overlapping development stages and cross-functional development teams are two of the most important components of CE (Clark and Fujimoto, 1991; Atuahene-Gima and Evangelista, 2000; Cooper and Kleinschmidt, 2007).

Overlapping refers to the partial parallel execution of development stages where the downstream stage starts before the completion of the upstream stage. Through executing multiple stages simultaneously rather than sequentially, the project completion time may be reduced in overlapped NPD process (Datar et al., 1997; Gerk and Qassim, 2008). In recent years, overlapping has become a common mode of product development as a result of increasing importance of time-to-market (Terwiesch et al., 2002; Yan et al., 2002; Yassine et al., 2008).

Although large reduction in project completion time may be achieved by applying overlapping approach (Smith and Reinertsen, 1998; Sobek et al., 1999; Helms, 2004), empirical studies also show that overlapping is not applicable to all NPD projects (Eisenhardt and Tabrizi, 1995; Liker et al., 1996; Gil et al., 2008). For example, based on the empirical study of 140 development projects in the electronics industries, Terwiesch and Loch (1999) concluded that overlapping was effective only if uncertainty resolution was fast. Because overlapping requires that downstream stages start on preliminary information, rework is often necessary to accommodate upstream design changes. If the uncertainty or the dependency between development stages is high, most of downstream tasks done on upstream preliminary information need to be reworked, which makes overlapping unfavorable (Krishnan et al., 1997; Helms, 2002; Minderhoud and Fraser, 2005; Lin et al., 2010). For instance, Terwiesch et al. (2002) showed that the downstream rework caused by overlapping frequently consumed as much as 50% of total engineering capacity in their case study company. Based on survey data from a sample of 120 projects in healthcare and telecommunications, Mitchell and Nault (2007) indicated that project delay was primarily due to downstream rework and downstream delay. Therefore, a key trade-off involved in overlapping development stages is time reduction versus additional effort for downstream rework.

### 2.2.1 Mathematical Models

Some mathematical models have examined the key trade-off involved in overlapping development stages and its associated drivers. Krishnan et al. (1997) developed an integer program to determine the optimal number of information transfer between two consecutive development stages, as well as the start time of downstream rework, such that project completion time would be minimized. The authors proposed that the optimal overlapping policies should be determined by two properties of the NPD process, "upstream evolution" and "downstream sensitivity", where "upstream evolution" denoted the speed at which upstream information narrows from an interval value to a final solution, and "downstream sensitivity" referred to the expected time needed for the downstream stage to incorporate upstream design changes. This principle was further developed by Loch and Terwiesch (1998), where they proposed that the optimal levels of overlapping and communication should be decided by the arrival rate of upstream design modifications, the impact of each modification (i.e. the percentage of downstream tasks would be affected by one upstream design change), and the downstream progress, i.e. the number of downstream tasks completed when the design change arrived.

Since then, a number of independent researchers have studied the optimal overlapping policies for projects with different characteristics. For example, Yassine et al. (1999b) developed a probabilistic model to determine the optimal overlapping policy for a set of activities with given information structure. Cantamessa and Villa (2000) proposed an analytical model to determine the optimal allocation of product and process designers' time with the objective of minimizing the overall design effort.

Roemer et al. (2000) addressed the time-cost trade-off in overlapped product development process and introduced an algorithm to determine the optimal amount of overlapping. Chakravarty (2001) examined the optimal overlapping policies for three overlapping modes and analyzed the impact of parameter values on overlapping decisions. Joglekar et al. (2001) proposed a performance generation model to determine the optimal overlapping strategies with the goal of maximizing project performance with deadline constraints. Chakravarty (2003) developed two approaches to determine the optimal start time of downstream rework for continuous and discrete upstream design changes, respectively. Yassine et al. (2003) developed a model to derive conditions under which churn was observed as an unintended consequence of information hiding due to local and system task decomposition. Bhuiyan et al. (2004) proposed a discrete event simulation model to study the impact of overlapping and functional interaction on project performance which was measured by total person-days.

Recently, Roemer and Ahmadi (2004) explicitly studied the interactions between overlapping and crashing, which are two common methods for reducing project completion time, and provided general guidelines for optimal overlapping and crashing policies. Gerk and Qassim (2008) extended the work of Roemer and Ahmadi (2004) by including another method of accelerating product development, substitution. A mixed-integer non-linear programming model was then presented for simultaneously determining the optimal crashing, overlapping, and substitution policies. Wang and Yan (2005) focused on the optimization of the overlap degree between an upstream stage and downstream stages with the goal of minimizing total cost of delay of project completion time and design revision workloads. The authors proved that the total cost is convex with respect to the overlap degree between design activities. Jun et al. (2005) classified product development processes into different patterns, and developed an approach to estimate the project completion time. Yassine et al. (2008) developed optimal decision rules to determine whether to incorporate a piece of information that just arrived (i.e. became available) or wait longer. Lin et al. (2009) presented an analytical model to derive the optimal amount of overlapping and functional interaction by balancing the positive and negative effects of overlapping and functional interaction. Lin et al. (2010) extended the work of Loch and Terwiesch (1998), and explicitly studied the time-cost trade-off involved in concurrent design process in order to derive optimal overlapping and communication policies.

The results of above overlapping models are insightful in many respects. However, all of them assume that testing strategies are predetermined. Analytical models are needed to combine these two decisions (i.e. test scheduling and overlapping degrees) into one modeling framework since they are interacted.

## 2.3 Project Scheduling

As mentioned earlier, in determining the testing and overlapping policies, the NPD process is generally viewed as consisting of a series of development stages. When the NPD process is broken down into development stages, it's no need to consider the sequencing problem since the execution sequence of these stages (such as the concept design and the detail design) is fixed and will not change. However, a development stage may further be broken down into smaller activities, and a complex NPD process may involves hundreds of decision-making activities with cyclic information flows among them (Eppinger et al., 1994; Chen et al., 2004; Meixell et al., 2006). Therefore, if the NPD process is further broken down into smaller design activities, then a key and challenging issue often faced by the management is how to plan the time and sequence of these activities (Krishnan and Ulrich, 2001; Anderson and Joglekar, 2005; Browning and Ramasesh, 2007). Over the years, various models for scheduling design activities have been developed. We classify these models from the methodological aspect: network-based scheduling techniques (discussed in Section 2.3.1), simulation models (discussed in Section 2.3.2), and design structure matrix (discussed in Section 2.3.3).

## 2.3.1 Network-based Scheduling Techniques

Network-based scheduling techniques, such CPM and PERT, utilize activity duration estimates and precedence relationships representing the network of activities (Badiru, 1993; Golenko-Ginzburg and Gonik, 1996). Figure 2.2 shows a simple example of such network diagram for CPM schedule management.

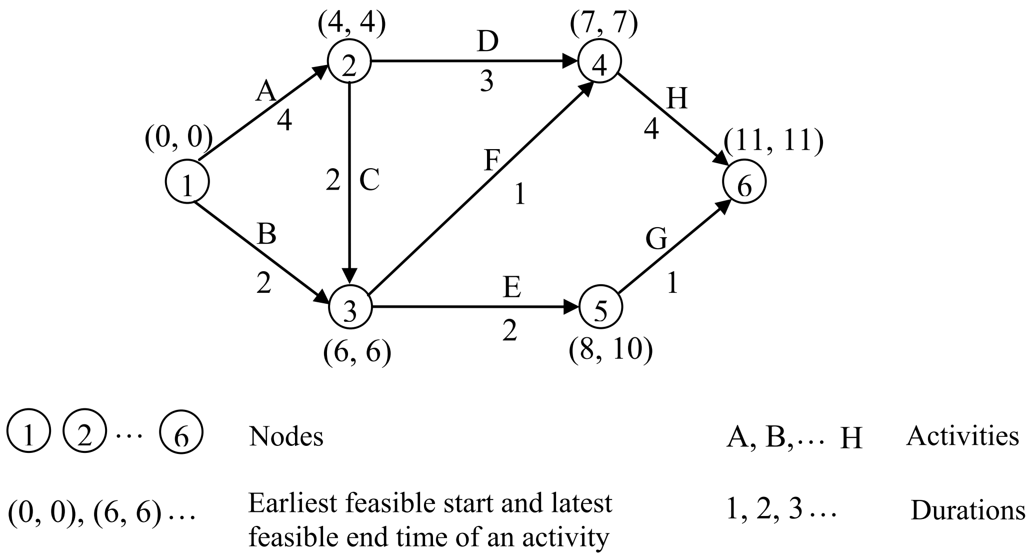


**Figure 2.2 A network diagram for CPM schedule management**

CPM enables the identification of a project's critical path, i.e. the sequence of activities whose combined durations define the minimum project completion time, as well as earliest and latest possible start and finish times of all activities. Moreover, CPM provides some tools for studying the trade-offs of different performance measures, such as the time-cost trade-off.

PERT complements CPM by incorporating the effect of project uncertainty into the estimates of activity duration. Three estimates, i.e. most likely, optimistic and pessimistic, are used to describe the variability of activity durations. Based on these parameters, the probabilities of a project meeting specific schedule objectives can be obtained. The incorporation of duration uncertainty makes PERT more valuable in managing projects with uncertainty.

However, for most development projects, the delay is usually caused by iteration instead of uncertain activity duration. Like CPM, PERT cannot explicitly represent the dynamic interaction between design activities and the iteration, limiting their capability of scheduling for NPD projects (Ahmadi et al., 2001; Denker et al., 2001; Kang and Hong, 2009).

It is known that the outputs of decision-making activities, such as engineering drawings, specifications and bill of materials, are often unstable and inaccurate, and need to be reworked when downstream activities detect some faults in the original design (Loch and Terwiesch, 2005; Love et al., 2008). Second, downstream activities may be repeated when modified information is passed along from upstream activities (Smith and Eppinger, 1997a; Lin et al., 2008). Thus, cyclic information flows among activities are quite common in NPD process and PERT/CPM could not deal with such loops effectively (Eppinger, 2001). Moreover, in PERT/CPM, the underlying precedence relationships among activities are assumed known and unchanged, but in many complex development projects, clear precedence constraints do not exist and are rarely known in advance (Ahmadi et al., 2001; Jun and Suh, 2008).

CPM and PERT were initially developed to control schedule, and later expanded to handle costs and resource allocation (see e.g. Mika et al., 2005; Kolisch and Hartmann, 2006; Lancaster and Ozbayrak, 2007; He and Xu, 2008; Waligora, 2008). Although an extension of PERT models, known as Graphical Evaluation and Review Technique (GERT), enables simulation-based analysis of activity networks with iteration loops, direct analysis of any but a simple GERT network is difficult (Smith and Eppinger, 1997b; Cho and Eppinger, 2005; Browning and Ramasesh, 2007).

### 2.3.2 Discrete Event Simulation Models

To address the shortfalls of network-based scheduling techniques, one stream of research utilizes simulation to model iteration and explores the linkage between activity sequences and project performance. Most commonly used method is discrete event simulation. For example, Carrascosa et al. (1998) presented a model to estimate project completion time for different activity sequences and overlapping degrees using concepts of probability of change and impact. Browning and Eppinger (2002) examined the effects of varying process architecture by simulating NPD process as a network of activities that exchange deliverables. The model outputs sample cost and schedule outcome distributions. Each distribution is used with a target and an impact function to determine a risk factor. Alternative process architectures can then be compared to reveal opportunities to trade cost and schedule risk.

More recently, Gil et al. (2004) simulated the concept development process for semiconductor fabrication facilities, and found that some decision-making postponement could help increase the predictability of concept development duration and reduce resources spent in design without increasing the risk of exceeding project deadlines. Cho and Eppinger (2005) extended the work of Browning and Eppinger

(2002) by addressing resource constraints. Kouskouras and Georgiou (2007) presented a discrete event simulation model for managing software projects.

### 2.3.3 Design Structure Matrix

To address the shortfalls of CPM and PERT, another known method is DSM. As illustrated in Figure 2.3, there are three possible sequences of two design activities based on the information dependencies between them (Eppinger et al., 1994; Carrascosa et al., 1998): (1) if there is a unidirectional information transfer from A to B, then activity B is dependent on activity A and would be performed after the completion of A; (2) if there is no information exchange between activity A and B, then the two activities are independent and can be executed in parallel; finally, (3) if there exist cyclic information flows, i.e. activity A needs information from activity B, and also activity B requires the output of activity A, then the two activities are interdependent or coupled.

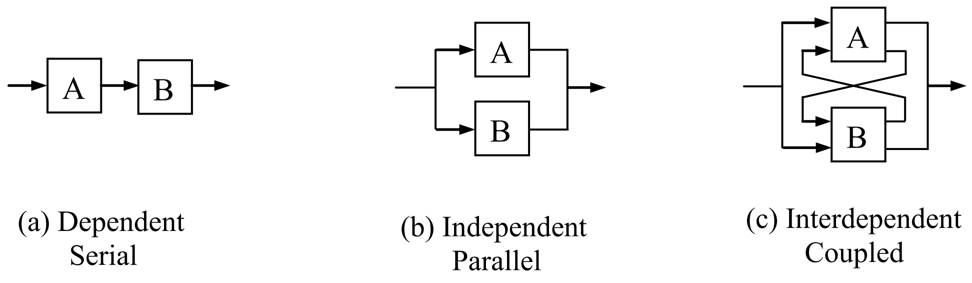


**Figure 2.3 Three possible sequences for two activities (Eppinger et al., 1994)**

The three types of information dependencies among design activities, as well as iterative nature of product development can be effectively addressed by using DSM. As shown in Figure 2.4, the basic DSM is a binary matrix representation of a project with elements denoting individual design activities and off-diagonal marks representing the information dependencies among these activities (Eppinger et al., 1994; Browning, 2001). Along each row, the off-diagonal marks indicate all of the activities whose output information is required to perform the activity corresponding to that row; reading down each column reveals that which other activities receive its output (Yassine et al., 1999a; Maheswari et al., 2006). When activities are executed in the order listed from top to bottom, sub-diagonal marks represent an input from upstream activities to downstream activities, and super-diagonal marks denote a feedback from downstream activities to upstream activities (Browning and Eppinger, 2002; Karniel and Reich, 2009). As such, DSM provides a compact representation of a complex system by showing information dependencies in a square matrix, as well as a useful tool for describing and investigating iteration (Denker et al., 2001; Cho and Eppinger, 2005).

It is known that iteration is a major driver for lengthy and costly product development. To improve NPD processes, the DSM approach suggests re-sequencing the design activities so as to minimize iterations, i.e. to get the DSM into a lower-triangular form as possible (Eppinger et al., 1994). If the design activities can be ordered such that no super-diagonal marks exist, then no iteration remains, and PERT/CPM are still applicable (Eppinger, 2001). However, this seldom occurs because of the complexities in product development processes, in other words, cyclic flows of information exist and PERT/CPM could not deal with such loops effectively (Eppinger, 2001). Moreover, DSM differs from network-based scheduling techniques in that it focuses on representing information flows rather than work flows (Yassine et al., 1999a; Eppinger, 2001). As reported by Eppinger (2001), conventional tools answer the question: “What other activities must be completed before I begin this one?” However, the DSM approach addresses the question: “What information do I need from other activities before I can complete this one?”

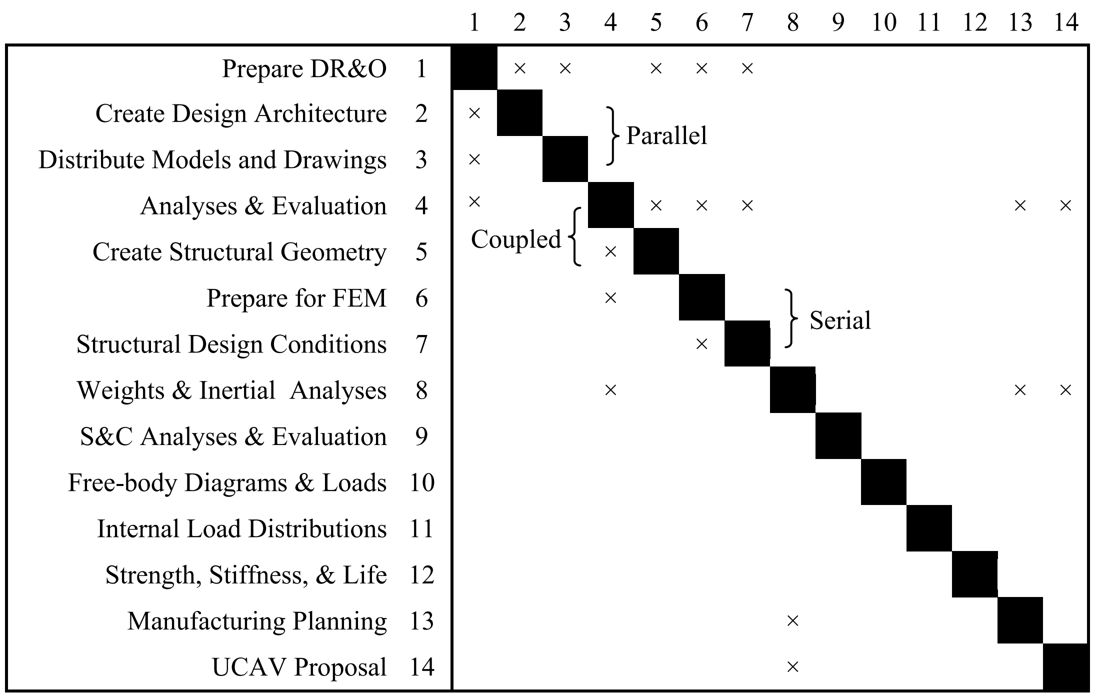

1 2 3 4 5 6 7 8 9 10 11 12 13 14
Prepare DR&O 1
Create Design Architecture 2
Distribute Models and Drawings 3
Analyses & Evaluation 4
Create Structural Geometry 5
Prepare for FEM 6
Structural Design Conditions 7
Weights & Inertial Analyses 8
S&C Analyses & Evaluation 9
Free-body Diagrams & Loads 10
Internal Load Distributions 11
Strength, Stiffness, & Life 12
Manufacturing Planning 13
UCAV Proposal 14
Parallel
Coupled
Serial

**Figure 2.4 DSM representation of UCAV preliminary design process (Adapted from Browning and Eppinger, 2002)**

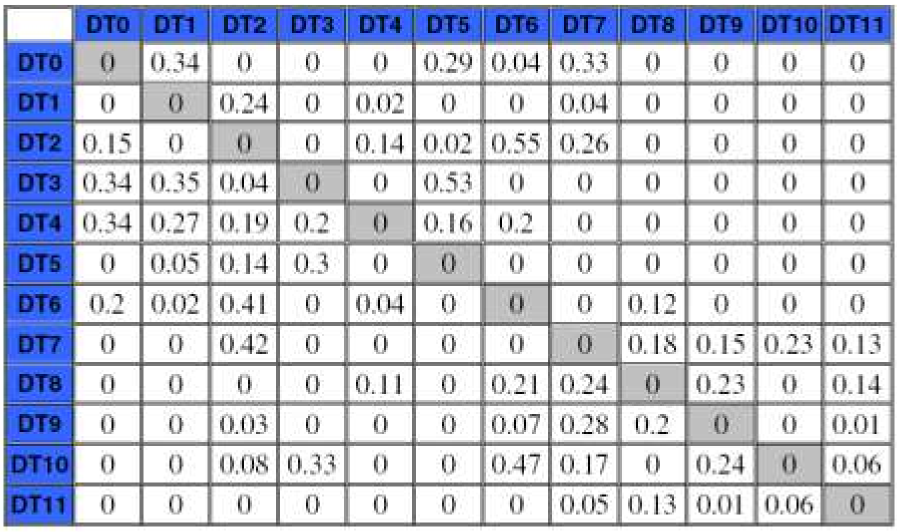
The DSM approach was first introduced by Steward (1981). Eppinger et al. (1994) extended Steward's work by explicitly including numerical measures of the degree of activity dependence, and proposed the Numerical DSM (NDSM). Compared to DSM, where "X" mark and empty cells signify the existence and absence of information dependencies among activities, NDSM could provide more detailed information on the relationships among design activities, and could be used to further improve NPD processes (Chen and Li, 2003; Chen et al., 2004). Eppinger et al. (1994) proposed that three measures could be used to capture the dependency between activities A and B, given that activity B depends on the output information from activity A:

(1) Variability: If the output information from activity A will change significantly (slightly), then the dependency tends to be strong (weak).

(2) Predictability: If the change of output information from activity A is unpredictable (predictable), then the dependency tends to be strong (weak).

(3) Sensitivity: If a slight (substantial) change of output information of activity A will lead to a large (small) magnitude of design iteration in B, then the dependency tends to be strong (weak).

Based on the three measures, activity dependencies can then be quantified through interviewing the engineers and managers familiar with the system and/or via surveying relevant documentation (Eppinger, 2001; Chen et al., 2004). Figure 2.5 shows an example of NDSM for the burn-in system (Chen et al., 2004), which consists of twelve activities, numbered from DT0 to DT11.



| | DT0 | DT1 | DT2 | DT3 | DT4 | DT5 | DT6 | DT7 | DT8 | DT9 | DT10 | DT11 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| DT0 | 0 | 0.34 | 0 | 0 | 0 | 0.29 | 0.04 | 0.33 | 0 | 0 | 0 | 0 |
| DT1 | 0 | 0 | 0.24 | 0 | 0.02 | 0 | 0 | 0.04 | 0 | 0 | 0 | 0 |
| DT2 | 0.15 | 0 | 0 | 0 | 0.14 | 0.02 | 0.55 | 0.26 | 0 | 0 | 0 | 0 |
| DT3 | 0.34 | 0.35 | 0.04 | 0 | 0 | 0.53 | 0 | 0 | 0 | 0 | 0 | 0 |
| DT4 | 0.34 | 0.27 | 0.19 | 0.2 | 0 | 0.16 | 0.2 | 0 | 0 | 0 | 0 | 0 |
| DT5 | 0 | 0.05 | 0.14 | 0.3 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| DT6 | 0.2 | 0.02 | 0.41 | 0 | 0.04 | 0 | 0 | 0 | 0.12 | 0 | 0 | 0 |
| DT7 | 0 | 0 | 0.42 | 0 | 0 | 0 | 0 | 0 | 0.18 | 0.15 | 0.23 | 0.13 |
| DT8 | 0 | 0 | 0 | 0 | 0.11 | 0 | 0.21 | 0.24 | 0 | 0.23 | 0 | 0.14 |
| DT9 | 0 | 0 | 0.03 | 0 | 0 | 0 | 0.07 | 0.28 | 0.2 | 0 | 0 | 0.01 |
| DT10 | 0 | 0 | 0.08 | 0.33 | 0 | 0 | 0.47 | 0.17 | 0 | 0.24 | 0 | 0.06 |
| DT11 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0.05 | 0.13 | 0.01 | 0.06 | 0 |

**Figure 2.5 NDSM for the burn-in system (from Chen et al., 2004)**

Over the years, several other methods have also been reported on how to reliably quantify activity dependencies. For instance, Carrascosa et al., (1998), as well as Browning and Eppinger (2002), Zhang et al. (2006), proposed that the dependency of activity *j* on *i* could be derived by multiplying the probability of change in activity *i* and the impact of that change on *j*. Chen and Li (2003) applied the analytic hierarchy process (AHP) to quantify activity dependencies, and showed that the clustering performance of using numerical DSM is better than that of using binary DSM. AHP allows diverse and often incommensurable elements to be compared to one another in a rational and consistent way, and has been demonstrated to be a suitable method for the selection of functionally most appropriate components of technical systems. Chen and Li (2003) reported that “In AHP, the hierarchy usually contains three major levels, such as the overall objective, criteria or sub-goals, and decision alternatives. Pairwise comparisons are made to determine the relative importance of the elements in each level. Relative priorities for decision alternatives can be evaluated under each specific criterion. The AHP converts these evaluations to numerical values that can be processed and compared over the entire range of the problem.” Chen et al. (2004) suggested a way of combining quality function deployment (QFD) and DSM in product design process, and proposed a systematic approach to quantify activity dependencies. Yassine (2007) developed a methodology that allowed a practical estimation of activity dependencies.

The advantages of DSM/NDSM have led to many successful applications in managing product development projects. For example, Osborne (1993) applied iteration maps and DSM to describe product development at Intel in terms of project completion time. Osborne’s work demonstrates the need for further investigation on the impact of activity dependencies on project completion time. Kusiak et al. (1994) presented a detailed reengineering approach based on models of DSM and Integration DEFinition (IDEF). Extensions of the work of Kusiak et al. (1994) can be found in Belhe and Kusiak (1996), Zakarian and Kusiak (2001), and Chin et al. (2006). Kusiak et al. (1995) described six patterns of design processes and presented a qualitative analysis approach for improving design processes. Morelli et al. (1995) used DSM to map and predict information flows among activities. Eppinger (2001) reported several successful applications of DSM for managing development projects. Clarkson et al.

(2004) used NDSM to map and predict the risk of change propagation for Westland Helicopters of rotorcraft design. Lee and Suh (2006) developed a workflow structuring method for identifying the reengineering issue and for transforming the complex design process into a well-structured workflow, based on DSM approach. Chen and Huang (2007) applied DSM approach to supply chain management. Tang et al. (2009) presented a tool for enhancing the axiomatic design method with DSM approach.

Recently, there is a growing interest in applying DSM/NDSM for identifying team arrangements (see e.g. Chen and Li, 2003; Batallas and Yassine, 2006; Yu et al., 2007; Amrit and van Hillegersberg, 2008; Collins et al., 2009; Karimian and Herrmann, 2009), and for product architecture decisions of developing modular products and product families (See e.g. Baldwin and Clark, 2000; Loch et al., 2003; Chen and Li, 2005; Fixson, 2005; Helo, 2006; Lopes and Bajracharya, 2006; MacCormack et al., 2006; Veenstra et al., 2006; De Weerd-Nederhof et al., 2007; Sosa et al., 2007b; Sosa, 2008; Zhuo et al., 2008; Bashir et al., 2009; Voss and Hsuan, 2009). Reviews of general areas of DSM application can be found in Browning (2001), Karniel and Reich (2009).

In project scheduling, DSM has spawned many research efforts on sequencing design activities (Eppinger et al., 1994; Browning and Ramasesh, 2007). In most of previous studies, the objective of sequencing is to find a sequence of activities that minimize feedbacks (Meier et al., 2007; Qian and Goh, 2007; Lancaster and Cheng, 2008). There are several reasons why this is meaningful and important. First, feedbacks determine where iteration loops occur (Eppinger et al., 1994; McCulley and Bloebaum, 1996). Studies have shown that iteration is a major driver for lengthy and costly product development (Terwiesch et al., 2002; Mihm et al., 2003; Love et al., 2008). Thus, minimizing feedbacks is a good approximation for concurrently reducing project completion time and development costs (Ahmadi et al., 2001; Meier et al., 2007; Collins et al., 2009). For example, Ahmadi et al. (2001) showed that the average errors resulting from the surrogate objective function of minimizing feedbacks was around 2.6% and the maximum error was less than 8%, over 540 problems solved. Second, as indicated by McCulley and Bloebaum (1996), in practice, especially for today's distributed product development, "the design managers are likely to desire choosing an evaluation procedure which requires the least amount of information to be guessed initially. Each feedback corresponds to one or more guesses, since the feedback input will be required by upstream activities before it is exactly known. Therefore, minimizing feedbacks will also minimize the amount of information required to be supplied as initial guesses".

Except for minimizing feedbacks, some other considerations have also been incorporated in sequencing design activities in a DSM, which are summarized as follows. Denote:

| | |
|---|---|
| $n$ | Total number of activities in a DSM |
| $i, j, k$ | Indices for activities |
| $d_{i,j}$ | Degree of information dependency of activity $i$ on $j$ ($d_{i,j} \geq 0$) |
| $f$, $c$ | The number of feedbacks and crossovers, respectively |
| $w_f$, $w_c$ | Weights for the number of feedbacks and crossovers, respectively |
| $w_T$, $w_C$ | Weights for project completion time and development cost, respectively |
| $L$, $D$, $U$ | The respective lower triangular, diagonal, and upper triangular matrices from Gaussian elimination |

| | |
|---|---|
| $b$ | The column vector which contain the normal activity duration |
| $f_{i,j}$ | The number of iterations in $j$ caused by activity $i$ |
| $b_k$ | Normal duration of activity $k$ |

Type I. Minimize feedbacks:

$$Min \sum_{i=1}^{n} \sum_{j=i+1}^{n} d_{i,j}$$

Type II. Minimize feedback lengths:

$$Min \sum_{i=1}^{n} \sum_{j=i+1}^{n} (j-i) d_{i,j}$$

Type III. Minimize weighted number of feedbacks and crossovers:

$$Min\ w_f f + w_c c$$

Type IV. Minimize weighted feedbacks, crossovers, time and cost:

$$Min\ w_f \cdot f + w_c \cdot c + w_T \cdot Time + w_C \cdot Cost$$

Type V. Minimize project completion time:

$$Min \sum_{i=1}^{n} x'_i$$

$$x' = D^{-1} L^{-1} b$$

Type VI. Minimize iteration time:

$$Min \sum_{i=1}^{n} \sum_{j=i+1}^{n} \left[ f_{i,j} \sum_{k=i}^{j} b_k \right]$$

Note that above objectives are all approximations for the actual goals of decreasing the NPD processes' time, cost and risk (Meier et al., 2007; Karniel and Reich, 2009). Based on the objective and solution approach used, Table 2.1 lists some activity sequencing models.

**Table 2.1 Comparison of some activity sequencing models**

| References | Objective | Solution Approach |
|---|---|---|
| Steward (1981) | Type I | Partition rules |
| Kusiak and Wang (1993) | Type I | Partition rules |
| Altus et al. (1996) | Type II | GA |
| McCulley and Bloebaum (1996) | Type III | Decomposition rules + GA |
| Rogers (1996, 1999) | Type IV | Decomposition rules + GA |
| Smith and Eppinger (1997a) | Type V | Enumeration + Heuristic |
| Ahmadi et al. (2001) | Type I | Block Decomposition + Branch-and-Bound |
| Whitfield et al. (2003) | Type I and II | GA |
| Abdelsalam and Bao (2006, 2007) | Type VI | Simulated Annealing |
| Banerjee et al. (2007) | Type V | Heuristic |
| Meier et al. (2007) | Type II | GA |
| Lancaster and Cheng (2008) | Type II | Evolutionary algorithm |
| Luh et al. (2009) | Type I | Heuristic |

Several independent researchers have reported that the DSM sequencing problem is NP-complete (e.g. McCulley and Bloebaum, 1996; Ahmadi et al., 2001; Meier et al., 2007). To solve large-scale activity sequencing problem, one stream of literature focused on developing heuristics and in particular meta-heuristic methods. For example, Altus et al. (1996) proposed a GA based model for organizing activities into a "satisfying" order such that the "total length of feedbacks" in the system could be minimized. Based on the assumption of sequential iteration, Smith and Eppinger (1997a) proposed that the expected duration for a sequence of coupled activities could be calculated by reward markov chain. For large systems, a heuristic procedure was presented to find an activity sequence with minimum expected duration. Banerjee et al. (2007) presented a quadratic integer program for the problem examined by Smith and Eppinger (1997a), and showed that the problem was NP-hard and its linear relaxation could only give a poor lower bound. Whitfield et al. (2003) presented a GA by including two criterions of DSM sequencing problem. More recently, Abdelsalam and Bao (2006, 2007) proposed Simulated Annealing approaches to determine the sequence of activities with the goal of minimizing iteration time. Meier et al. (2007) proposed a competent GA for finding an activity sequence with minimum feedback lengths. Lancaster and Cheng (2008) developed an evolutionary algorithm for obtaining an activity sequence with minimum feedback lengths. Luh et al. (2009) proposed the concept of fuzzy DSM, and developed a heuristic to reorder design activities so as to minimize feedbacks.

Another stream of literature focused on decomposition based methods. For instance, based on the rules developed by Steward (1981), Rogers (1989) developed an expert system called Design Manager's Aid for Intelligent Decomposition (DeMAID) for performing DSM analysis. In DeMAID, two steps were performed to get a hierarchical ordering: (1) Identifying the tightly coupled activities and forming them into blocks; and (2) Sequencing the activities within each block so as to minimize feedbacks. The DeMAID was further extended to DeMAID/GA by incorporating Genetic Algorithm (GA) to sequence the activities within each block (McCulley and Bloebaum, 1996; Rogers, 1996, 1999). Kusiak and Wang (1993) presented a triangularization algorithm for organizing design activities. Tang et al. (2000) presented an algorithm for identifying coupled activities, and an algorithm for figuring out the order levels of activities. Some other schemes for identifying coupled activities can be found in Gebala and Eppinger (1991), Chen and Li (2003), Chen et al. (2005), Chen et al. (2007).

Ahmadi et al. (2001) explicitly develop mathematical models for solving large-scale DSM sequencing problem. In their paper, the overall project was first decomposed into smaller sub-problems that were easier to solve; a Lagrangian relaxation based Branch-and-Bound method was then used to solve the sub-problems, and finally, the solutions of sub-problems were concatenated to a solution of the overall problem. They considered block structures as possible sub-problems, and presented a Block Decomposition problem with the objective of minimizing feedbacks among blocks. In what follows, we will describe in detail the Block Decomposition problem.

Assume that there are overall of *N* design activities. Let $i, j$ be the indexes for activities ($i, j = 1, 2, ..., N$), and $a_k$ be the degree of dependency of activity $j$ on activity *i*, where $k = (i, j)$ represents an arc from activity *j* to activity *i*. Defining $n_k = w$ if $a_k = 1$ ; $n_k = 1$ otherwise, where $w$ is a large positive number. Let $m$ ($m = 1, 2, ..., M$) be the index for the position of sequenced blocks, where $M$ is the number of blocks to be formed, and *C* be the maximum number of activities to be allowed in a block. The decision variables are: $x_{im} = 1$ if activity *i* is assigned to the *m*th block, $x_{im} = 0$ otherwise; $y_k = 1$ if arc $k = (i, j)$ is a feedback arc from a high-positioned block back to a low-positioned block, $y_k = 0$ otherwise. The Block Decomposition problem is then formulated as follows:

$$Min \sum_{k \in A} a_k n_k y_k \tag{2.1}$$

$$\text{s.t. } \sum_{m=1}^{M} x_{im} = 1, \text{ for } i = 1, 2, ..., N \tag{2.2}$$

$$\sum_{i=1}^{N} x_{im} \leq C, \text{ for } m = 1, 2, ..., M \tag{2.3}$$

$$x_{im} - \sum_{h=m+1}^{M} x_{jh} - y_k \leq 0, \text{ for } k = (i, j) \text{ and each } m \tag{2.4}$$

$$x_{im}, y_k \in \{0, 1\} \quad \forall i, m, k \tag{2.5}$$

The objective function (2.1) minimizes the feedbacks among blocks. Equations (2.2) and (2.3) are assignment constraints. Constraint (2.4) guarantees that the objective function (2.1) sums only the feedbacks among blocks. In particular, if activity *j* is assigned to a low-positioned block, and activity *i* is assigned to a high-positioned block, then $x_{im} = 1$, $\sum_{h=m+1}^{M} x_{jh} = 0$, and so $y_k = 1$ (since $y_k$ equals either zero or one), i.e. $a_k$ is a feedback and is included in (2.1); otherwise, $\sum_{h=m+1}^{M} x_{jh} = 1$, and consequently $y_k = 0$, i.e. $a_k$ is not a feedback and is not included in (2.1). Constraint (2.5) indicates that $x_{im}$ and $y_k$ are binary decision variables. The Block Decomposition problem can be solved using the Branch-and-Bound method, and the computational effort of sub-problems can be controlled by choosing the value of *C*.

The work of Ahmadi et al. (2001) has clearly shed light on the analysis of DSM sequencing problem, and has greatly advanced our understanding of managing NPD processes. However, a major disadvantage of above Block Decomposition method is that the resulting overall sequence of design activities (obtained by solving all sub-problems) may be a sequence that is worse than the initial one. For example, suppose that there are four activities, numbered from 1 to 4. The NDSM representation of the initial activity sequence is shown in Figure 2.6(a), where the numbers in the square matrix represent the information dependency levels among activities. Assume that these four activities need to be grouped into two blocks, and the number of activities in each block should be less than three. Then, applying the Block Decomposition method will lead to the activity sequence shown in Figure 2.6(b), i.e., the first block contains activities 1 and 3, and the second block contains activities 2 and 4, such that the total feedbacks among the two blocks is minimized at 0.9. Clearly, the total feedbacks of the NDSM in Figure 2.6(a) are 1.6, while the total feedbacks of the NDSM in Figure 2.6(b) are 1.8. In other words, the total feedbacks are increased, which imply that the resulting overall activity sequence by the Block Decomposition method is worse than the initial one.

| | **1** | **2** | **3** | **4** |
|---|---|---|---|---|
| **1** | ■ | 0.1 | 0.5 | 0.3 |
| **2** | 0.2 | ■ | 0.1 | 0.4 |
| **3** | 0.9 | 0.3 | ■ | 0.2 |
| **4** | 0.6 | 0.7 | 0.8 | ■ |

(a) Initial solution

| | **1** | **3** | **2** | **4** |
|---|---|---|---|---|
| **1** | ■ | 0.5 | 0.1 | 0.3 |
| **3** | 0.9 | ■ | 0.3 | 0.2 |
| **2** | 0.2 | 0.1 | ■ | 0.4 |
| **4** | 0.6 | 0.8 | 0.7 | ■ |

(b) Solution by Block Decomposition

**Figure 2.6 Disadvantage of block decomposition: an example**

## 2.4 Concluding Comments

How to optimally schedule various tests along the NPD process such that the highest product development performance, in terms of time, cost and quality can be achieved is an important decision problem. Past analytical models, which are developed to determine the optimal scheduling of tests, often focused on single-stage testing of sequential NPD process. While these studies are useful for managing the sequential process, the solutions and insights they get may not be applicable to overlapped NPD process, which has become a common mode of product development over the last decade. Moreover, testing strategies combined with overlapping policies may affect project performance differently compared with testing strategies in the sequential process. Therefore, it is meaningful and worthwhile to investigate the testing strategies in overlapped NPD process, which are the objectives of Chapter 3 to 4 of this thesis.

A key trade-off involved in overlapped NPD process is time reduction versus additional effort for downstream rework. Previous overlapping models (e.g. Krishnan et al., 1997; Loch and Terwiesch, 1998; Roemer et al., 2000; Wang and Yan, 2005; Lin et al., 2009) have clearly shown that, the optimal overlapping levels is determined by the arrival rate of upstream design modifications, the impact of each modification, and the downstream progress, i.e. the number of downstream tasks completed when the design change arrives.

Generally, in the testing and overlapping models, the NPD process is viewed as consisting of a series of development stages. When the NPD process is broken down into development stages, it's no need to consider the sequencing problem since the execution sequence of these stages (such as the concept design and the detail design) is fixed and will not change. However, when the NPD process is further broken into a number of smaller activities, then, a key and challenging issue, i.e. the planned timing and sequence of activities, arises because clear precedence constraints among design activities do not exist and are rarely known in advance (Eppinger et al., 1994; Ahmadi et al., 2001; Jun and Suh, 2008; Karniel and Reich, 2009).

Formal network-based planning techniques, such as CPM and PERT, cannot effectively model cyclic information flow and iteration, limiting their capability of scheduling for NPD projects (Ahmadi et al., 2001; Denker et al., 2001; Krishnan and Ulrich, 2001; Browning and Ramasesh, 2007). To address this shortfall, one popular approach is DSM, which has spawned many research efforts on sequencing design activities with the objective of minimize feedbacks. It is known that DSM sequencing problem is NP-complete. Previous decomposition based approaches provide an efficient way to solve large-scale problems (e.g. McCulley and Bloebaum, 1996;

Rogers, 1996, 1999; Ahmadi et al., 2001). In Chapter 5 and Chapter 6, we extend this line of research by developing some rules for feedback reduction, as well as two new decomposition approaches for solving large DSM sequencing problem.

In some real world situations, information dependencies among activities may be difficult to predict accurately, especially when the activities have never been performed before (Chen et al., 2004; Yassine, 2007; Luh et al., 2009). As reported by Dubois et al. (2003a), in dealing with imprecise parameters, fuzzy PERT/CPM models have been around for more than two decades (see e.g., Lootsma, 1989; Nasution, 1994; Wang, 1999, 2002, 2004; Chanas et al., 2002; Dubois et al., 2003b; Zielinski, 2005; Ke and Liu, 2007; Muhuri and Shukla, 2008; Petrovic et al., 2008; Liberatore, 2008). Moreover, for product development projects, we are often lack of enough data to derive the probability distributions for activity dependencies. Therefore, in Chapter 7 of this thesis, we resort to fuzzy set theory for tackling uncertain parameters, and present a fuzzy approach to DSM sequencing problem.

# CHAPTER 3

# OPTIMAL TESTING STRATEGIES IN OVERLAPPED DESIGN PROCESS

To model and structure NPD processes, an important decision problem is how to optimally schedule various tests along the NPD process so as to maximize the product development performance. Past studies, which are developed to determine the optimal scheduling of tests, often focused on single-stage testing of sequential design process. Meanwhile, overlapping development stages has become a common mode of product development (Terwiesch et al., 2002; Loch and Terwiesch, 2005; Mitchell and Nault, 2007; Yassine et al., 2008), and the testing strategies combined with overlapping policies may affect project performance differently compared with testing strategies in the sequential process. Thus, it is worthwhile to investigate the testing strategies in overlapped NPD process. The purpose of this chapter is therefore to present an analytical model for the scheduling of tests in overlapped design process, where a downstream stage starts before the completion of upstream testing.

The outline of this chapter is as follows. After a brief introduction in Section 3.1, we formally present the model in Section 3.2. The optimal policies are characterized in Section 3.3, followed by two problem variations introduced in Section 3.4. In Section 3.5, we illustrate the methodology with a case study at a handset design company. Section 3.6 concludes this study. Proofs of all the results in this chapter can be found in Appendix A.

## 3.1 Introduction

For most product development projects, the initial design inevitably contains design problems, such as mismatches with customer needs, technical design faults, or issues regarding manufacturability of the product (Thomke and Bell, 2001; Dahan and Hauser, 2002; Gil et al., 2008). Testing, which is a primary way to detect and resolve these problems, is central to product development (Thomke, 2003).

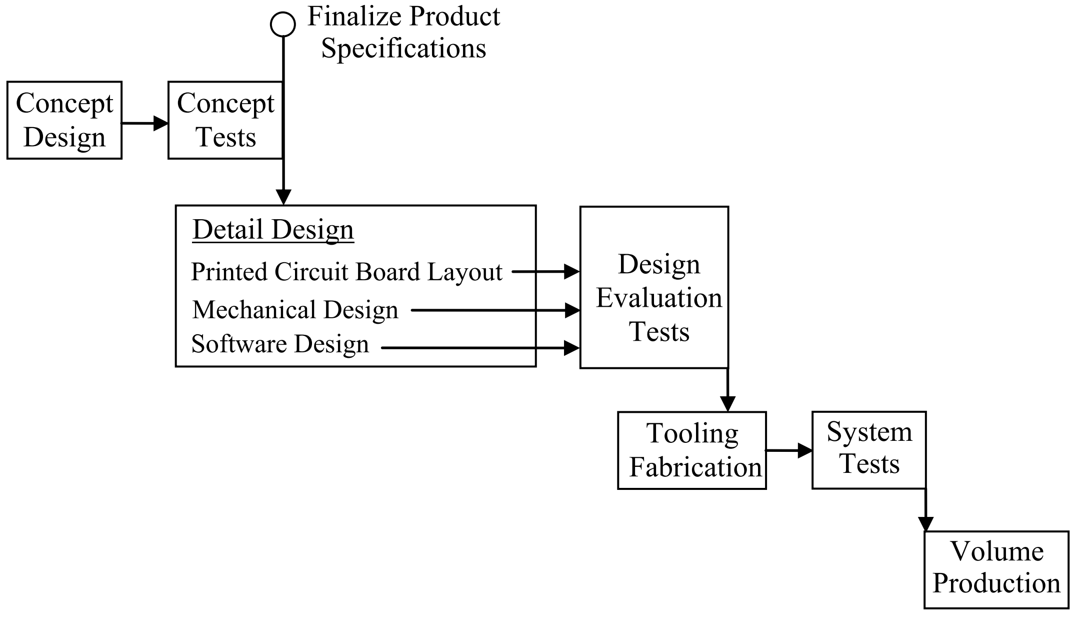


**Figure 3.1 Typical testing stages in the development of mobile phones**

Typically, tests are carried out in several stages with increasing fidelity (Wheelwright and Clark, 1992; Thomke, 2007). For example, as shown in Figure 3.1, there are four stages in the development of mobile phones: *Concept Design*, *Detail Design*, *Tooling Fabrication*, and *Volume Production*. Following these development stages, different testing activities are carried out. *Concept tests* are performed by asking customers to evaluate the 3D drawings or digital mock-ups, making sure that the design meets their requirements. *Design Evaluation Tests* are carried out separately in each subsystem of the product, to ensure compliance with product specifications and remove variations. In *System Tests*, physical prototypes are used to improve the overall performance of the product.

Due to the accumulative and proliferous nature of design problems, the outputs of an upstream stage need to be tested extensively before releasing them to downstream stages. Inadequate testing would allow design problems to propagate, and finally deteriorate the product quality. On the other hand, testing also incurs time and cost. Too much testing at one stage would impede the project's progress and increase development costs. Hence, the key issue is how to optimally schedule various testing activities along the design process so that the best project performance, in terms of time, cost and quality, can be achieved.

Some models have been developed to determine the optimal scheduling of tests and/or reviews for product development projects. For example, Ha and Porteus (1995) developed a dynamic program to determine the optimal frequency of upstream design reviews within two overlapped design phases. Dahan and Mendelson (2001) modeled the concept testing as a probabilistic search process and proposed an extreme-value model to determine the optimal number of tests for concept development. Thomke and Bell (2001) developed a model to decide the optimal timing, frequency and fidelity of sequential tests. They showed that the optimal testing strategy should balance several things, including the testing cost and the cost of redesign. Test scheduling problem has been studied extensively in software development literature (see e.g. Xie and Yang, 2001; Dai et al., 2003; Pham and Zhang, 2003; Xie and Yang, 2003; Serich, 2005; Kapur et al., 2007; Wu et al., 2007; Yang et al., 2008). These models have clearly shed light on the analysis of test scheduling problem. However, they focus on the testing policies at one design stage and do not take into account the multi-stage nature of testing process.

Ahmadi and Wang (1999), as well as Kogan and Raz (2002), built analytical models and explicitly examined how to optimally schedule tests for multiple development stages. The former assumed that all development stages are carried out in fully sequential, while the latter assumed that all stages start and finish simultaneously. However, in practice, the development stages are often overlapped (i.e. in partial parallel) rather than in fully sequential or parallel (Krishnan, 1996; Terwiesch and Loch, 1999; Roemer and Ahmadi, 2004; Yassine et al., 2008). As far as we know, no analytical model exists for scheduling tests in overlapped NPD process.

Recently, Lin et al. (2008) built a simulation model for overlapped iterative product development. Given a set of testing strategies and overlapping policies, their work can be used to compare these solutions and identify which one is best. However, in their model, verifying the optimality of a solution requires enumerating all possible solutions. Thus, for problems with continuous decision variables, it's unlikely to get a good solution quickly and efficiently. Unlike their research, we built an analytical model to examine the test scheduling in overlapped process, and to help management derive the optimal testing and overlapping strategies. Moreover, analysis of our model yields several useful insights (Propositions 3.1-3.5 and Corollaries 3.1-3.3) which cannot be derived from their model.

## 3.2 Model Formulation

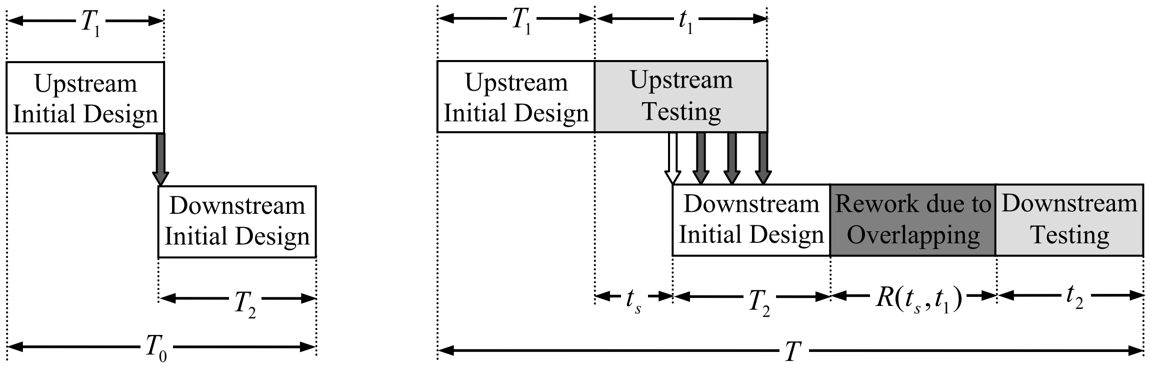
In this section, we first formulate the tradeoffs involved in scheduling tests in overlapped design process: product quality improvement versus additional costs caused by testing and overlapping. Based on existing literature, we then model testing as a continuous NHPP process of discovering and solving design problems, and establish functions of product quality and downstream rework duration caused by overlapping. The objective function, constraints, model parameters, and decision variables are summarized at the end of this section.

## 3.2.1 Overview of the Model

Consider the product development process with two design stages, we call the first stage *upstream* and the second *downstream*. The downstream stage (e.g. Tooling Fabrication) depends on the output information from the upstream stage (e.g. Detail Design). Figure 3.2(a) shows the product development process where testing and overlapping are not applied. Clearly, the completion time for this process is: $T_0 = T_1 + T_2$ , where $T_1$ and $T_2$ respectively denote the regular durations for performing initial development of upstream and downstream stages.

Figure 3.2(b) shows the product development process where testing and overlapping are applied. For most development projects, the initial outputs of each stage inevitably contain design problems, such as mismatches with customer needs or technical design faults (Thomke and Bell, 2001). We assume that testing activities are then carried out to detect and resolve these problems. Moreover, we assume that these design problems can only be detected by testing activities. There are three decision variables: $t_1$ and $t_2$, the respective testing durations at the upstream and downstream stages, and $t_s$, the time elapsed between beginning the upstream tests and beginning the downstream development. In an overlapped process, downstream stage can start at any time after the initial upstream design is available and before the completion of upstream testing (i.e. $0 \le t_s \le t_1$), and so project completion time may be reduced.

However, because the downstream stage starts on the preliminary information of the upstream, some of the downstream tasks completed during the overlapped period may have to be reworked when upstream design changes. Hence, the time savings from overlapping must be weighed against the additional effort caused by downstream rework (Krishnan et al., 1997; Loch and Terwiesch, 1998; Roemer et al., 2000; Chakravarty, 2001; Lin et al., 2009). The downstream rework duration due to overlapping is captured by $R(t_s, t_1)$. To ensure that all design changes are absorbed by downstream stage, upstream testing should be terminated before the completion of downstream initial development, i.e. $t_1 \le t_s + T_2$.



**(a) No testing and overlapping** **(b) With testing and overlapping**

| | | | |
|---|---|---|---|
| ⇨ | Preliminary information | $R(t_s, t_1)$ | Rework duration due to overlapping |
| ➡ | Modified information | $T$ | Completion time for the process with testing and overlapping |
| $T_1$ | Regular duration for performing upstream initial development | | **Decision variables** |
| $T_2$ | Regular duration for performing downstream initial development | $t_1$ | Upstream testing duration |
| | | $t_2$ | Downstream testing duration |
| $T_0$ | Completion time for the process without testing and overlapping | $t_s$ | The time elapsed between beginning upstream tests and beginning downstream initial design |

**Figure 3.2 Product development processes**

We define product quality as the number of design problems remaining in the system. Clearly, product quality is not only influenced by design problems introduced at the downstream stage, but also by the propagated variations from upstream stage.

Let $N_0$ be the product quality for the process where testing and overlapping are not applied (shown in Figure 3.2(a)), i.e. the number of design problems cumulated at the end of downstream initial design. Let $N(t_1,t_2)$ be the product quality for the process with testing and overlapping (shown in Figure 3.2(b)), i.e. the number of residual design problems at the end of downstream testing. It is clear that design problems are reduced through testing, and thus $N(t_1,t_2)$ is smaller than $N_0$, i.e. product quality is improved. Let $c_q$ be the expected cost of removing one remaining problem after downstream testing. The benefits of product quality improvement can then be written as: $c_q[N_0 - N(t_1,t_2)]$.

However, the benefits of product quality improvement must be weighed against the additional costs caused by testing and overlapping, which include opportunity costs (i.e. the financial loss of delaying the new product's time-to-market), testing costs, and downstream rework costs. We first consider opportunity costs. Clearly, the completion time for the process with testing and overlapping is:

$$T = T_1 + T_2 + t_s + t_2 + R(t_s,t_1) \tag{3.1}$$

Hence, the increased completion time is: $(T - T_0) = t_s + t_2 + R(t_s,t_1)$. Let $\beta$ be the opportunity cost (per unit of time). As in previous paper (e.g. Chakravarty, 2001; Wang and Yan, 2005; Lin et al., 2009), we assume $\beta$ is constant, the overall opportunity costs can then be written as: $\beta[t_s + t_2 + R(t_s,t_1)]$. Let $c_1$ and $c_2$ respectively denote the unit testing costs at the upstream and downstream stages, $c_r$ denote the downstream rework cost (per unit of time). The additional costs caused by testing and overlapping $C_D$ can then be represented as:

$$C_D = \beta[t_s + t_2 + R(t_s,t_1)] + c_1t_1 + c_2t_2 + c_rR(t_s,t_1) \tag{3.2}$$

The overall gain from testing and overlapping is: $c_q\left[N_0 - N(t_1,t_2)\right] - C_D$. The objective is thus to determine the optimal $t_1$, $t_2$, and $t_s$ so as to maximize the overall gain. Since $N_0$ is constant, $\max : c_q\left[N_0 - N(t_1,t_2)\right] - C_D$ is equivalent to minimize the overall cost:

$$Min\ C = \beta\left[t_s + t_2 + R(t_s,t_1)\right] + c_1 t_1 + c_2 t_2 + c_r R(t_s,t_1) + c_q N(t_1,t_2) \tag{3.3}$$

The functions of $N(t_1,t_2)$ and $R(t_s,t_1)$ will be established in the following subsections.

### 3.2.2 Modeling Testing Processes

In this study, we treat testing as a continuous process of identifying and solving design problems. Previous studies (e.g. Boehm, 1981; Shooman, 1983; Thomke, 1998) have shown that with increasing amount of testing, the number of residual design problems tends to decrease, and thus the rate of discovering and solving design problems should decrease. In accordance with these empirical studies, in the System Dynamics models of testing in product development (e.g. Cooper, 1980; Ford and Sterman, 1998; Lin et al., 2008; Love et al., 2008), the problem discovering and solving rate is modeled as proportional to the number of residual problems at that time. Consistent with these studies, the problem discovering and solving rate $\lambda_j(t)$ is modeled as:

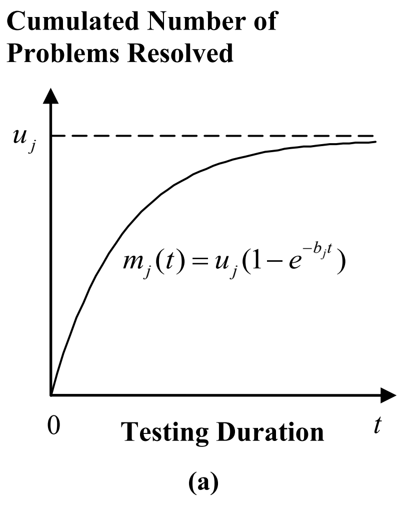
$$\lambda_j(t) = \frac{dm_j(t)}{dt} = b_j\left[u_j - m_j(t)\right] \tag{3.4}$$

$j = 1, 2$ is the index of upstream and downstream stages. $m_j(t)$ denotes the cumulated number of problems eliminated by time $t$. $u_j$ represents the number of design problems cumulated before testing, and so $[u_j - m_j(t)]$ is the number of residual design problems in stage $j$ by time $t$. Parameter $b_j > 0$ denotes degree by which problems are solved per time unit in stage $j$ : higher $b_j$ represents faster problem-solving.

Solving the equation (3.4), together with the initial condition $m_j(0) = 0$, we get:

$$m_j(t) = u_j\left(1 - e^{-b_j t}\right) \tag{3.5}$$

Figure 3.3(a) and Figure 3.3(b) show the shape of $m_j(t)$ and $\lambda_j(t)$, respectively.



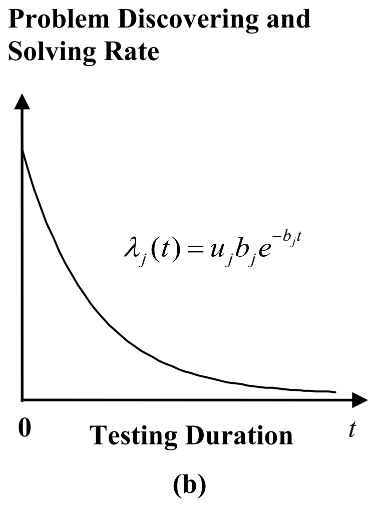


**Figure 3.3 The shape of $m_j(t)$ and $\lambda_j(t)$**

From equation (3.5), we know the expected number of remaining design problems at the upstream stage after testing is: $u_1 e^{-b_1 t_1}$. These undetected design problems will accumulate and then have negative impact on the relevant tasks at the downstream stage. Let $w_2$ be the work content or the total number of tasks at the downstream stage, and $k$ be the percentage of downstream tasks affected by one upstream design problem, $a_2$ be the number of incorrectly done tasks (i.e. design problems) by downstream designers in the initial design. Then, the total number of downstream design tasks which are "flawed" by these upstream design problems is: $kw_2u_1e^{-b_1t_1}$. As shown in Figure 3.4, there is an overlap between the design problems introduced by downstream designers and the downstream design tasks that are "flawed" by upstream design problems, and so we should not double count these design problems. Thus, the number of design problems cumulated at the downstream stage before testing, $u_2$, can be written as:

$$u_2 = a_2 + k(w_2 - a_2)u_1e^{-b_1t_1} \tag{3.6}$$

In other words, $u_2$ is the sum of (1) the number of design problems introduced by downstream designers, and (2) the relevant tasks that are correctly performed by downstream designers but are "flawed" by undetected design problems at the upstream stage.

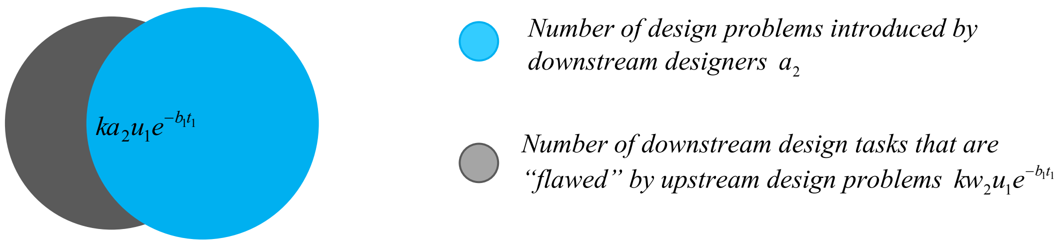

$ka_2u_1e^{-b_1t_1}$
*Number of design problems introduced by downstream designers* $a_2$
*Number of downstream design tasks that are "flawed" by upstream design problems* $kw_2u_1e^{-b_1t_1}$

**Figure 3.4 Illustration of the formulation of $u_2$**

Product quality for the process with testing and overlapping, i.e. the expected remaining design problems at the end of downstream testing, is:

$$N(t_1,t_2) = u_2 - m_2(t_2) = [a_2 + k(w_2 - a_2)u_1e^{-b_1t_1}]e^{-b_2t_2} \tag{3.7}$$

## 3.2.3 Modeling Downstream Rework

For sequential process, downstream stage starts only when most of the upstream design problems have been resolved. For example, in a fully sequential process, tooling design only starts after the design evaluation tests have been completed and most of the quality problems in detail design stage have been removed. However, in practice, tooling design usually starts before the design evaluation tests have been completed in order to reduce project completion time. Hence, during the overlapped period, certain percentage of tooling design tasks is completed based on wrong information (i.e. design problems) of detail design stage, and will have to be reworked when these detail design problems are identified and resolved.

Previous research (e.g. Krishnan et al., 1997; Loch and Terwiesch, 1998; Roemer et al., 2000; Chakravarty, 2001; Lin et al., 2008) have clearly shown that, the amount of downstream rework is determined by the arrival rate of upstream design modifications, the impact of each modification, and the downstream progress, i.e. the number of downstream tasks completed when the design modification arrives. More downstream work will be reworked when the modification rate and/or impact are high. Consistent with past studies (e.g. Chakravarty, 2001; Roemer and Ahmadi, 2004; Lin et al., 2009), we focus on the development projects where the information exchange between design stages is costless. In other words, when an upstream design problem is identified and design modification is made, this design modification will be incorporated immediately by downstream stage. The downstream rework duration $R(t_s, t_1)$ can then be represented as:

$$R(t_s, t_1) = \int_{T_1+t_s}^{T_1+t_1} \lambda_1(x-T_1) \times k \times (x-T_1-t_s)dx = ku_1\left[\frac{e^{-b_1 t_s}}{b_1} - \left(t_1 - t_s + \frac{1}{b_1}\right)e^{-b_1 t_1}\right] \tag{3.8}$$

Where $x$ represents the time elapsed since beginning upstream initial development. The upstream tests start on time $T_1$, and so the problem discovering and solving rate of the upstream stage at time $x$ is $\lambda_1(x-T_1)$, which equals to: $u_1 b_1 e^{-b_1(x-T_1)}$. $k$ is the impact factor, i.e. the percentage of downstream tasks affected by one upstream design problem, and $(x - T_1 - t_s)$ represents the downstream progress at time *x*.

### 3.2.4 Summary

We summarize the model parameters and decision variables in Table 3.1, and state the optimization problem as follows:

$$MinC = \beta(t_s + t_2) + ku_1(c_r + \beta)\left[\frac{e^{-b_1 t_s}}{b_1} - \left(t_1 - t_s + \frac{1}{b_1}\right)e^{-b_1 t_1}\right] + c_q[a_2 + k(w_2 - a_2)u_1 e^{-b_1 t_1}]e^{-b_2 t_2}$$
$$+ c_1 t_1 + c_2 t_2 \tag{3.9}$$

Subject to:

$$0 \le t_1 - t_s \le T_2 \tag{3.10}$$

$$t_1,\ t_2,\ t_s \ge 0 \tag{3.11}$$

**Table 3.1 Model parameters and decision variables**

| **Parameters** | **Definition** |
|---|---|
| $T_1$, $T_2$ | Regular duration for performing initial development of upstream and downstream stages, respectively |
| $u_1$, $u_2$ | The respective number of design problems cumulated at the upstream and downstream stages before testing |
| $a_2$ | The amount of problems introduced by downstream designers |
| $w_2$ | Total number of tasks at the downstream stage |
| $b_1$, $b_2$ | Problem-solving capacity at upstream and downstream stages, respectively |
| $k$ | The percentage of downstream tasks affected by one upstream design problem |
| $\beta$ | Opportunity cost (per unit of time) |
| $c_r$ | Downstream rework cost (per unit of time) |
| $c_1$, $c_2$ | Unit testing cost at upstream and downstream stages, respectively |
| $c_q$ | The expected cost of removing one remaining design problem after downstream testing |
| **Decision Variables** | **Definition** |
| $t_s$ | The time elapsed between beginning the upstream tests and beginning the downstream development |
| $t_1$, $t_2$ | The respective testing durations at upstream and downstream stages |

## 3.3 Policy Analysis

### 3.3.1 Model Solution

In this section, we will describe how to solve the optimization problem. We first derive the optimal time elapsed between beginning the upstream tests and beginning the downstream development $t_s^*$, and optimal downstream testing duration $t_2^*$, by assuming the optimal upstream testing duration $t_1^*$ has been derived. The results are stated in Proposition 3.1.

Let $\theta_1 = \dfrac{\beta}{ku_1(c_r+\beta)}$, $\theta_2 = \dfrac{c_2+\beta}{c_q b_2 k u_1 (w_2 - a_2)}$, and $\theta_3 = \dfrac{a_2}{ku_1(w_2-a_2)}$.

**PROPOSITION 3.1** *Suppose that $t_1^*$ is given,*

*(a) $t_s^*$ can be determined by the following equation:*

$$t_s^* = \max\left[0, -\frac{1}{b_1}\ln\left(\theta_1 + e^{-b_1 t_1^*}\right)\right] \tag{3.12}$$

*(b) If $\theta_1 \geq 1$, then $t_s^* = 0$ for any value of $t_1^*$.*

*(c) The optimal downstream testing duration $t_2^*$ is:*

$$t_2^* = \max\left[0, \frac{1}{b_2}\ln\left(\frac{\theta_3 + e^{-b_1 t_1^*}}{\theta_2}\right)\right] \tag{3.13}$$

*(d) If $\theta_2 - \theta_3 \geq 1$, then $t_2^* = 0$ for any value of $t_1^*$.*

The proof of all the results in this chapter can be found in Appendix A.

Proposition 3.1(a) characterizes $t_s^*$ when $t_1^*$ is fixed. Part (b) generalizes the sufficient condition for complete overlapping ($t_s^* = 0$), i.e. starting downstream stage as soon as preliminary information is available. Since this conclusion is independent on $t_1^*$, it is useful for managers to decide whether complete overlapping policy should be applied to a development project. Proposition 3.1(c) defines the optimal downstream testing duration $t_2^*$ when $t_1^*$ is given. Similarly, we generalize the sufficient condition for $t_2^* = 0$ in Proposition 3.1(d), which is independent on $t_1^*$.

Our next result expresses the optimal product quality $N(t_1^*, t_2^*)$ as a function of the model's parameters.

**COROLLARY 3.1** *If* $\theta_2 - \theta_3 < e^{-b_1 t_1^*}$*, i.e.* $t_2^* \neq 0$*, then* $N(t_1^*, t_2^*) = \dfrac{c_2 + \beta}{c_q b_2}$.

Corollary 3.1 shows that when $t_2^* \neq 0$, with lower downstream testing cost $c_2$ and opportunity cost $\beta$, the company should increase its target product quality. Higher $c_q$ as well as higher downstream problem-solving capacity $b_2$ also lead to higher target level of product quality. For many development projects in practice, $c_q$ can be very costly and so $t_2^* \neq 0$. Therefore, this result is of high applicability.

With the result from Proposition 3.1, our next proposition determines the optimal upstream testing duration $t_1^*$. By substituting (3.12) and (3.13) into (3.9) and (3.10), we get:

$$MinC(t_1) = \beta\left(1 + \frac{e^{-b_1 t_1}}{\theta_1}\right) \times \max\left[0, -\frac{1}{b_1}\ln\left(\theta_1 + e^{-b_1 t_1}\right)\right] + (c_2 + \beta) \times \max\left[0, \frac{1}{b_2}\ln\left(\frac{\theta_3 + e^{-b_1 t_1}}{\theta_2}\right)\right]$$
$$+ c_1 t_1 + \frac{\beta}{b_1 \theta_1}\left\{\min\left[1, \theta_1 + e^{-b_1 t_1}\right] - (b_1 t_1 + 1)e^{-b_1 t_1}\right\} + a_2 c_q\left(1 + \frac{e^{-b_1 t_1}}{\theta_3}\right) \times \min\left[1, \frac{\theta_2}{\theta_3 + e^{-b_1 t_1}}\right] \qquad (3.14)$$

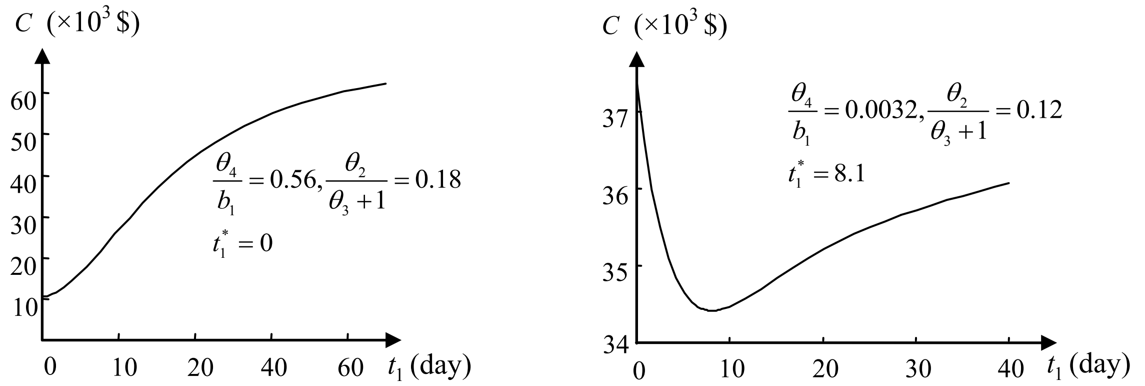
Subject to:

$$0 \leq t_1 - \max\left[0, -\frac{1}{b_1}\ln\left(\theta_1 + e^{-b_1 t_1}\right)\right] \leq T_2 \qquad (3.15)$$

**Proposition 3.2** Let $\theta_4 = \dfrac{c_1}{c_q k u_1 (w_2 - a_2)}$, $\Phi(t_1) = t_1 - \max\left[0, -\dfrac{1}{b_1}\ln\left(\theta_1 + e^{-b_1 t_1}\right)\right]$.

*(a)* $\Phi(t_1)$ *is strictly increasing in* $t_1$.

*(b) if* $\dfrac{\theta_4}{b_1} \geq \min\left[1, \dfrac{\theta_2}{\theta_3 + 1}\right]$, *the total cost in (3.14) is strictly increasing in* $t_1$, *leading to* $t_1^* = 0$, *which is illustrated in Figure 3.5(a). Otherwise, the total cost is first convex then concave increasing in* $t_1$, *which is shown in Figure 3.5(b).*



**(a):** $\{u_1, b_1, a_2, w_2, b_2, k, \beta, c_1, c_2, c_r, c_q\} =$ $\{30, 0.05, 80, 150, 0.9, 0.03, 3000, 100, 150, 50, 200\}$

**(b):** $\{u_1, b_1, a_2, w_2, b_2, k, \beta, c_1, c_2, c_r, c_q\} =$ $\{30, 0.2, 150, 300, 0.1, 0.03, 1000, 30, 200, 150, 350\}$

**Figure 3.5 Effect of upstream testing on total cost: numerical example**

Proposition 3.2(a) reveals that in constraint (3.15), $\Phi(t_1)$ strictly increases with $t_1$. Therefore, there exist a unique value of $t_1$ which satisfies: $\Phi(t_1) = T_2$. This value, denoted as $\bar{t}_1$, is the upper bound of $t_1^*$. We can easily find $\bar{t}_1$ using the Binary Search method. The idea behind the Binary Search method is that whether $\Phi(t_1)$ is greater or less than $T_2$ at a trail solution indicates whether $t_1$ should be decreased or increased. Specifically, if $\Phi(t_1) > T_2$, then $t_1$ should be decreased, otherwise, if $\Phi(t_1) < T_2$, then $t_1$ should be increased. Note that because $\theta_1 \geq 0$, the left side of constraint (3.15), $0 \leq \Phi(t_1)$, will automatically hold for all feasible $t_1$.

Proposition 3.2(b) implies that there exists a unique value of $t_1^*$ which minimizes the total cost in (3.14). When $\frac{\theta_4}{b_1} < \min\left[1, \frac{\theta_2}{\theta_3 + 1}\right]$, the optimal upstream testing duration $t_1^*$ locates in $\left[0, \overline{t_1}\right]$. Here the total cost in (3.14) is first convex then concave increasing in $t_1$. This implies that for $x_1$ and $x_2$ ($0 \le x_1 \le x_2 \le \overline{t_1}$), if $C(x_1) > C(x_2)$, then $t_1^* \in [x_1, \overline{t_1}]$; conversely, if $C(x_1) \le C(x_2)$, then $t_1^* \in [0, x_2]$. To identify $t_1^*$, the method of Golden Section Search is employed to generate points $x_1$ and $x_2$, see Hillier and Lieberman (2001) for more details of the classical Binary and Golden Section Search techniques.

**Notation**

$\underline{t_1}$ = lower bound of $t_1^*$, $\overline{t_1}$ = upper bound of $t_1^*$, $M =$ large positive value, $\varepsilon =$ small positive value.

*Step 1.* Calculate $\theta_1$, $\theta_2$, $\theta_3$, $\theta_4$. If $\frac{\theta_4}{b_1} \ge \min\left[1, \frac{\theta_2}{\theta_3 + 1}\right]$, $t_1^* = 0$; Else, let $\underline{t_1} = 0$, $\overline{t_1} = M$,

$$r = \frac{\sqrt{5} - 1}{2}.$$

*Step 2.* **Iteration 1** [Binary Search for $\overline{t_1}$]:

*(a)* Let $\tilde{t}_1 = \left(\underline{t_1} + \overline{t_1}\right)/2$. Calculate $\Phi(\tilde{t}_1)$.

*(b)* If $\Phi(\tilde{t}_1) < T_2$, then $\underline{t_1} = \tilde{t}_1$, go to *Step (a)*; if $\Phi(\tilde{t}_1) > T_2$, then $\overline{t_1} = \tilde{t}_1$, go to *Step (a)*. Else, let $\underline{t_1} = 0$, $\overline{t_1} = \tilde{t}_1$.

*Step 3.* **Iteration 2** [Golden Section Search for $t_1^*$]:

*(a)* If $\overline{t_1} - \underline{t_1} \le \varepsilon$, then $t_1^* = \left(\underline{t_1} + \overline{t_1}\right)/2$, stop.

*(b)* Let $x_1 = \overline{t_1} - r\left(\overline{t_1} - \underline{t_1}\right)$, $x_2 = \underline{t_1} + r\left(\overline{t_1} - \underline{t_1}\right)$. Calculate $C(x_1)$ and $C(x_2)$.

*(c)* If $C(x_1) > C(x_2)$, then $\underline{t_1} = x_1$, else, $\overline{t_1} = x_2$. Go to *Step (a).*

## 3.3.2 Impact of Parameters on the Optimal Solution

In this section, we will discuss the influence of model parameters on the optimal solution, where the optimal solution does not take the boundary values, i.e. $t_s^* \neq 0$, $t_2^* \neq 0$, $t_1^* \neq 0$ and $t_1^* + \frac{1}{b_1}\ln\left(\theta_1 + e^{-b_1 t_1^*}\right) \leq T_2$. Proposition 3.3 lists the factors that directly affect the optimal upstream testing duration $t_1^*$. Corollaries 3.2 and 3.3 discuss the factors that affect $t_s^*$ and the optimal downstream testing duration $t_2^*$, respectively.

**PROPOSITION 3.3**

*(a) $t_1^*$ increases with $u_1$ and $k$, and decreases with $c_r$.*

*(b) There exists a critical $b_1^0$ such that: if $b_1 < b_1^0$, $t_1^*$ increases with $b_1$; if $b_1 > b_1^0$, $t_1^*$ decreases with $b_1$.*

*(c) If $c_r < c_2 < c_1$, then $t_1^*$ increases with $\beta$. Otherwise, there exists a critical $\beta^0$ such that: if $\beta < \beta^0$, $t_1^*$ decreases with $\beta$; if $\beta > \beta^0$, $t_1^*$ increases with $\beta$.*

*In the above, $b_1^0$ and $\beta^0$ are defined in Appendix A.*

**COROLLARY 3.2** *$t_s^*$ increases with $u_1$ and $k$.*

**COROLLARY 3.3** *$t_2^*$ decreases with $c_2$, and increases with $a_2$.*

Note that if we define overlapping degree $\gamma = 1 - t_s / t_1$, then based on above results, our model suggests that the optimal degree of overlapping decreases with $u_1$ and $k$. This result is consistent with Loch and Terwiesch (1998).

### 3.3.3 Testing Strategies in Sequential Process

In this section, we compare the testing strategies in a concurrent design process with those in a sequential development process. Let $\tau_1$ and $\tau_2$ be the respective testing durations of upstream and downstream stages in sequential process. By setting $t_s = t_1$ in (3.9), the test scheduling problem in sequential process can be represented as:

$$Min\ \ C = \beta(\tau_1 + \tau_2) + c_1\tau_1 + c_2\tau_2 + c_q[a_2 + k(w_2 - a_2)u_1 e^{-b_1\tau_1}]\, e^{-b_2\tau_2} \tag{3.16}$$

Subject to $\qquad \tau_1, \tau_2 \geq 0$ $\qquad$ (3.17)

**Proposition 3.4** *The optimal testing durations for upstream and downstream stages in the sequential process, denoted as $\tau_1^*$ and $\tau_2^*$, differ from $t_1^*$ and $t_2^*$. In particular,*

*(a) $t_1^* \geq \tau_1^*$, and $t_2^* \leq \tau_2^*$.*

*(b) if $c_2 \geq c_1$, then $t_1^* + t_2^* \geq \tau_1^* + \tau_2^*$.*

Proposition 3.4(a) reveals that compared with sequential process, in overlapped process, the upstream testing duration is increased, while the downstream testing duration is reduced. Part (b) states that if the unit testing cost at the downstream stage is not less than the unit testing cost at the upstream stage, then compared with sequential process, the whole testing time in overlapped process is increased. The intuitive interpretation of Proposition 3.4 is that: in a sequential process, increasing upstream testing will significantly delay downstream start time, while in a concurrent process, downstream stage can start before the completion of upstream testing, and thus the impact is much smaller. Hence, compared with a sequential process, in an overlapped process, the testing duration of upstream stage is increased.

More importantly, Proposition 3.4 shows that the optimal testing strategies in overlapped process differ from those in sequential process. In other words, testing and overlapping are interacted. Thus, to improve project performance, testing strategies and overlapping policies should be adjusted coordinately. For example, in practice, some companies may intend to shift from sequential to overlapped development process. Then, according to Proposition 3.4, these companies may also need to adjust their existing testing policies (even existing testing policies are optimal in the sequential process). Proposition 3.4 can be used as a guideline for structuring product development processes.

## 3.4 Problem Variations

In the previous sections, we have developed a model for minimizing the total cost, which we refer to as the *cost minimization problem*. However, in some situations, Pareto-optimal solutions should be considered. These are strategies that achieve the required product quality at minimum cost, or conversely, strategies whose product quality is best for a given budget, which we refer to as the *target quality problem* and the *budget constraint problem*, respectively. In this section, we will present these two variations of the *cost minimization problem*, and show that these two variations can then be solved by the same approach as the *cost minimization problem*.

**Target Quality Problem**

In this version of problem, we assume that product quality, i.e., the target number of remaining design problems after downstream testing $\overline{N}$ is given, and the problem is to determine the optimal $t_1$, $t_2$, and $t_s$ so as to achieve the required product quality at minimal cost:

$$\left.\begin{aligned}
&Min\ C_D = \beta[t_s + t_2 + R(t_s,t_1)] + c_1 t_1 + c_2 t_2 + c_r R(t_s,t_1)\\
&\text{Subject to: } 0 \le t_1 - t_s \le T_2,\\
&t_1,\ t_2,\ t_s \ge 0,\\
&N(t_1,t_2) = [a_2 + k(w_2 - a_2)u_1 e^{-b_1 t_1}]e^{-b_2 t_2} \le \overline{N}
\end{aligned}\right\} \qquad (3.18)$$

**Budget Constraint Problem**

In this version of the problem, we need to determine the optimal $t_1$, $t_2$, and $t_s$ so as to achieve the highest product quality for a given budget $\overline{C}$. This problem, which can described as the "dual" of the *target quality problem*, can be formulated as follows:

$$\left.\begin{aligned}
&Min\ N(t_1,t_2) = [a_2 + k(w_2 - a_2)u_1 e^{-b_1 t_1}]e^{-b_2 t_2}\\
&\text{Subject to: } 0 \le t_1 - t_s \le T_2,\\
&t_1,\ t_2,\ t_s \ge 0,\\
&C_D = \beta[t_s + t_2 + R(t_s,t_1)] + c_1 t_1 + c_2 t_2 + c_r R(t_s,t_1) \le \overline{C}
\end{aligned}\right\} \qquad (3.19)$$

**PROPOSITION 3.5** *The target quality problem in (3.18) and budget constraint problem in (3.19) have similar solution structure as the cost minimization problem.*

## 3.5 Model Application

In this section, we illustrate the model on the completed projects at a handset design company in China. The company designs mobile phones according to market and technology trends and sells the design to manufacturers. As of July 2006, this company employs approximately 2,600 professionals, of whom 90% are engineers, and has provided more than 100 design solutions for companies such as NEC, Kyocera, and Mitsubishi. However, in recent years, it has been facing increasing pressure to develop better products in short intervals. The development process of cellular phones has been shown in Figure 3.1. Our focus is on the *Design Evaluation Tests* at the Detail Design stage (i.e. upstream stage) and the *system tests* at the Tooling Fabrication stage (i.e. downstream stage), since they consume a large proportion of design resources.

### 3.5.1 Data Gathering

The data collection was quite challenging since the company did not know how past project data could be used in guiding future projects. Multiple sources were used to estimate the model parameters, including available company documents, extensive interviews and so on. We analyzed five projects with similar technical complexity. The projects were completed in the first half of 2006. Without special explanation, the data presented in this subsection were average values of these five projects. We now describe our data collection efforts in detail.

Our first step was to collect the data of testing, and derive the functions of design modifications at the Detail Design stage and at the Tooling Fabrication stage. This information was obtained by interviewing project managers, design engineers, tooling engineers, and by checking the project schedules and the quality control system of the projects. After completing the preliminary detail design, the drawings were reviewed by the experienced engineers, as well as tested by dozens of working samples fabricated using soft dies. The design drafts were continuously modified to remove identified design problems. On the average, the tests at this stage lasted for 12 working days. Tooling was made by suppliers. Faced with high time pressure, the engineers released 2D and 3D drawings to suppliers and pushed them to start Tooling Fabrication before the completion of *Design Evaluation Tests*. The cost of information transformation in our application was relatively low. *System tests* were performed after product designs were realized as physical prototypes. At this stage, more than 100 different items of tests were conducted, and the average testing duration was 15 weeks (6 days per week).

**Table 3.2 Design problems in detail design**

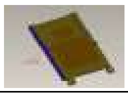
| No. | Issue | Severity | Root Cause | Action | Picture | Closure Date |
|---|---|---|---|---|---|---|
| 1 | B1, B2, and B3 whistle. | A | Microphone rubber holder is inside the housing, which results in a loop between the speaker and the microphone. | Extend the microphone holder and make it out of the housing. Lay the microphone holder on the housing surface. | | 10 Apr. 2006 |
| 2 | ESD failed. | A | Metal LCD holder is contacted with PCB pad. | Add a pad on the metal LCD holder. |  | 20 Apr. 2006 |

**Table 3.3 Cumulated design modifications in design evaluation tests**

| Days | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Cumulated Modifications | 0 | 12 | 20 | 27 | 32 | 35 | 38 | 40 | 42 | 45 | 47 | 49 | 50 |

**Table 3.4 Cumulated design modifications in system tests**

| Weeks | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Cumulated Modifications | 0 | 33 | 53 | 117 | 129 | 135 | 146 | 194 | 203 | 211 | 223 | 245 | 248 | 256 | 260 | 261 |

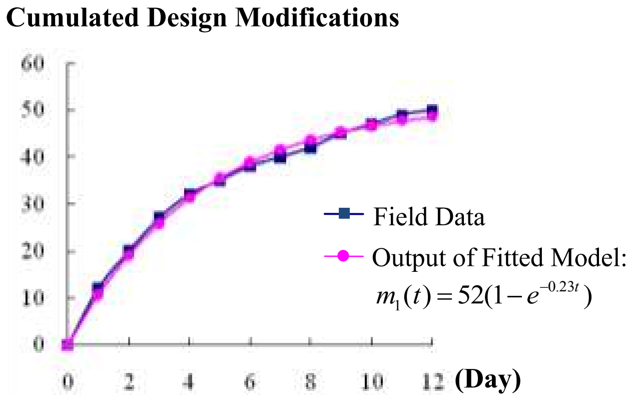


**Figure 3.6 Cumulated design modifications in design evaluation tests**

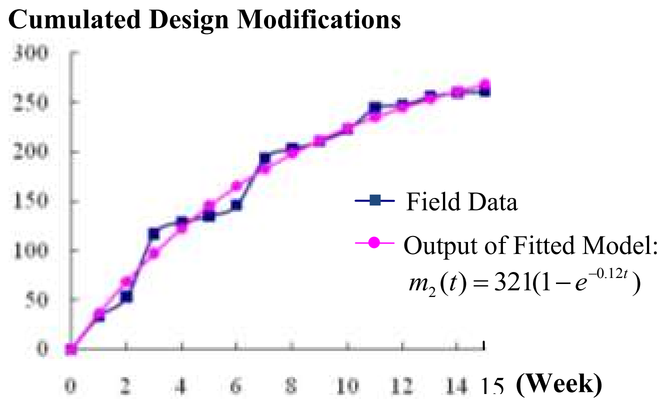


**Figure 3.7 Cumulated design modifications in system tests**

As an integral part of quality control system, all the design problems detected during tests, together with their severity level and root causes, their closure dates and the person in charge, the cost and time delay of each problem, were well documented in the company. Table 3.2 lists two examples of design problems detected in the *Design Evaluation Tests*. Such historical records provide us invaluable information on the problem identification and solution process (Lin et al., 2009). With the help of project managers, design engineers, and tooling engineers, we analyzed these problems in detail and derived the number of design modifications at two stages (See Table 3.3 and 3.4). Note that in practice, a design modification can be large or small according to their impact on downstream progress. To make different design modifications comparable, we together with the engineers standardized all the design modifications into dimensions. For example, if a slot is revised at time *t* and it affects 6 dimensions of the size and position of the slot, we record that 6 engineering changes occur at time *t*. Thus, here a design modification refers to the change of one dimension. Based on the collected data, we conducted a regression analysis to derive the functions of design modifications by minimizing Sum of Squared Residuals (SSE). The outputs of the regression analysis are shown in Figure 3.6 and Figure 3.7.

Then, we operationalized the estimation of $c_q$. In our application, after completion of *system tests*, the products were launched into the market. Thus, $c_q$ denotes the expected cost of removing one remaining design problem in the operational phase. When customers found design problems, the defective products were sent back to the company, and thus $c_q$ can be calculated based on the service record of the projects. Let $c$ be the expected cost of fixing one faulty product, $D$ be the cumulated number of defective products be sent back to the company, and $N$ be the overall number of design problems reported by the service center. Mathematically, $c_q = \frac{c \times D}{N}$ . The average value of $c_q$ in our data set is \$3,000.

**Table 3.5 Summary of other parameter values**

| Items | | Estimated value |
|---|---|---|
| $T_2$ | Regular duration for performing initial development at the Tooling Fabrication | 30 days |
| $w_2$ | Number of tasks at the Tooling Fabrication | 740 |
| $k$ | Percentage of Tooling Fabrication tasks affected by one detail design problem | 0.013 |
| $c_r$ | Rework cost at the Tooling Fabrication (per unit of time) | \$700 |
| $c_1$ | Unit testing cost at the Detail Design stage | \$500 |
| $c_2$ | Unit testing cost at the Tooling Fabrication | \$1,100 |
| $\beta$ | Opportunity cost (per unit of time) | Less than \$5,000 |

Finally, the estimations of other model parameters were relatively simple (Table 3.5). For instance, $T_2$ and $w_2$ can be directly obtained from the previous projects. The costs of rework were largely dominated by the engineering hours spent on the activities, and so $c_r$ was estimated by adding the average hourly wage of involved engineers. Testing costs were derived by adding the expenses of using equipment, materials and engineering resources. Parameter $\beta$ was obtained by interviewing project managers. They were asked to provide the values of what amount of money the company would lose, if the new product is delayed into the market for one month.

We then obtained the values of $\beta$, i.e. the financial loss of delaying the new product's time-to-market for one day.

## 3.5.2 Results

After getting the parameters, we applied the proposed method, and supplied management with the testing and overlapping strategies for the projects with different values of opportunity cost. The effect of inaccurate estimate of parameters was also evaluated and shown in Figure 3.8. In the worst case, all the estimated values were changed by 20% in the unfavorable direction, while in the best case, the parameter values were changed in favorable direction.

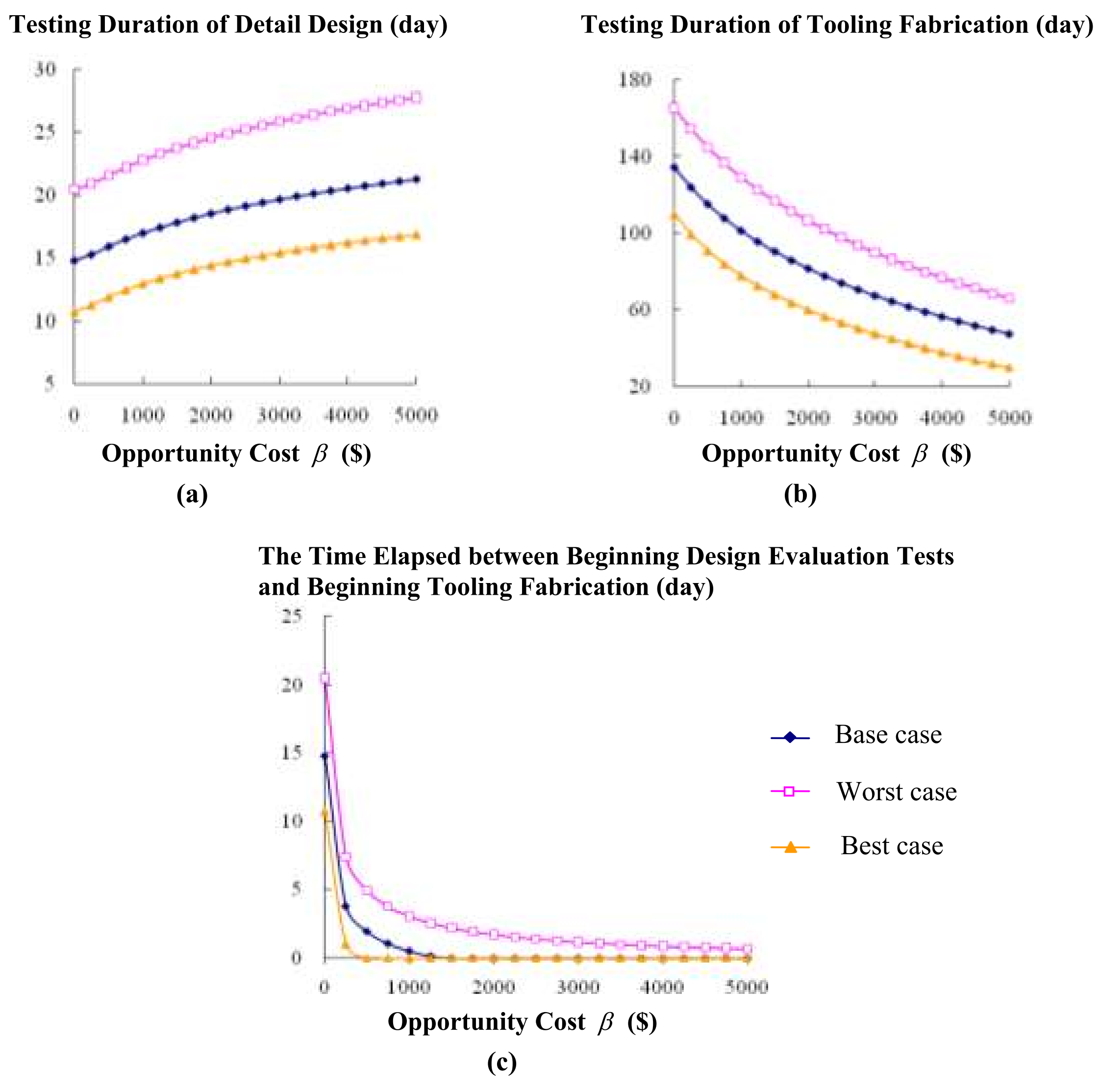


**Figure 3.8 Optimal solutions for projects with different opportunity cost**

Although the results may not be very accurate because of estimation errors of the parameters, they are sufficient to guide the management of similar development projects in the company. Previously, the average testing duration of Detail Design was 12 days. Our research shows that it should be increased, as illustrated in Figure 3.8(a). In the past, the start time of Tooling Fabrication was intuitively decided by the engineers and managers. The Tooling Fabrication starts one or two days before the completion of *Design Evaluation Tests*. In other words, low level of overlapping was applied for all the projects in the company. The overlapping policy made by intuition is far from optimal. Our research defines that even in the worst case, high level of overlapping should be adopted when $\beta > 1000$, as illustrated in Figure 3.8(c).

To facilitate the decision making of the management, we also plot the Pareto optimal fronts for the handset development projects. In particular, for a given opportunity cost $\beta$, the value of $c_q$ is changed from \$250 to \$6000, and the optimal $t_1$, $t_2$, and $t_s$ are then identified by the method developed in this study. The Pareto optimal fronts for $\beta = 0$, $\beta = 1{,}000$, $\beta = 3{,}000$ and $\beta = 5{,}000$ are shown in Figure 3.9.

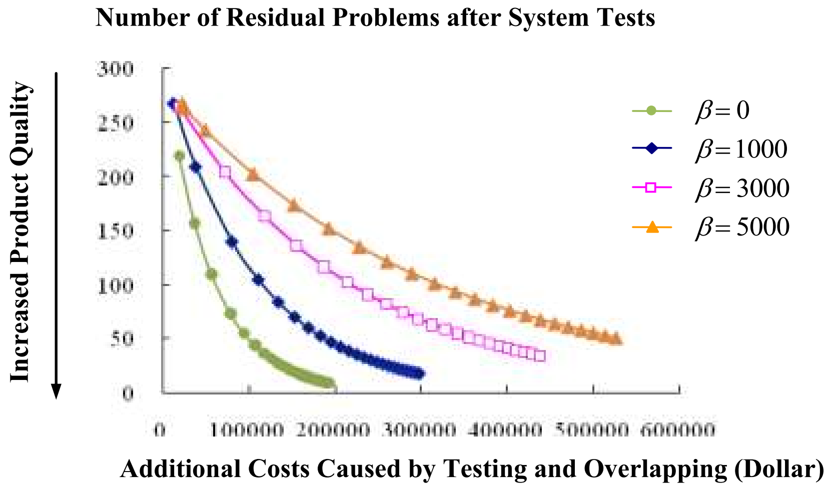


**Figure 3.9 Pareto optimal fronts for handset development projects**

## 3.6 Discussion and Conclusion

A product, to be competitive, needs to be introduced expeditiously without compromising product quality (Chakravarty 2001). Testing has been regarded as a primary way to detect design problems and improve product quality. However, testing can also be costly and time-consuming. Therefore, some researchers have developed models to determine the optimal scheduling of tests (e.g Ahmadi and Wang 1999; Thomke and Bell 2001; Serich 2005). We extend previous research by developing an analytical model for the optimal scheduling of tests in overlapped process which is a common practice to reduce project completion time. The propagation of design problems within development stages, as well as the cost-quality trade-offs of project performance, are explicitly studied.

Analysis of the model in this chapter yields following results:

- We provide closed-form expressions for the optimal time elapsed between beginning the upstream tests and beginning the downstream development, and the optimal downstream testing duration (Proposition 3.1).
- We find that the overall cost is first convex then concave increasing with respect to upstream testing duration, and prove that there exists a unique optimum that minimizes the overall cost (Proposition 3.2). Based on these properties, a one-dimensional search algorithm is proposed for solving the problem.
- The impact of different model parameters, such as problem-solving capacity and opportunity cost, on the optimal solution is discussed (Proposition 3.3, Corollaries 3.2-3.3). For instance, with higher upstream design problems and/or higher impact of upstream design change on downstream rework, the optimal upstream testing duration should be increased.

- We prove that the testing strategies in overlapped process differ from those in sequential process. Thus, to improve project performance, testing strategies and overlapping policies should be adjusted coordinately, and the results in Proposition 3.4 can be used as a guideline for structuring NPD processes.
- We present two variations of the cost minimization model, the *target quality problem* and the *budget constraint problem*, and show that both of these problems have similar solution structure as the *cost minimization problem* (Proposition 3.5).

It should be pointed out that there are two major limitations for the application of the results in this chapter. First, the testing process is modeled as a continuous NHPP process of discovering and solving design problems. To apply the NHPP model, the testing process must have independent and stationary increments. Specifically, numbers of design problem identified and resolved in disjoint intervals are independent, and the number of design problem identified and resolved in any interval of time depends only on the length of the testing time. The NHPP model is frequently made in models of quality and reliability (Xie et al., 2007), and in system dynamic models of NPD testing process (see e.g. Cooper, 1993a, 1993b, 1993c; Ford and Sterman, 1998; Serich, 2005; Lin et al., 2008; Love et al., 2008). It is justified that when design problems arise from many components or modules, each being a potential source of design modifications. Although the NHPP model has been proved to hold in many product development cases, there are still projects where this modeling may not appropriate, especially when the testing set-up time is long.

Second, as in previous papers (e.g. Ha and Porteus, 1995; Loch and Terwiesch, 1998; Chakravarty, 2001; Joglekar et al., 2001; Lin et al., 2009), we examine the concurrent execution of two consecutive development stages in this chapter. We call the first stage "upstream" and the second "downstream". For a development project consisting of a series of consecutive stages, our analysis and results are valid for any "upstream/downstream" overlap. For example, for a typical project consisting of four consecutive stages: (1) Concept Development, (2) Product Design, (3) Process Design, and (4) Pilot Production, the "upstream" stage could refer to stage 2, Product Design, and the "downstream" stage could refer to stage 3, Process Design; or the "upstream" stage could include stages 1 and 2, and the "downstream" stage could include stages 3 and 4. Nonetheless, in some cases decision makers may be more interested in the whole NPD process rather than in two stages.

Motivated by above two issues, in Chapter 4, we further explore the test scheduling problem for N-stage overlapped design process.

Future research may also be carried out in the areas discussed below. Firstly, like most previous studies that focus on the test scheduling problem, the present models are deterministic and therefore do not directly address risk. In the future, we may extend our model by taking into account the estimation errors of model parameters into decision making. Secondly, we assume the opportunity cost of time is constant. Although it should be constant in short run, it may change in long run. Hence, it may be worthwhile to investigate in detail how will the opportunity cost of time affect the product development performance, as well as the optimal testing and overlapping polices. Thirdly, like previous studies, e.g. Loch and Terwiesch (1998), Pham and Zhang (2003), Roemer and Ahmadi (2004), Huang and Lyu (2005), Ji et al. (2005), Wang and Yan (2005), Lin et al. (2009), we assume that the expected cost of removing one remaining design problem after downstream testing $c_q$ , and the percentage of downstream tasks affected by one upstream design problem *k*, are constant for all problems/faults. As mentioned earlier, to make different design problems/faults comparable, all design problems/faults are standardized into dimensions. Hence, such assumption is likely to hold in real situations. Finally, to keep the model simple, we assume that the overall testing cost is a linear function of the time spent on testing, and overall rework cost is a linear function of the amount of rework. As demonstrated by many studies, e.g. Thomke and Bell (2001), Loch et al. (2001), Chakravarty (2001), Wang and Yan (2005), Gerk and Qassim (2008), Lin et al. (2010), the linear functions are appropriate approximations of many real situations. Future research can relax these assumptions and discuss the corresponding testing and overlapping policies.

# CHAPTER 4

# SCHEDULING TESTS IN N-STAGE OVERLAPPED DESIGN PROCESS

## 4.1 Introduction

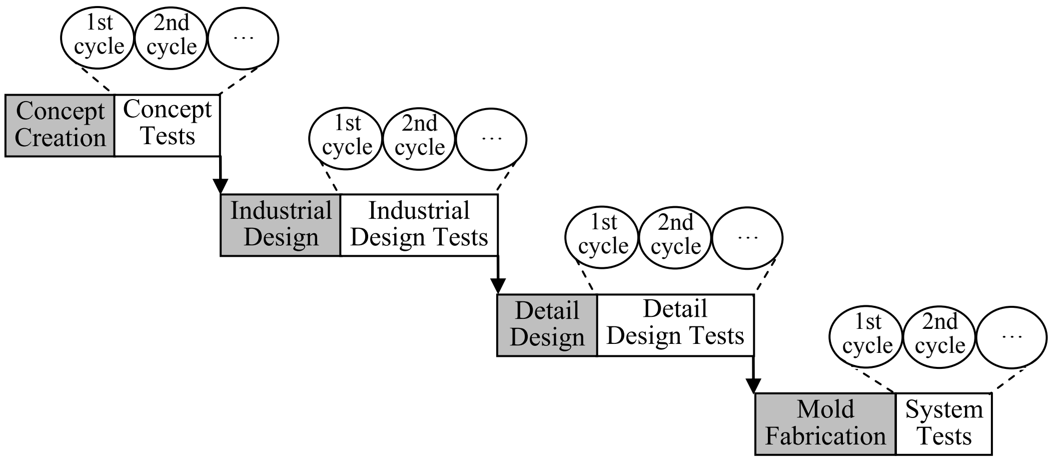
Testing has been regarded as an important tool for evaluating and improving product design. Its importance for successful NPD has been emphasized by many researchers (Loch and Terwiesch, 2001; Erat and Kavadias, 2008). At the same time, testing can also be expensive and time-consuming: researches have shown that testing activities can account for nearly half of total development effort (Cusumano and Selby, 1995; Thomke, 2003). Therefore, these studies highlighted a key trade-off involved in testing: the quality improvement versus additional time and cost spent on testing.

Some mathematical models have examined this key trade-off. We categorize them into two groups. The first group of studies assumed that the testing set-up time can be ignored, and modeled testing as a continuous process of discovering and solving design problems. Examples of such studies include Pham and Zhang (2003), Serich (2005), Love et al. (2008), Lin et al. (2008), Yang et al. (2008). These models have clearly shed light on the analysis of trade-off involved in testing, and are useful for guiding NPD projects where there is very little time wasted on setting up the tests. However, for projects with long testing set-up time, the methodologies and results may not be applied.

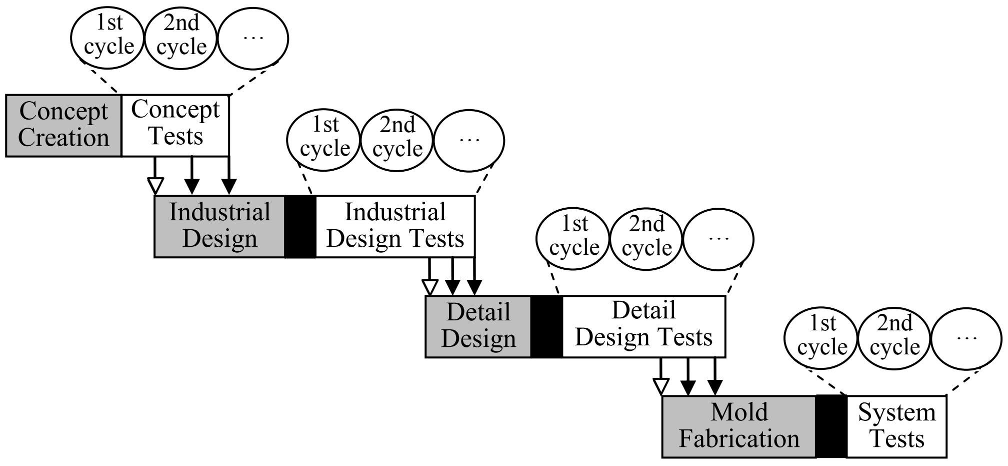
On the contrary, the second group of studies postulated that each time a test takes place, a certain amount of set-up time (such as the time to get organized for the test, to construct prototypes and to prepare documents) is required (Ha and Porteus, 1995; Thomke, 1998, 2003), and modeled testing as a discrete cyclic process. Examples of such studies include Ha and Porteus (1995), Dahan and Mendelson (2001), Erat and Kavadias (2008). Following this line of research, the model presented in this chapter considers testing set-up time and treats testing as a discrete cyclic process. The model investigates the test scheduling problem for N-stage overlapped design process, where downstream stages start before the completion of upstream tests. Such investigation is of practical importance since overlapping development stages has become a common mode of product development (Terwiesch et al., 2002; Gerk and Qassim, 2008; Gil et al., 2008). Moreover, the two decisions, test scheduling and overlapping policies, are interacted. Therefore, analytical models are needed to examine their combined effect on product development performance. However, the optimization problem becomes more complex, not only because the nonlinearity of functions involved, but also because the simultaneous presence of both discrete and continuous decision variables.

### 4.1.1 A Practical Example

Here we use a practical example to further illustrate the problem studied in this chapter. As shown in Figure 4.1, the refrigerator development usually consists of four stages: concept creation, industrial design, detail design, and mold fabrication. We distinguish between the initial design of these stages and testing activities that are carried out after the initial design. There are four types of tests which are performed in cycles. During each cycle, virtual or physical prototypes are built, tested to identify the design problems, and then modifications are made to remove the problems. As shown in Table 4.1, concept tests use CAD model to test customers’ reaction to the proposed new product. Industrial design tests build digital mockups to verify the feasibility of the industrial design. Detail design tests construct engineering prototypes to verify that the design can function, and finally system tests produce concrete refrigerators to improve the overall performance of the product. Completion of the system tests at last stage signals that the product is ready for volume production.



**(a) Sequential refrigerator development process**



**(b) Overlapped refrigerator development process**

| | | | |
|---|---|---|---|
| ▷ | Preliminary information | ▭ | Tests |
| ➤ | Modified information | ▨ | Initial design of development stages |
| | | ■ | Rework due to overlapping |

**Figure 4.1 Sequential and overlapped refrigerator development processes**

**Table 4.1 Prototype tests in the refrigerator development process**

| Prototype types | Form | Fidelity | Cost (Per cycle) |
|---|---|---|---|
| Concept Tests | CAD Model | Low | About $3,000 |
| Industrial Design Tests | Digital Mockups | Medium | About $4,000 |
| Detail Design Tests | Engineering Prototype | High | About $10,000 |
| System Tests | Physical Prototype | High | About $18,000 |

Typically, tests at earlier development stages (e.g. industrial design tests) are cheaper, but are of lower representativeness or fidelity, than the system tests at the last stage. Then, how much budget should be allocated to testing the initial design at each stage? Or how many rounds of tests should be conducted? Figure 4.1(a) shows the sequential refrigerator development process, where downstream stages (e.g. mold fabrication) start after the completion of upstream tests (e.g. detail design tests), and after most of the design problems are identified and resolved. While in overlapped process, as illustrated in Figure 4.1(b), downstream stages (e.g. mold fabrication) start earlier and can start at any time after the initial upstream design is available and before the completion of upstream tests (e.g. detail design tests). However, because downstream stages start on preliminary information, rework is often necessary to accommodate upstream design changes. Then, what is the optimal start time of downstream stages (e.g. mold fabrication)? If overlapping is applied, how should we adjust the testing strategies? In this chapter, we present an analytical model to answer these questions which are of concern to design managers.

The remainder of this chapter is organized as follows. Section 4.2 presents a mixed-integer programming model which captures the relationship between project properties, test scheduling, overlapping policies, and project profit. Section 4.3 introduces an algorithm to solve the problem. To illustrate the proposed methodology, we present a case study at a consumer electronics company in Section 4.4.

Discussions and conclusions are summarized in Section 4.5. Proofs of all the results in this chapter can be found in Appendix B.

## 4.2 Model Formulation

**Table 4.2 Symbols and decision variables**

| **Symbols** | **Definition** |
|---|---|
| $n$ | Number of development stages |
| $i$ | Index for development stages $(1 \le i \le n)$ |
| $d_i$ | Regular duration for initial design of stage $i$ |
| $a_i$ | Number of design problems introduced in the initial design of stage $i$ |
| $p_i$ | Testing quality |
| $t_i^s$ | Set-up time for a test in stage $i$ |
| $t_i^r$ | Average time for solving one design problem in stage $i$ |
| $b_i$ | Impact factor |
| $c_i^t$ | Testing cost in stage $i$ (per test) |
| $c_i^p$ | Penalty cost incurred per undetected design problem in stage $i$ |
| $c_i^h$ | Rework cost per time unit in stage $i$ due to overlapping |
| $\beta$ | Opportunity cost (per time unit) |
| $z_i$ | Number of tests completed in stage $i$ before starting stage $i+1$ |
| **Decision Variables** | **Definition** |
| $x_i$ | Number of tests to be conducted in stage $i$ |
| $y_{i+1}$ | Overlap between development stages $i$ and $i+1$ |
| $T_i(x_i)$ | Accumulated testing duration after $x_i$ rounds of tests in stage $i$ |
| $H_{i+1}(x_i, y_{i+1})$ | Rework duration in stage $i+1$ due to overlapping |

The model presented in this section examines the projects with a “nominal sequential” structure, i.e. the principal information exchange between consecutive stages is unidirectional: from upstream to downstream (Krishnan et al., 1997; Kogan and Raz, 2002; Lin et al., 2010). Moreover, to keep our analysis simple, we impose the additional constraint that at most two stages can be performed concurrently. Such model setting is consistent with previous papers (e.g. Roemer et al., 2000; Roemer and Ahmadi, 2004), and is an extension of Qian et al. (2007) that deals with two development stages.

Before formulating an analytic model, we define in Table 4.2 the notation used throughout the rest of this chapter. In the following subsections, we first formulate the trade-offs involved in scheduling tests for N-stage overlapped process. Then we establish the functional relationship between project properties and rework caused by overlapping development stages. The objective function and constraints are summarized at the end of this section.

### 4.2.1 Overview of the Model

Consider an NPD process consisting of a series of $n$ development stages. As previous papers (e.g. Cooper, 1993a, 1993b, 1993c; Ford and Sterman, 1998; Williams et al., 2003; Lin et al., 2008), we distinguish between the initial design of development stages and testing activities. When the initial design at each stage is finished, its output inevitably contains design problems (such as mismatches from customer needs or technical design faults), certain type of tests is then carried out to detect and remove these problems. Tests are conducted in cycles, and tests will not introduce design problems. The product is launched into the market after the completion of testing in the last stage $n$. Figure 4.1 shows an example of refrigerator NPD process with four development stages.

It is known that tests are often imperfect and cannot find all design problems through a single round of test. To capture this, let $a_i$ denote the number of design problems introduced in the initial design of stage $i$, $p_i$ denote the testing quality, i.e., the proportion of residual design problems detected by a single test, which is same as the testing fidelity in Thomke and Bell (2001), and testing quality in Ford and Sterman (1998), Lin et al. (2008). Then, in the first round of test in stage $i$, an amount of $a_i p_i$ design problems are detected and resolved, leaving $a_i(1-p_i)$ unsolved. In the second round of test, an amount of $a_i p_i(1-p_i)$ design problems are resolved, leaving $a_i(1-p_i)^2$ unsolved. In general, after $x_i$ rounds of tests, the overall number of design problems removed are: $a_i[1-(1-p_i)^{x_i}]$, and the residual design problems in stage $i$ are: $a_i(1-p_i)^{x_i}$.

In accordance with previous papers (e.g. Ha and Porteus 1995; Thomke and Bell 2001), we assume that if a design problem is detected in stage $i$, then an additional time $t_i^r$ must be spent on resolving the problem. Let $t_i^s$ be the set-up time for a test in stage $i$. The accumulated testing duration after $x_i$ rounds of tests in stage $i$, $T_i(x_i)$, can then be formulated as:

$$T_i(x_i) = t_i^s x_i + t_i^r a_i[1-(1-p_i)^{x_i}] \tag{4.1}$$

As stated earlier, for sequential process, downstream stages (e.g. mold fabrication) only start after the completion of upstream testing (e.g. detail design tests). However, in practice, downstream stages (e.g. mold fabrication) usually start earlier and before the completion of upstream testing (e.g. detail design tests) in order to reduce project completion time. Consequently, during the overlapped period, certain percentage of downstream design tasks is completed based on wrong information, and will have to be reworked when these design problems are discovered and resolved. Defining $H_{i+1}(x_i, y_{i+1})$ be the rework duration in stage $i$+1 caused by overlapping stages $i$ and $i$+1. We will formally establish this function in Section 4.2.2.

The trade-off in our model is as follows: increasing tests will decrease the residual design problems and improve design quality at that stage, but requires additional time and cost spending on the tests. If conducting too many repeated tests at one stage, the marginal benefit of design quality improvement may not be compensated for the time and cost required for the additional round of test. Overlapping helps to reduce project completion time, i.e. help to reduce part of the negative effect of testing, but requires additional effort (including time and cost) spent on downstream rework. To balance these tensions, we define project profit as the gain from conducting tests minus additional costs caused by testing and overlapping. The objective is thus to determine the optimal number of tests needed at each stage and the optimal overlap between consecutive stages, such that project profit will be maximized. To keep our analysis simple and aligned to previous papers, we define $c_i^p$ be the penalty cost for each undetected design fault in stage $i$. In practice, $c_i^p$ can be estimated as the cost of rework that design fault would cause in the operational phase (for more details, see e.g. Slaughter et al., 1998; Kogan and Raz, 2002; Ji et al., 2005). Clearly, without any test, the overall penalty costs would be: $\sum_{i=1}^{n} c_i^p a_i$ . Thus, the overall gain from conducting tests can be expressed as:

$$\sum_{i=1}^{n} \left[ c_i^p a_i - c_i^p a_i (1-p_i)^{x_i} \right] \tag{4.2}$$

We next consider the additional costs caused by testing and overlapping. Compared with the process where testing and overlapping are not applied, the additional costs associated with testing and overlapping include opportunity costs, costs of tests, and downstream rework costs. Clearly, the project completion time for the processes with testing and overlapping, can be represented as:

$$d_n + T_n(x_n) + \sum_{i=1}^{n-1}\left[d_i + T_i(x_i) - y_{i+1} + H_{i+1}(x_i, y_{i+1})\right] \tag{4.3}$$

It is clear that without testing and overlapping, the project completion time is: $\sum_{i=1}^{n} d_i$ . Let $\beta$ be the opportunity cost per time unit. As previous papers (e.g. Chakravarty, 2001; Wang and Yan, 2005; Lin et al. 2009), we assume $\beta$ is constant. Therefore, the overall opportunity costs is: $\beta\sum_{i=1}^{n-1}\left[T_i(x_i) - y_{i+1} + H_{i+1}(x_i, y_{i+1}) + T_n(x_n)\right]$.

Let $c_i^t$ be the cost for a test in stage $i$, $c_{i+1}^h$ be the rework cost per time unit in stage $i$+1 caused by overlapping. The additional costs caused by testing and overlapping are:

$$\sum_{i=1}^{n}\left[c_i^t x_i + \beta T_i(x_i)\right] + \sum_{i=1}^{n-1}\left[(c_{i+1}^h + \beta) H_{i+1}(x_i, y_{i+1}) - \beta y_{i+1}\right] \tag{4.4}$$

Thus, profit maximization objective can be represented as:

$$Max\ G = \sum_{i=1}^{n}\left[c_i^p a_i - c_i^p a_i (1-p_i)^{x_i} - c_i^t x_i - \beta T_i(x_i)\right] + \sum_{i=1}^{n-1}\left[\beta y_{i+1} - (c_{i+1}^h + \beta) H_{i+1}(x_i, y_{i+1})\right] \tag{4.5}$$

### 4.2.2 Rework due to Overlapping

In overlapped process, downstream stages (e.g. mold fabrication) start before the completion of upstream tests (e.g. detail design tests). Hence, some downstream tasks completed during the overlapped period, might based on incorrect information of upstream stages. When tests identify upstream design problems, we not only need to correct the design problems but also need to redo these downstream tasks. Consistent with previous papers (e.g. Loch and Terwiesch, 1998; Wang and Yan, 2005; Lin et al., 2009), we assume that the speed of performing the workload of a development stage (including initial design and rework caused by overlapping), is constant and equals one. Based on this assumption, the workload of a development stage can be described as duration, in other words, they are equivalent.

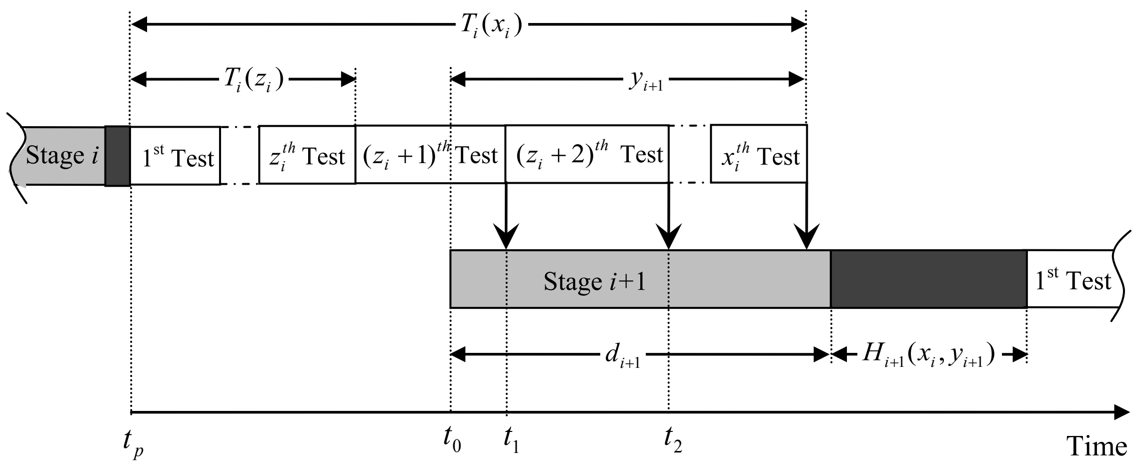
Previous model-based overlapping studies (e.g. Loch and Terwiesch, 1998; Chakravarty, 2003; Roemer and Ahmadi, 2004; Wang and Yan, 2005; Lin et al., 2009) have shown that the amount of rework caused by overlapping is decided by: the number of upstream design changes, the impact of each change, and the downstream progress, i.e. the number of downstream tasks completed when the upstream design change arrives. Following these studies, we define $b_i$ be the impact factor, i.e. the percentage of tasks in stage $i+1$ that will be affected by one design change in stage $i$.

Figure 4.2 graphically represents the impact of design changes in stage $i$ on the downstream rework in stage $i+1$. For easier explanation, suppose that preliminary information of upstream stage $i$ is available at time $t_p$, and downstream stage $i+1$ starts at time $t_0$. Let $y_{i+1}$ be the overlap between stages $i$ and $i+1$, where $0 \le y_{i+1} \le T_i(x_i)$, $z_i$ be the number of tests completed in stage $i$ before starting stage $i+1$. It is clear that during the overlapped period, there are $(x_i - z_i)$ bursts of design changes transferred from stage $i$ to stage $i+1$. To ensure that all design changes are absorbed in the initial design of stage $i+1$, tests in stage $i$ should be terminated before the completion of initial design in stage $i+1$, i.e. $y_{i+1} \le d_{i+1}$.

After the completion of $(z_i+1)^{th}$ test in stage $i$, the first batch of design changes are transferred to stage $i+1$ at time $t_1$. At that time, the progress of stage $i+1$ is: $[y_{i+1} + T_i(z_i+1) - T_i(x_i)]$. Then, if one design change of stage $i$ arrives at $t_1$, the amount of rework caused in stage $i+1$ will be $b_i[y_{i+1} + T_i(z_i+1) - T_i(x_i)]$. In the

$(z_i+1)^{th}$ test of stage $i$, an expected number of $a_i p_i (1-p_i)^{z_i}$ design problems are identified and resolved, and the amount of rework required in stage $i+1$ is: $a_i b_i p_i (1-p_i)^{z_i}[y_{i+1}+T_i(z_i+1)-T_i(x_i)]$. Similarly, there are $a_i p_i (1-p_i)^{z_i+1}$ design changes arrive at time $t_2$, and the expected rework caused by these changes is $a_i b_i p_i (1-p_i)^{z_i+1}[y_{i+1}+T_i(z_i+2)-T_i(x_i)]$. In general, rework duration in stage $i+1$ caused by overlapping stages *i* and $i+1$, $H_{i+1}(x_i, y_{i+1})$, can be represented as:

$$H_{i+1}(x_i, y_{i+1}) = \sum_{k=1}^{(x_i - z_i)} a_i b_i p_i (1-p_i)^{z_i+k-1}[y_{i+1}+T_i(z_i+k)-T_i(x_i)] \tag{4.6}$$



| | | | |
|---|---|---|---|
| → | Design changes | $t_p$ | Preliminary information available |
| ▭ | Tests | $t_0$ | Starting time of stage *i*+1 |
| ▨ | Initial design of development stages | $t_1$ | The arrival time of first batch design changes |
| ■ | Rework due to overlapping | $t_2$ | The arrival time of second batch design changes |
| $T_i(x_i)$ | Accumulated testing duration after $x_i$ rounds of tests in stage *i* | $z_i$ | Number of tests completed in stage *i* before start stage *i*+1 |
| $T_i(z_i)$ | Accumulated testing duration after $z_i$ rounds of tests in stage *i* | **Decision Variables** | |
| $d_{i+1}$ | Regular duration for initial design of stage *i*+1 | $x_i$ | Number of tests to be conducted in stage *i* |
| $H_{i+1}(x_i, y_{i+1})$ | Rework duration in stage *i*+1 due to overlapping stages *i* and *i*+1 | $y_{i+1}$ | Overlapped period between stages *i* and *i*+1 |

**Figure 4.2 Rework in stage $i+1$ caused by overlapping stages *i* and $i+1$**

### 4.2.3 Summary

Based on the elements discussed above, we summarize the optimization problem as following non-linear mixed-integer program:

$$\left.\begin{array}{l}
Max\, G = \sum_{i=1}^{n}\left\{c_i^p a_i[1-(1-p_i)^{x_i}] - c_i^t x_i - \beta T_i(x_i)\right\} + \sum_{i=1}^{n-1}\left[\beta y_{i+1} - (c_{i+1}^h + \beta) H_{i+1}(x_i, y_{i+1})\right] \\
\text{Subject to: Equations (4.1) and (4.6),} \\
x_i \text{ nonnegative integer, for } i = 1,2,\ldots,n \\
y_{i+1} \text{ nonnegative real number, for } i = 1,2,\ldots,n-1 \\
y_{i+1} \le d_{i+1} \quad \text{for } i = 1,2,\ldots,n-1 \\
y_{i+1} \le T_i(x_i) \quad \text{for } i = 1,2,\ldots,n-1
\end{array}\right\} \quad (4.7)$$

In the optimization problem (4.7), the objective is to maximize project profit by selecting appropriate number of tests $x_i$ and appropriate overlap $y_{i+1}$. The functions of $T_i(x_i)$ and $H_{i+1}(x_i, y_{i+1})$ are established in Equations (4.1) and (4.6), respectively. The constraints $y_{i+1} \le T_i(x_i)$ and $y_{i+1} \le d_{i+1}$ defines the maximum overlap between stages *i* and $i+1$.

## 4.3 Analysis of Testing and Overlapping Policies

The difficulty for solving problem (4.7) lies not only in the nonlinearity of functions involved, but also in the simultaneous presence of both discrete and continuous decision variables. Moreover, the objective function in the optimization problem is non-concave, which further increases the difficulty of finding optimal solutions. In this section, we will describe how to solve problem (4.7).

**PROPOSITION 4.1** *Given $x_i^*$ and $z_i^*$, downstream stage starts directly after the* *completion of an upstream test, i.e., for* $i = 1,2,\ldots,n-1$,

$$y_{i+1}^* = T_i(x_i^*) - T_i(z_i^*) = t_i^s(x_i^* - z_i^*) + t_i^r a_i[(1-p_i)^{z_i^*} - (1-p_i)^{x_i^*}] \qquad (4.8)$$

The proof of all the results in this chapter can be found in Appendix B.

Proposition 4.1 shows that it is optimal to start initial design of downstream stage $i+1$ directly after the completion of a test in stage *i*. This is so because otherwise the gain from overlapping development stages will be decreased, without any other changes. Based on the result of Proposition 4.1, (4.6) can be rewritten as:

$$H_{i+1}(x_i, z_i) = t_i^s a_i b_i \left[ (z_i - x_i)(1-p_i)^{x_i} + \frac{(1-p_i)^{z_i} - (1-p_i)^{x_i}}{p_i} \right]$$

$$+ t_i^r a_i^2 b_i \left[ \frac{(1-p_i)^{2z_i} + (1-p_i)^{2x_i+1}}{2-p_i} - (1-p_i)^{x_i+z_i} \right] \qquad (4.9)$$

The mixed-integer problem in (4.7) can then be simplified as following integer program over decision variables $x_i$ and $z_i$:

$$Max\ G = \sum_{i=1}^{n} \left\{ c_i^p a_i[1-(1-p_i)^{x_i}] - c_i^t x_i \right\} - \beta\left\{ t_n^s x_n - t_n^r a_n[1-(1-p_n)^{x_n}] \right\}$$

$$-\sum_{i=1}^{n-1} \left\{ \beta t_i^s z_i + \beta t_i^r a_i[1-(1-p_i)^{z_i}] + (\beta + c_{i+1}^h) H_{i+1}(x_i, z_i) \right\} \qquad (4.10)$$

Subject to: Equation (4.9),

$0 \le z_i \le x_i$, for $i = 1,2,\ldots,n-1$ (4.11)

$t_i^s(x_i - z_i) + t_i^r a_i[(1-p_i)^{z_i} - (1-p_i)^{x_i}] \le d_{i+1}$, for $i = 1,2,\ldots,n-1$ (4.12)

$z_i$ ($i = 1,2,\ldots,n-1$) and $x_i$ ($i = 1,2,\ldots,n$) are integers (4.13)

In the optimization problem (4.10)-(4.13), the testing strategies are determined by $x_i$, i.e., the number of tests to be conducted at each stage, and the overlapping polices are determined by $z_i$, i.e., the number of upstream tests completed before starting the downstream stage. Our next result characterizes $z_i^*$ that solves the optimization problem, by assuming that $x_i^*$ has been derived.

**PROPOSITION 4.2** *For $i = 1,2,\ldots,n-1$, given $x_i^*$,*

*(a) $z_i^*$ can be decided as follows: if $F_1 \le 0$, then $z_i^* = z_i^l$; if $F_2 \ge 0$, then $z_i^* = x_i^*$; otherwise, setting $\frac{\partial G}{\partial z_i} = 0$ yields a unique value of $z_i^c$, and $z_i^* = \lceil z_i^c \rceil$ or $z_i^* = \lceil z_i^c \rceil - 1$.*

*(b) $z_i^c$ increases with $a_i$, $b_i$, $t_i^s$, and decreases with $t_i^r$.*

*In the above, $z_i^l$, $F_1$ and $F_2$ are defined in Appendix B.*

Proposition 4.2(a) characterizes the optimal number of tests completed in stage $i$ before starting stage $i+1$, $z_i^*$, when $x_i^*$ is fixed. If $F_1 \le 0$, then project profit in (4.10) decreases monotonically in $z_i$, leading to $z_i^* = z_i^l$, in other words, it's optimal to start initial design of stage $i+1$ as early as possible. On the other hand, if $F_2 \ge 0$, then the project profit increases monotonically in $z_i$, and so $z_i^* = x_i^*$, i.e., it is optimal to perform stages $i$ and $i+1$ sequentially. Otherwise, the project profit first increases then decreases in $z_i$, and is maximized at $z_i^c$. $z_i^*$ can then be identified through comparing project profits at $\lceil z_i^c \rceil$ and $\lceil z_i^c \rceil - 1$.

Proposition 4.2(b) discusses the impact of parameters on $z_i^c$. With higher initial design problems in stage *i*, $a_i$, higher impact factor $b_i$, and higher testing set-up time in stage *i*, $t_i^s$, the company should delay the start time of stage $i+1$, i.e. reduce the overlap between stages $i$ and $i+1$. Slower problem solving in stage $i$, $t_i^r$, also reduce the optimal overlap.

In some real situations, the overlapping policies are fixed, and managers are faced with the problem of determining the number of tests to be conducted at each stage. Our next proposition can help managers make such decisions.

**PROPOSITION 4.3** *For $i = 1,2,\ldots,n-1$, given $z_i^*$,*

*(a) The optimal number of tests to be conducted in stage $i$, $x_i^*$, can be decided as follows: if $F_3 \le 0$, then $x_i^* = z_i^*$; if $F_4 \ge 0$, then $x_i^* = x_i^u$; otherwise, setting $\frac{\partial G}{\partial x_i} = 0$ yields a unique value of $x_i^c$, and $x_i^* = \lceil x_i^c \rceil$ or $x_i^* = \lceil x_i^c \rceil - 1$.*

*(b) $x_i^c$ decreases with $t_i^r$, $t_i^s$, and $b_i$.*

*(c) If $a_i \ge \bar{a}_i$, then $x_i^c$ decreases with $a_i$. Otherwise if $a_i < \bar{a}_i$, there exists a unique $\bar{x}_i$ such that: if $x_i^c < \bar{x}_i$, $x_i^c$ increases with $a_i$; if $x_i^c > \bar{x}_i$, $x_i^c$ decreases with $a_i$.*

*In the above, $F_3$, $F_4$, $x_i^u$, $\bar{a}_i$, and $\bar{x}_i$ are defined in Appendix B.*

Proposition 4.3(a) describes the optimal $x_i^*$ when $z_i^*$ is fixed. If $F_3 \le 0$, then the project profit in (4.10) decreases monotonically in $x_i$, leading to $x_i^* = z_i^*$. On the other hand, if $F_4 \ge 0$, then $x_i^* = x_i^u$, i.e. it is optimal to conduct as many tests in stage $i$ as possible. Otherwise, the project profit first increases then decreases in $x_i$, and is maximized at $x_i^c$. The optimal $x_i^*$ can then be derived through comparing project profits at $\lceil x_i^c \rceil$ and $\lceil x_i^c \rceil - 1$. Proposition 4.3(b) and 4.3(c) discuss the impact of parameters on $x_i^c$. Part (b) shows that with faster problem solving $t_i^r$, longer set-up time of a test $t_i^s$, and/or higher impact factor $b_i$, the number of tests in stage $i$ should be reduced. Part (c) shows that higher initial design problems $a_i$ may increase or decrease the optimal number of tests needed in stage $i$. Based on the result of Proposition 4.3(b) and 4.3(c), the decision makers can easily adjust the testing strategies for NPD projects with different characteristics.

We have shown the respective effect of testing strategies or overlapping policies on project profit. However, testing strategies and overlapping policies are interacted, and their combined effect on project profit is much more complex. In the rest of this section, we will discuss how to solve the optimization problem.

**PROPOSITION 4.4**

*(a) $x_i^*$ must satisfy the following condition:*

$$x_i^* < \frac{c_i^p a_i}{c_i^t} \qquad \text{for } i = 1,2,\ldots,n-1 \tag{4.14}$$

*(b) If $t_n^r\beta < c_n^p$, else if $c_n^t + t_n^s\beta \geq (t_n^r\beta - c_n^p)a_n \ln(1-p_n)$, then $x_n^* = 0$; otherwise, $x_n^* = \lfloor x_n^c \rfloor$ or $x_n^* = \lceil x_n^c \rceil - 1$.*

*In the above,* $$x_n^c = \frac{\ln(c_n^t + t_n^s\beta) - \ln[(t_n^r\beta - c_n^p)a_n \ln(1-p_n)]}{\ln(1-p_n)} \tag{4.15}$$

Proposition 4.4(a) defines the upper bound of $x_i^*$ for $i = 1,2,\ldots,n-1$. To maximize project profit, testing costs should be less than the potential benefit, and so we get equation (4.14). Proposition 4.4(b) characterizes the optimal number of tests to be conducted at final stage $n$, $x_n^*$. Defining $G^*$ be the optimal project profit, and $G^*(x_i)$ be the optimal project profit for a given value of $x_i$. We now introduce the following algorithm 4.1 to solve the optimization problem:

**Algorithm 4.1**

*Step 1*. Determine $x_n^*$: If $t_n^r\beta < c_n^p$ , else if $c_n^t + t_n^s\beta \geq (t_n^r\beta - c_n^p)a_n \ln(1-p_n)$, then $x_n^* = 0$; Else, calculate $x_n^c$, compare $\lceil x_n^c \rceil$ and $\lceil x_n^c \rceil - 1$, and identify the one yields higher project profit.

For $i = 1,2,\ldots,n-1$, do:

*Step 2* . Set $x_i^* = 0$ and $z_i^* = 0$, calculate $G^*$. Let $x_i = 1$.

*Step 3*. If $x_i > \dfrac{c_i^p a_i}{c_i^t}$, stop.

*Step 4*. For the current value of $x_i$, determine $z_i^*$ according to Proposition 4.2(a).

(a) Identify $z_i'$ that satisfies $t_i^s(x_i - z_i') + t_i^r a_i[(1-p_i)^{z_i'} - (1-p_i)^{x_i}] = d_{i+1}$ . Let $z_i^l = \max\{0, \lceil z_i' \rceil\}$.

(b) If $F_1 \leq 0$, then $z_i = z_i^l$, calculate $G^*(x_i)$, go to *Step 5*; if $F_2 \geq 0$, then $z_i = x_i$, calculate $G^*(x_i)$, go to *Step 5*. Else, identify the unique $z_i^c$ that satisfies $\dfrac{\partial G}{\partial z_i} = 0$. Compare $\lceil z_i^c \rceil$ and $\lceil z_i^c \rceil - 1$, and identify the one yields higher project profit $G^*(x_i)$.

*Step 5*. If $G^*(x_i) > G^*$, then $x_i^* = x_i$, $z_i^* = z_i$, and $G^* = G^*(x_i)$.

*Step 6*. $x_i = x_i + 1$. Go to *Step 3*.

In *Step 1* of above algorithm, we determine $x_n^*$ according to Proposition 4.4(b). The main loop from *Step 2* to *Step 6* identifies the optimal testing strategies and overlapping policies, $x_i^*$ and $z_i^*$, for $i = 1,2,\ldots,n-1$. We first let $x_i^* = 0$, $z_i^* = 0$, and get an initial $G^*$ in *Step 2*. Then, in *Step 3*, we check whether $x_i$ exceeds its upper bound, and in *Step 4*, we determine $z_i^*$ for the current value of $x_i$ according to

Proposition 4.2(a). *Step 5* compares $G^*(x_i)$ with $G^*$. If $G^*(x_i)$ is larger, then $x_i^*$ and $z_i^*$ will be adjusted accordingly. $x_i^*$ and $z_i^*$ can then be identified after all possible values of $x_i$ are examined.

## 4.4 Case Study

In this section, we apply the model to the refrigerator design process at a consumer electronics company. The corporation manufactures a wide range of household electrical appliances, such as refrigerators, air conditioners, washing machines, televisions, and so on, and its global revenue in 2005 was more than 12 billion US dollars. There are six types of refrigerator projects in the company, one of which is analyzed in this section to illustrate the proposed methodology. The main components of the refrigerator are shown in following Figure 4.3.

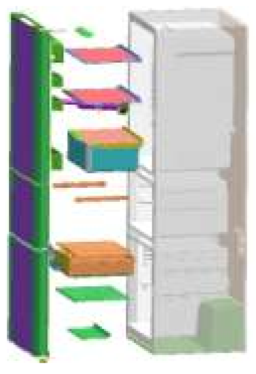


**Figure 4.3 Main Components of the Refrigerator**

The refrigerator development process has been shown in Figure 4.1. In the refrigerator development, four types of tests: concept tests, industrial design tests, detail design tests, and system tests, are carried out, and the main characteristic of these tests have been listed in Table 4.1.

### 4.4.1 Data Collection

In order to illustrate our model, we collected detailed data based on historical records, such as project schedule and documents of design problems detected and resolved over the entire period of the project. These data were double checked together with the engineers familiar with this project. In most companies, these data are available which allows us to derive the parameter values with reasonable validity (Krishnan et al., 1997; Kogan and Raz, 2002; Jun et al., 2005; Gerk and Qassim, 2008).

The regular duration for initial design $d_i$, and testing set-up time $t_i^s$, were directly derived from the project schedule. Consistent with previous studies (e.g. Cooper, 1993b, 1993c; Thomke and Bell, 2001; Lin et al., 2008; Love et al., 2009), the initial design problems $a_i$ , testing quality $p_i$, and average time for solving one design problem $t_i^r$ were derived based on the historical records of the quality problems found and solved over the entire period of the project.

As is common in consumer electronics industry, concurrent engineering was applied in the case study company. For example, two rounds of detail design tests were conducted in the project, and mold fabrication started after the completion of first round of test. In the second round of test, more than 80 detail design problems were resolved and transferred, resulting in about 15% of rework in mold fabrication. We then derived the impact of each modification $b_3$ 0.18%. We got $b_i$ for other stages in the same way (Consistent with Krishnan et al., 1997; Roemer et al., 2000; Jun et al., 2005; Lin et al., 2009).

The other parameters are commonly used in past models (e.g. Slaughter et al., 1998; Thomke and Fujimoto, 2000; Kogan and Raz, 2002; Lin et al., 2008).

Following these studies, we got these parameters together with engineers who were familiar with the project. For example, the costs of a test $c_i^t$ were derived by adding the expenses of building prototypes, using equipment, materials and engineering resources. The penalty cost per undetected design problem $c_i^p$ was estimated as the cost of rework that design problem would cause in the operational phase. The rework cost per time unit were largely dominated by the engineering hours spent on the activities, and so $c_i^h$ was estimated by adding the average hourly wage of the involved engineers from industrial design, mechanic design, hardware design, and so on. We summarize parameter values of the project in Table 4.3.

Finally, we checked the testing and overlapping policies adopted by the company. For the previous projects, three rounds of concept tests, two rounds of industrial design tests and detail design tests, and three rounds of system tests were conducted. Usually, the industrial design started after the completion of two rounds of concept tests, the detail design and mold fabrication started after the completion of one round of industrial design test, and one round of detail design test, respectively.

**Table 4.3 Model inputs for the refrigerator development project**

| **Stages** | $d_i$ (day) | $a_i$ | $p_i$ | $t_i^s$ (day) | $t_i^r$ (day) | $b_i$ | $c_i^t$ ($\times 10^3$) | $c_i^p$ ($\times 10^3$) | $c_i^h$ ($\times 10^3$) |
|---|---|---|---|---|---|---|---|---|---|
| Concept Creation | 9 | 47 | 0.4 | 3 | 0.229 | 1.4% | 3 | 20 | - |
| Industrial Design | 12 | 120 | 0.52 | 12 | 0.086 | 0.6% | 4 | 10 | 0.35 |
| Detail Design | 20 | 420 | 0.7 | 4 | 0.103 | 0.18% | 10 | 5 | 0.8 |
| Mold Fabrication | 40 | 259 | 0.75 | 6 | 0.082 | - | 18 | 3.5 | 7 |

## 4.4.2 Results and Sensitivity Analysis

After deriving the data, we investigated the optimal testing and overlapping policies using the Algorithm 4.1 in Section 4.3. The optimal testing strategies identified are: $x_1^* = 6$, $x_2^* = 2$, $x_3^* = 4$, $x_4^* = 2$. Previously, three rounds of concept tests, two rounds of industrial design tests, two rounds of detail design tests, and three rounds of system tests were conducted in the project, i.e., $x_1 = 3$, $x_2 = 2$, $x_3 = 2$, $x_4 = 3$. Our study shows that the company should add three more rounds of concept tests and two more rounds detail design tests, while reduce system tests to two rounds.

The optimal overlapping policies identified by Algorithm 4.1 are: $z_1^* = 3$, $z_2^* = 1$, $z_3^* = 1$, i.e., it is optimal to start industrial design after three rounds of concept tests, and to start detail design after one round of industrial design test, and start mold fabrication after one round of detail design test. Compared with the existing overlapping policy, i.e. $z_1 = 2$, $z_2 = 1$, $z_3 = 1$, the company should delay the start of industrial design. The benefit from the optimal testing and overlapping policies identified by our research can be increased from US$ 2,725,600 to US$ 3,007,500. Meanwhile, the project completion time can be decreased slightly from 190 days to 189 days.

The sensitivity of the optimal solution to cost parameters, including penalty cost per undetected design problem $c_i^p$, opportunity cost per time unit $\beta$, cost for a test $c_i^t$, and rework cost per time unit $c_i^h$, were evaluated. Note that changing the value of $c_i^h$ from 50% to 200% will not affect the optimal solution, and changing the value of $c_i^t$ will not affect $z_i^*$. The results are presented in Table 4.4-4.6, which suggest that the sensitivity of the optimal solution with respect to cost parameters is relatively low. For example, as shown in Table 4.4, the optimal solution is not affected when $c_i^t$ change from 125% to 200%.

**Table 4.4 Impact of testing cost on optimal testing policies**

| **Stages** | 50% $c_i^t$ | 75% $c_i^t$ | 100% $c_i^t$ | 125% $c_i^t$ | 150% $c_i^t$ | 175% $c_i^t$ | 200% $c_i^t$ |
|---|---|---|---|---|---|---|---|
| Concept Creation | 6 | 6 | 6 | 6 | 6 | 6 | 6 |
| Industrial Design | 2 | 2 | 2 | 2 | 2 | 2 | 2 |
| Detail Design | 5 | 5 | 4 | 4 | 4 | 4 | 4 |
| Mold Fabrication | 2 | 2 | 2 | 1 | 1 | 1 | 1 |

**Table 4.5 Impact of $c_i^p$ on the optimal solution**

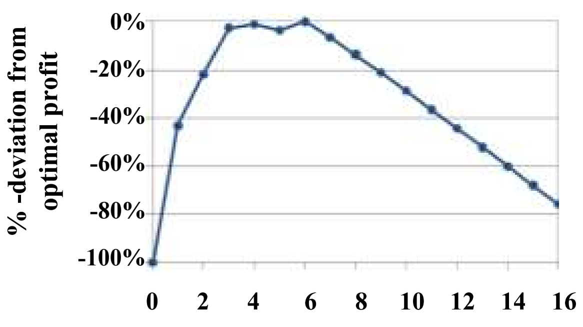
| **Stages** | 50% $c_i^p$ | | 75% $c_i^p$ | | 100% $c_i^p$ | | 125% $c_i^p$ | | 150% $c_i^p$ | | 175% $c_i^p$ | | 200% $c_i^p$ | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | $x_i^*$ | $z_i^*$ | $x_i^*$ | $z_i^*$ | $x_i^*$ | $z_i^*$ | $x_i^*$ | $z_i^*$ | $x_i^*$ | $z_i^*$ | $x_i^*$ | $z_i^*$ | $x_i^*$ | $z_i^*$ |
| Concept Creation | 3 | 1 | 3 | 1 | 6 | 3 | 6 | 3 | 6 | 3 | 6 | 3 | 6 | 3 |
| Industrial Design | 1 | 0 | 2 | 1 | 2 | 1 | 2 | 1 | 3 | 2 | 3 | 2 | 3 | 2 |
| Detail Design | 4 | 1 | 4 | 1 | 4 | 1 | 5 | 1 | 5 | 1 | 5 | 1 | 5 | 1 |
| Mold Fabrication | 0 | - | 1 | - | 2 | - | 2 | - | 2 | - | 2 | - | 2 | - |

**Table 4.6 Impact of opportunity cost on the optimal solution**

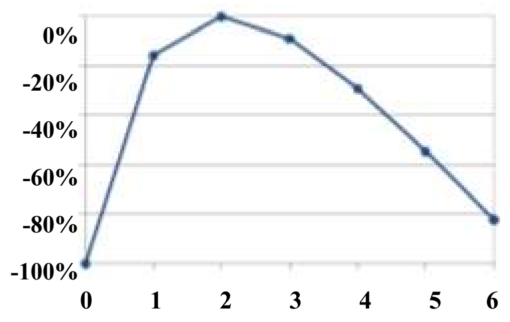
| **Stages** | 50% $\beta$ | | 75% $\beta$ | | 100% $\beta$ | | 125% $\beta$ | | 150% $\beta$ | | 175% $\beta$ | | 200% $\beta$ | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | $x_i^*$ | $z_i^*$ | $x_i^*$ | $z_i^*$ | $x_i^*$ | $z_i^*$ | $x_i^*$ | $z_i^*$ | $x_i^*$ | $z_i^*$ | $x_i^*$ | $z_i^*$ | $x_i^*$ | $z_i^*$ |
| Concept Creation | 6 | 3 | 6 | 3 | 6 | 3 | 3 | 1 | 3 | 1 | 3 | 1 | 3 | 1 |
| Industrial Design | 3 | 2 | 2 | 1 | 2 | 1 | 2 | 1 | 1 | 0 | 1 | 0 | 1 | 0 |
| Detail Design | 5 | 1 | 4 | 1 | 4 | 1 | 4 | 1 | 4 | 1 | 4 | 1 | 4 | 1 |
| Mold Fabrication | 2 | - | 2 | - | 2 | - | 1 | - | 1 | - | 1 | - | 1 | - |

### 4.4.3 Combined Effect of Testing and Overlapping on Project Profit

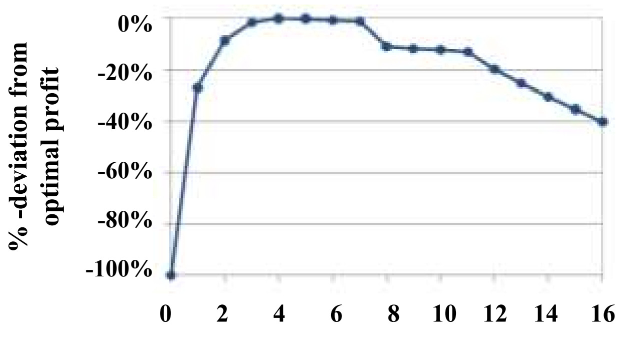
We solve a series of numerical examples in this section to illustrate the complex relationship between project profits, and testing and overlapping policies. Figure 4.4(a) shows that the project profit is "M shaped" with the increase of concept tests. Figure 4.4(b) and 4.4(c) indicate that project profit first increases then decreases as the number of industrial design tests or detail design tests increase.



**(a) Number of Concept Tests $x_1$**



**(b) Number of Industrial Design Tests $x_2$**



**(c) Number of Design Evaluation Tests $x_3$**

**Figure 4.4 Combined effect of testing and overlapping on project profit**

## 4.5 Discussion and Conclusion

In Chapter 3, we have presented an analytical model for the scheduling of tests in overlapped design process, where a downstream stage starts before the completion of upstream testing. In this chapter, we further explore the test scheduling problem for N-stage overlapped design process. The model presented in this chapter can be used to determine the optimal number of tests needed at each development stage, together with the optimal overlapping policies. It complements the work in Chapter 3 by taking testing set-up time into consideration, and by examining the testing and overlapping polices for the whole NPD process.

Analysis of the model in this chapter yields following results some of which may not be intuitive:

- Proposition 4.1 shows that it is optimal to start downstream stages directly after the completion of an upstream test. This is so because otherwise the gain from overlapping development stages will be decreased, without any other changes.
- Proposition 4.2 describes the optimal overlapping policies when the number of tests is predetermined. It also shows that with higher initial design problems, higher impact factor, higher test set-up time, and/or slower problem solving, the company should delay the downstream initial design.
- In some real situations, the overlapping policies are fixed, and managers are faced with the problem of determining the number of tests at each stage. Proposition 4.3 can help managers make such decisions. The impact of model parameters on the optimal testing policies is also discussed.
- Finally, an efficient algorithm is proposed for solving a practical problem that has not been addressed so far in the literature.

Nonetheless, we would like to point out following limitations for application of the results in this chapter:

- Firstly, our model can only be built and analyzed when historical data is available. For derivative projects which account for the majority of product development projects (Leifer et al., 2000; Jun and Suh, 2008), these data are typically available (see e.g. Krishnan et al., 1997; Thomke and Fujimoto, 2000; Jun et al., 2005). However, for totally new projects, we can only build a model based on estimated data which may lead to inaccurate results. For instance, if a manager wants to apply our methodologies to improve the refrigerator development process, then he/she should collect the data from previous similar refrigerator development projects.
- Secondly, in the data collection process, we find that many firms are unwilling to share their product development experiences for security reasons. Even for the firm

that has participated in our study, the data collection was quite challenging and time-consuming since the companies did not know how past project data could be used in guiding future projects. Therefore, in spite of our best effort, we were able to do only a case study of refrigerator development. In the future, it is meaningful to apply the proposed methodologies to other development projects and processes.

- Thirdly, our model is deterministic and therefore does not directly address risk, although we have addressed many facets of risk through sensitivity analysis of the optimal solution to parameters.

- Finally, to keep the model simple, we assume that the overall penalty cost is a linear function of the number of remaining design faults, and overall rework cost is a linear function of the amount of rework. Even with such linear assumptions, our model is complex and is difficult to solve. Moreover, the linear functions are appropriate approximations of many real situations, as demonstrated by many studies (see e.g. Kogan and Raz, 2002; Roemer and Ahmadi, 2004; Ji et al., 2005; Wang and Yan, 2005; Gerk and Qassim, 2008; Lin et al., 2009). Future research can relax these assumptions and discuss the corresponding testing and overlapping policies.

# CHAPTER 5

# A DECOMPOSITION APPROACH FOR SEQUENCING DESIGN ACTIVITIES

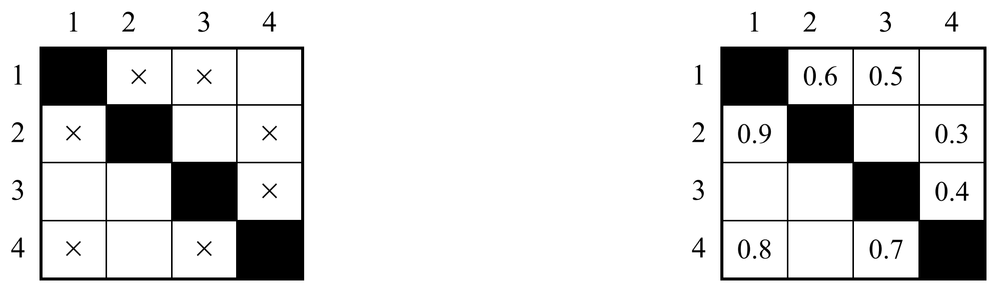
To structure NPD processes, in addition to the test scheduling problem discussed in Chapter 3 to 4, the management is often faced with another important decision problem, i.e., how to plan the sequence of many interrelated activities (Krishnan and Ulrich, 2001; Karniel and Reich, 2009). Formal scheduling techniques, such as CPM and PERT, cannot deal with interrelated activities (Eppinger et al., 1994; Browning and Ramasesh, 2007). One effective tool for addressing this shortfall is DSM, which has spawned many research efforts on sequencing design activities with the objective of minimizing feedbacks. It is known that DSM sequencing problem is NP-complete and difficult to solve. Since many real world NPD projects often involve dozens/hundreds of interrelated activities, the development and improvement of solution procedures for large DSM-based activity sequencing problem is very important. As such, in this chapter a new approach is proposed for solving this problem.

The organization of this chapter is as follows. After a brief introduction, we introduce a 0-1 Quadratic Integer Program (QIP) for DSM sequencing problem in Section 5.2. Section 5.3 presents a decomposition based solution strategy for solving the problem. In Section 5.4, we perform a number of experiments, and apply the solution strategy to three real data sets. Conclusions are summarized in Section 5.5. Proofs of all the results in this chapter are provided in Appendix C.

## 5.1 Introduction

Many researchers have highlighted weaknesses of network-based approaches, such as PERT and CPM, for scheduling NPD projects. First, PERT/CPM cannot effectively model iteration, which is a fundamental characteristic of NPD processes. Iteration occurs in an NPD process for two main reasons: (1) the outputs of activities, such as engineering drawings, specifications and bill of materials, are often unstable and inaccurate, and need to be reworked when downstream activities detect some problems in the original design (Loch and Terwiesch, 2005; Love et al., 2009); (2) downstream activities may be repeated when modified information is passed along from upstream activities (Smith and Eppinger, 1997a; Jun et al., 2005; Lin et al., 2008). Thus, cyclic information flows are quite common in NPD processes and PERT/CPM could not deal with such loops effectively. Second, in PERT/CPM, the underlying precedence relationships among activities are assumed known and unchanged. However, for most complex development projects, clear precedence constraints do not exist and are rarely known in advance (Eppinger et al., 1994; Ahmadi et al., 2001; Jun and Suh, 2008).

To address these shortfalls, one known method is DSM. As illustrated in Figure 5.1, the basic form of DSM is a binary matrix representation of a project with elements denoting individual design activities which are executed in the temporal order listed from top to bottom (Eppinger et al., 1994; Chen et al., 2004). Sub-diagonal marks represent inputs from upstream activities to downstream, and super-diagonal marks denote feedbacks from downstream activities to upstream (Denker et al., 2001; Karniel and Reich, 2009). As such, DSM provides a compact representation of a complex system by showing information dependencies in a square matrix, as well as a useful tool for describing and investigating iteration (Cho and Eppinger, 2005).



(a) DSM Representation (b) NDSM Representation

**Figure 5.1 DSM/NDSM representation of iterative NPD process: an example**

The DSM approach was first introduced by Steward (1981). Eppinger et al. (1994) extended Steward's work by explicitly including numerical measures of the degree of activity dependence, and introduced NDSM, such that more complex analytical procedures could be used to further improve the design process. Since then, many practitioners and researchers have reported successful applications of DSM approach in managing NPD projects (see e.g. Eppinger, 2001; Sharman and Yassine, 2004; MacCormack et al., 2006; Veenstra et al., 2006; De Weerd-Nederhof et al., 2007; Amrit and van Hillegersberg, 2008; Sosa, 2008). Reviews of DSM approach can be found in Browning (2001), Karniel and Reich (2009).

In project scheduling, DSM has spawned many research efforts on sequencing design activities (Eppinger et al., 1994; Browning, 2001; Karniel and Reich, 2009). In most of previous studies, the objective of sequencing is to minimize feedbacks (Meier et al., 2007; Lancaster and Ozbayrak, 2008). There are several reasons why this is meaningful and important. First, feedbacks determine where iteration loops occur (Eppinger et al., 1994; McCulley and Bloebaum, 1996). Iteration is a major driver for lengthy and costly product development (Mihm et al., 2003; Lee and Suh, 2006; Love et al., 2008). Thus, minimizing feedbacks is a good approximation for concurrently reducing cycle time and development costs (Ahmadi et al., 2001; Collins et al., 2009). For instance, Ahmadi et al. (2001) showed that the average errors resulting from the surrogate objective function of minimizing feedbacks was only 2.6% and 2.7%, and the maximum error was less than 8%, over 540 problems solved. Second, as reported by McCulley and Bloebaum (1996), in practice, especially for distributed product development, "the design manager is likely to desire choosing an evaluation procedure which requires the least amount of information to be guessed initially. Each feedback corresponds to one or more guesses. Therefore, minimizing feedbacks will also minimize the amount of information required to be supplied as initial guesses".

Several independent researchers have reported that DSM sequencing problem is NP-complete (e.g. McCulley and Bloebaum, 1996; Ahmadi et al., 2001; Meier et al., 2007). Therefore, there is almost no chance to develop solution algorithms with a polynomial-time complexity (Li and Sun, 2006). To solve large-scale sequencing problem, one stream of research focused on meta-heuristic methods, such as Genetic Algorithm (Altus et al., 1996; Whitfield et al., 2003; Meier et al., 2007), Simulated Annealing (Abdelsalam and Bao, 2006, 2007), and Evolutionary Algorithm (Lancaster and Cheng, 2008).

Another stream of research focused on decomposition based methods. For instance, Rogers (1989) developed DeMAID which involved two steps: (1) grouping activities into blocks; and (2) sequencing the activities within each block so as to minimize feedbacks. The DeMAID was further extended to DeMAID/GA by incorporating GA for sequencing the activities within each block (McCulley and

Bloebaum, 1996; Rogers, 1996, 1999). Ahmadi et al. (2001) complemented this line of research by explicitly developing mathematical models for solving large-scale DSM sequencing problem. In their paper, a Block Decomposition method was presented for grouping activities into blocks, with the objective of minimizing feedbacks among blocks. A Branch-and-Bound algorithm was then used to identify optimal solutions of sub-problems, and finally, the sub-problem solutions were concatenated to a solution of the overall problem. Their work has clearly shed light on the analysis of DSM sequencing problem, and has greatly advanced our understanding of managing NPD processes. However, a major disadvantage of their Block Decomposition method is that the resulting overall activity sequence (obtained by solving all sub-problems) may be a sequence that is worse than the initial one.

In this study we therefore develop a new decomposition approach for solving large-scale DSM sequencing problem. Based on the approach proposed, the overall activity sequence can be gradually improved. Another contribution of this study lies in that it establishes two simple rules for direct feedback reduction, and presents a heuristic for improving feasible solutions through activity exchange. Finally, we perform a number of random examples, and apply the proposed methodology to three real data sets. Solutions presented in previous studies are used to benchmark the effectiveness of the proposed methodology.

## 5.2 Mathematical Model

Consider a product development project consisting of *n* design activities, where each individual design activity can be viewed as information-processing units that receives information from upstream activities and transform it into new information to be passed on to downstream activities (Clark and Fujimoto, 1991; Browning, 2009).

We assume that the degree to with each activity depends on the output information of other activities is known. Let parameter $d_{i,j}$ ($d_{i,j} \geq 0$) denote the dependence level of activity $i$ on activity $j$. Note that the model and methodologies developed in this study can be applied to either a binary DSM or a numerical DSM. In binary DSM, $d_{i,j}$ only takes binary values, while in numerical DSM, parameter $d_{i,j}$ can take real numbers. In practice, the following three measures, which are proposed by Eppinger et al. (1994) and extended by Browning and Eppinger (2002), Yassine (2007) and so on, can be used to estimate the value of $d_{i,j}$:

(1) Variability: if the output from activity *j* will change significantly (slightly), then the dependency tends to be strong (weak).

(2) Predictability: if the change of output from activity *j* is unpredictable (predictable), then the dependency tends to be strong (weak).

(3) Sensitivity: If a slight (substantial) change of output of activity *j* will lead to a large (small) magnitude of design iteration in activity *i*, then the dependency tends to be strong (weak).

Based on above measures, activity dependencies can then be quantified through interviewing the engineers and managers familiar with the projects and/or via surveying relevant documentation (Eppinger, 2001; Chen et al., 2004). Activity dependencies can also be effectively quantified by applying the well-known techniques, such as AHP (see Chen and Li, 2003), and QFD (see Chen et al., 2004).

To formulate the DSM sequencing problem, we introduce following notation:

$i, j =$ index for activities, $i, j = 1, 2, \ldots, n$

$m, h =$ index for activity positions, $m, h = 1, 2, \ldots, n$

**Decision Variables** (for $i = 1, 2, \ldots, n$; $m = 1, 2, \ldots, n$):

$$x_{im} = \begin{cases} 1 & \text{if activity } i \text{ is assigned to the } m^{\text{th}} \text{ position,} \\ 0 & \text{Otherwise.} \end{cases}$$

The objective of DSM sequencing problem is to find an activity sequence so as to minimize feedbacks, which can be formulated as following 0-1 Quadratic Integer Program (QIP):

$$Min\ f(x) = \sum_{i=1}^{n} \sum_{\substack{j=1 \\ j \neq i}}^{n} \sum_{m=1}^{n} \left( x_{im} \times \sum_{h=m+1}^{n} x_{jh} \right) d_{i,j} \tag{5.1}$$

Subject to:

$$\sum_{i=1}^{n} x_{im} = 1 \quad \forall m \tag{5.2}$$

$$\sum_{m=1}^{n} x_{im} = 1 \quad \forall i \tag{5.3}$$

$$x_{im} \in \{0, 1\} \quad \forall i, m \tag{5.4}$$

The objective (5.1) minimizes feedbacks or the sum of super-diagonal numbers in a NDSM. For instance, in the NDSM shown in 5.1(b), the total feedbacks for the activity sequence of {1, 2, 3, 4} are 1.8. The feedback minimization objective works as follows: if activity *j* is assigned to precede activity *i*, then we get $\sum_{h=m+1}^{n} x_{jh} = 0$, in other words, $d_{i,j}$ is not included in the objective (5.1); on the other hand, if activity *i* is assigned to precede activity *j*, then $\sum_{h=m+1}^{n} x_{jh} = 1$, in other words, $d_{i,j}$ is a feedback and is included in (5.1). Constraints (5.2) and (5.3) are assignment constraints. As mentioned earlier, DSM sequencing problem is NP-complete. Therefore, it's hard to solve the optimization problem (5.1)-(5.4).

## 5.3 Proposed Solution Strategy

In this section, we will describe how to solve the problem in (5.1)-(5.4). We first develop a heuristic for improving feasible solutions, and show that optimal solutions of small-scale problems can be obtained by a Branch-and-Bound method. This Branch-and-Bound method is then extended by a heuristic decomposition approach for solving large-scale problems. More specifically, the heuristic decomposition approach iteratively (1) selects a sub-problem from a feasible solution of the main problem, (2) employs the Branch-and-Bound method to find the optimal solution of the sub-problem, and (3) reincorporates the sub-problem solution into the solution of the main problem.

### 5.3.1 A Heuristic for Improving Feasible Solutions

In this subsection, we first develop two simple rules (Theorem 5.1 and 5.2) for direct feedback reduction. Based on the results, a heuristic is then presented for improving feasible solutions through activity exchange.

**THEOREM 5.1** *For an activity sequence of* $\{1,2,...,i-1, i, i+1,...,n\}$*, if* $d_{i,i+1} > d_{i+1,i}$*, then through exchanging activity* $i$ *and activity* $i+1$*, the total feedbacks can be reduced by* $\left(d_{i,i+1} - d_{i+1,i}\right)$.

The proof of all the results in this chapter can be found in Appendix C.

Theorem 5.1 shows that for two adjacent activities $i$ and $i+1$, if the dependency level of activity $i$ on activity $i+1$ is larger than that of activity $i+1$ on activity $i$, i.e., $d_{i,i+1} > d_{i+1,i}$, then the execution sequence of activity $i$ and activity $i+1$ should be exchanged. Let us use a real NDSM published in Chen and Li (2003) to further illustrate this. Figure 5.2 shows the original NDSM of nine activities for a chemical processing system, which is built based on the well-known technique of AHP. The total feedbacks for the original activity sequence of {1,4,5,8,10,11,17,18,19} are 3.837. Clearly, the dependency level of activity 1 on activity 4 is larger than that of activity 4 on activity 1. Thus, by exchanging the positions of activity 1 and activity 4, the total feedbacks can be reduced to 3.678. The result is shown in Figure 5.3.

| | | 1 | 4 | 5 | 8 | 10 | 11 | 17 | 18 | 19 |
|---|---|---|---|---|---|---|---|---|---|---|
| Operating Structure Design | 1 | ■ | 0.654 | | 0.232 | | 0.140 | | | 0.253 |
| Shipping Design | 4 | 0.495 | ■ | 0.451 | 0.310 | | | 0.165 | | |
| Structure Lifting Design | 5 | 0.286 | | ■ | | 0.161 | 0.136 | 0.224 | | |
| Structural Documentation | 8 | | 0.137 | 0.351 | ■ | | 0.089 | | 0.118 | |
| Wind Load Design | 10 | 0.117 | | | 0.194 | ■ | | | | 0.119 |
| Seismic Design | 11 | | | 0.200 | | | ■ | | 0.300 | |
| Foundation Load Design | 17 | | 0.160 | | | | | ■ | | 0.485 |
| Insulation Structural Design | 18 | 0.177 | | | 0.239 | | 0.071 | | ■ | |
| Structural Bill of Materials | 19 | | | 0.128 | | 0.795 | | | 0.193 | ■ |

**Figure 5.2 Original NDSM for a chemical processing system (From Chen and Li, 2003)**

| | | 4 | 1 | 5 | 8 | 10 | 11 | 17 | 18 | 19 |
|---|---|---|---|---|---|---|---|---|---|---|
| Shipping Design | 4 | ■ | 0.495 | 0.451 | 0.310 | | | 0.165 | | |
| Operating Structure Design | 1 | 0.654 | ■ | | 0.232 | | 0.140 | | | 0.253 |
| Structure Lifting Design | 5 | | 0.286 | ■ | | 0.161 | 0.136 | 0.224 | | |
| Structural Documentation | 8 | 0.137 | | 0.351 | ■ | | 0.089 | | 0.118 | |
| Wind Load Design | 10 | | 0.117 | | 0.194 | ■ | | | | 0.119 |
| Seismic Design | 11 | | | 0.200 | | | ■ | | 0.300 | |
| Foundation Load Design | 17 | 0.160 | | | | | | ■ | | 0.485 |
| Insulation Structural Design | 18 | | 0.177 | | 0.239 | | 0.071 | | ■ | |
| Structural Bill of Materials | 19 | | | 0.128 | | 0.795 | | | 0.193 | ■ |

**Figure 5.3 Improved NDSM through exchanging activities 1 and 4**

Our next result further establishes the rule for exchanging two arbitrary activities. Given an activity sequence of $\{1,2,...,j-1,j,j+1,...,i-1,i,i+1,...,n\}$, defining:

$$\theta_{j,i} = \sum_{k=j}^{i-1} d_{k,i} + \sum_{k=j+1}^{i-1} d_{j,k} - \sum_{k=j+1}^{i} d_{k,j} - \sum_{k=j+1}^{i-1} d_{i,k} \tag{5.5}$$

**THEOREM 5.2** *If $\theta_{j,i} > 0$, then through exchanging activity j and activity i, the total feedbacks can be reduced by $\theta_{j,i}$.*

Theorem 5.2 describes that for an initial activity sequence of $\{1,2,...,j-1,j,j+1,...,i-1,i,i+1,...,n\}$, if $\theta_{j,i} > 0$, then the execution sequence of activity $j$ and activity $i$ should be exchanged. For example, for the NDSM shown in Figure 5.3, inspection of activity 4 and activity 8 shows that $\theta_{4,8} = 0.495 + 0.451 + 0.310 + 0.232 - 0.654 - 0.137 - 0.351 = 0.346$. Hence, through exchanging activities 4 and 8, the feedbacks can be reduced to 3.332. The result is shown in Figure 5.4.

| | | 8 | 1 | 5 | 4 | 10 | 11 | 17 | 18 | 19 |
|---|---|---|---|---|---|---|---|---|---|---|
| Structural Documentation | 8 | ■ | | 0.351 | 0.137 | | 0.089 | | 0.118 | |
| Operating Structure Design | 1 | 0.232 | ■ | | 0.654 | | 0.140 | | | 0.253 |
| Structure Lifting Design | 5 | | 0.286 | ■ | | 0.161 | 0.136 | 0.224 | | |
| Shipping Design | 4 | 0.310 | 0.495 | 0.451 | ■ | | | 0.165 | | |
| Wind Load Design | 10 | 0.194 | 0.117 | | | ■ | | | | 0.119 |
| Seismic Design | 11 | | | 0.200 | | | ■ | | 0.300 | |
| Foundation Load Design | 17 | | | | 0.160 | | | ■ | | 0.485 |
| Insulation Structural Design | 18 | 0.239 | 0.177 | | | | 0.071 | | ■ | |
| Structural Bill of Materials | 19 | | | 0.128 | | 0.795 | | | 0.193 | ■ |

**Figure 5.4 Improved NDSM through exchanging activities 4 and 8**

Based on the results of Theorems 5.1 and 5.2, we now introduce following Procedure 5.1 for improving feasible solutions of DSM sequencing problem:

**Procedure 5.1**

*Step 1*. Calculate the total feedbacks $f$ for an initial activity sequence, and store it as the best sequence.

*Step 2*. Set mobile-activity $j$ to activity 1.

*Step 3*. Calculate $\theta_{j,i}$ for $i = 1,2,...,n$, and $i \neq j$. Identify activity $k$ that gives the maximum $\theta_{j,k}$. If $\theta_{j,k} > 0$, exchange activity *j* and activity *k*, save the result as the new best sequence and set $f = f - \theta_{j,k}$.

*Step 4*. Set the next activity as the new mobile-activity, and go back to *Step 3* until all the activities have been set as the mobile-activity.

### 5.3.2 The Branch-and-Bound Method

In this subsection, a Lagrangian relaxation based Branch-and-Bound method is presented to solve small DSM Sequencing problem instances. The main features of the Branch-and-Bound method are based on those proposed by Ahmadi et al. (2001). We first show that the 0-1 QIP in (5.1)-(5.4) can be transferred into following equivalent 0-1 Linear Integer Program (LIP):

$$Min\ f(x,z) = \sum_{i=1}^{n} \sum_{\substack{j=1 \\ j \neq i}}^{n} d_{i,j} z_{i,j} \tag{5.6}$$

Subject to: Equations (5.2)-(5.4),

$$z_{i,j} \geq x_{im} - \sum_{h=1}^{m-1} x_{jh}\,, \quad \forall m \tag{5.7}$$

$$z_{i,j} \in \{0,1\} \qquad \forall i,j \tag{5.8}$$

**THEOREM 5.3** *The optimal objective value of the 0-1 QIP is equal to that of the 0-1 LIP.*

Based on the result in Theorem 5.3, the 0-1 QIP in (5.1)-(5.4) is transferred to the 0-1 LIP. By using non-negative Lagrangian multipliers $\lambda_{i,j}^{m}$ to relax constraints (5.7), the following Lagrangian relaxation problem is obtained:

$$d(\lambda) = \min \sum_{i=1}^{n} \sum_{\substack{j=1 \\ j \neq i}}^{n} \left[ d_{i,j} z_{i,j} + \sum_{m=1}^{n} \lambda_{i,j}^{m} \left( x_{im} - \sum_{h=1}^{m-1} x_{jh} - z_{i,j} \right) \right] \tag{5.9}$$

Subject to: Equations (5.2)-(5.4), (5.8).

It has been shown in Ahmadi et al. (2001) that the Lagrangian relaxation of DSM sequencing problem can be reduced to an assignment problem. Their result is adapted and presented in the following Theorem 5.4.

**THEOREM 5.4** *For given non-negative Lagrangian multipliers* $\lambda_{i,j}^{m}$ *that satisfy:* $\sum_{m=1}^{N}\lambda_{i,j}^{m} \le d_{i,j}$, *the Lagrangian relaxation problem can be reduced to the following assignment problem:*

$$d(\lambda) = \min \sum_{i=1}^{n}\sum_{m=1}^{n}\left(\sum_{\substack{j=1\\ j\neq i}}^{n}\lambda_{i,j}^{m} - \sum_{\substack{j=1\\ j\neq i}}^{n}\sum_{h=m+1}^{n}\lambda_{j,i}^{h}\right)x_{im} \tag{5.10}$$

Subject to: Equations (5.2)-(5.4).

It is well-known that assignment problem can be effectively solved by the Hungarian method (Burkard et al., 2009). Therefore, for given non-negative Lagrangian multipliers $\lambda_{i,j}^{m}$ that satisfy $\sum_{m=1}^{N}\lambda_{i,j}^{m} \le d_{i,j}$, the Hungarian method can be used to solve the problem and obtain a lower bound for the 0-1 LIP. The best lower bound can be derived by choosing Lagrangian multipliers $\lambda_{i,j}^{m}$ to be the solution of the following Lagrangian dual problem:

$$\max\ d(\lambda) \tag{5.11}$$

Subject to:

$$\sum_{m=1}^{n}\lambda_{i,j}^{m} \le d_{i,j}, \quad \text{for all non-zero } d_{i,j} \tag{5.12}$$

$$\lambda_{i,j}^{m} \ge 0, \quad \text{for all } m \text{ and non-zero } d_{i,j} \tag{5.13}$$

The Lagrangian dual problem can then be solved by the well-known subgradient method. Based on the lower bounds identified by the Lagrangian relaxation and subgradient method, the following Branch-and-Bound procedures can then be used to find optimal solutions of DSM sequencing problem (Ahmadi et al., 2001).

**Procedure 5.2**

*Step 1.* Compute a feasible solution by Procedure 5.1, save the result as $x_{opt}$, and save the corresponding objective value as $f_{opt}$. Compute the Lagrangian bound $d(\lambda^*)$. If $d(\lambda^*) = f_{opt}$, stop.

*Step 2.* If all the variables have been fixed, then stop and $x_{opt}$ is the optimal solution. Otherwise, generate two new nodes by setting an unfixed variable to 0 or 1, and choose one of the two nodes to be explored first.

*Step 3.* Compute the Lagrangian bound $d(\lambda^*)$ for the current node. If $d(\lambda^*) \geq f_{opt}$, then the current node is fathomed, go to *Step 2*; Else, go to *Step 4*.

*Step 4.* Let $x^*$ be the respective optimal solution to the Lagrangian relaxation problem corresponding to the optimal Lagrangian multipliers. If $f(x^*) < f_{opt}$, set $x_{opt} = x^*$ and $f_{opt} = f(x^*)$. Go to *Step 2*.

## 5.3.3 The Heuristic Decomposition Approach

We now explain our heuristic decomposition approach in detail. First, given a feasible solution of the main problem, select a block of consecutive activities from position $M_1$ to position $M_2$ ($M_2 > M_1$) such that a smaller sub-problem is obtained. Second, solve this sub-problem by the Branch-and-Bound method. Third, reincorporate the sub-problem solution into the solution of the main problem. After an iteration of these three steps, we can get an improved solution of the main problem.

We use this improved solution to restart and iteratively change the value of $M_1$ and $M_2$ such that the new sub-problem focuses on a later part of the main sequence and also has an overlap with the subsequence of the previous iteration. The process is repeated until $M_2$ reaches the last activity $n$. These deliberations give rise to the following Procedure 5.3 for solving large-scale DSM sequencing problem. Denote:

| | |
|---|---|
| $N$ | Number of activities contained in a sub-problem |
| $L$ | The iteration step length, where $L$ is a positive integer and $L < N$ |
| $S$ | Activity set that store the sequence of activities from position 1 to position $n$ |
| $S_1$ | Activity set that store the sequence of activities from position 1 to position $M_1$ |
| $S_2$ | Activity set that store the sequence of activities from position $M_2$ to position $n$ |
| $S_b$ | Activity set that store the sequence of activities from position $M_1$ to position $M_2$ |
| $S_b^*$ | Activity set that stores the optimal activity sequence of the sub-problem |

**Procedure 5.3**

*Step 0*. Compute a feasible solution by Procedure 5.1, save the result as $S$. Initialize $N$ and $L$. Set $M_1 = 1$, $M_2 = N$.

While $M_2 < n$, do:

*Step 1*. Based on $S$, construct $S_1$, $S_b$ and $S_2$.

*Step 2*. Compute the optimal solution for $S_b$ by Procedure 5.2, save the result as $S_b^*$.

*Step 3*. Combine $S_1$, $S_b^*$ and $S_2$, save the result as $S$.

*Step 4*. $M_1 = M_1 + L, M_2 = M_2 + L$.

In the above procedure, parameter $N$ defines the size of the sub-problem. Theoretically, the optimal value of $N$ is $n$ (i.e. the number of activities in the main problem), such that the main problem can be solved to optimality. If the value of $N$

is too small, then solving the sub-problem by the Branch-and-Bound method may not improve the solution of the main problem. However, the activity sequencing problem is NP-complete, and thus if $N$ is too large, it will be computationally infeasible for the Branch-and-Bound method, because the calculation time increases drastically with the problem size. Therefore, in choosing the value of $N$, we need to avoid the extreme cases of setting $N$ too large or too small. Parameter L defines the iteration step length. We select $L$ such that the new sub-problem has an overlap with the subsequence of the previous iteration. To get an overlap, we set $L < N$.

## 5.4 Computational Experiments

As mentioned earlier, a major disadvantage of previous decomposition-based method in Ahmadi et al. (2001) is that the resulting overall activity sequence may be a sequence that is worse than the initial one, whereas the proposed solution strategy guarantees that the final activity sequence is better than the initial one. To fully understand how the proposed solution strategy is significantly different and better than previous method, in this section, we first perform a number of experiments to examine how previous method could lead to a sequence that is worse than the original one. After that, the proposed solution strategy is applied to three real-world NDSMs: (1) a NDSM for a chemical processing system (Chen and Li, 2003); (2) a NDSM for the turbopump concept design (Ahmadi et al., 2001), and (3) a NDSM for Power Line Communication (PLC) (Luh et al., 2009). Solutions presented in previous studies are used to benchmark the effectiveness of the proposed solution strategy. All the experiments are performed on a Pentium Dual-Core 2.1GHz Compaq laptop with 2G memory. Our objective in this section is to provide some insight, but not conclusive evidence, into the efficiency of our approach.

## 5.4.1 Test Examples

Previous decomposition-based methods in Ahmadi et al. (2001) involve two steps: (1) decomposing the interrelated activities into smaller blocks (i.e. sub-problems) with the objective of minimizing the sum of super-diagonal numbers among blocks, and (2) within each block, sequencing the interrelated activities so as to minimize the sum of super-diagonal numbers. We code this procedure in Matlab and perform four groups of experiments. Note that the decomposition problem (i.e. the first step), which is equivalent to the problem of partitioning N nodes into k disjoint subsets in a graph of G=(N, E)(where N is the number of nodes, and E is a set of weighted edges), is known to be NP-hard (Yu et al., 2007).

To perform more experiments and to be able to report our computational findings within reasonable time, in each group of experiments, the number of activities is set to 8, and the maximum number of activities allowed in a block is set to 4. We believe that the better the initial solution, the higher the probability that the procedure in Ahmadi et al. (2001) leads to a sequence that is worse than the original sequence. Consequently, we perform four groups of experiments with different initial solutions. In the first group of experiments, the degree of information dependency among activities are uniformly distributed on the interval (0,1). In the second group of experiments, the degree of information dependency among activities in the sub-diagonal region and those in the super-diagonal region of a DSM, are uniformly distributed on the interval (0.1,1) and (0,0.9), respectively. In the third group of experiments, the degree of information dependency among activities in the sub-diagonal region and those in the super-diagonal region of a DSM, are uniformly distributed on the interval (0.2,1) and (0,0.8), respectively. In the fourth group of experiments, the degree of information dependency among activities in the sub-diagonal region and those in the super-diagonal region of a DSM, are uniformly distributed on the interval (0.3,1) and (0,0.7), respectively.

We generate 1,000 random instances for each group. The computational results are shown in Table 5.1. In the first group of experiments, we observe 6 instances where the procedure in Ahmadi et al. (2001) leads to a sequence that is worse than the original sequence. As the initial solutions get better, the number of observations increases dramatically. For instance, in the fourth group of experiments, the number of observations increases to 420.

**Table 5.1 Computation results of test examples**

| **Groups** | **Dependency levels** | **Occurrence Rate**[1] |
|---|---|---|
| 1 | The dependency levels are uniformly distributed on (0,1) | 0.6% |
| 2 | The dependency levels in the sub-diagonal region and those in the super-diagonal region of a DSM, are uniformly distributed on (0.1,1) and (0,0.9), respectively | 3.9% |
| 3 | The dependency levels in the sub-diagonal region and those in the super-diagonal region of a DSM, are uniformly distributed on (0.2,1) and (0,0.8), respectively | 25.3% |
| 4 | The dependency levels in the sub-diagonal region and those in the super-diagonal region of a DSM, are uniformly distributed on (0.3,1) and (0,0.7), respectively | 42.0% |

[1]Occurrence rate is calculated as the number of observations where the procedure in Ahmadi et al. (2001) leads to a sequence that is worse than the original sequence divided by 1,000.

In our approach, we do not perform the first step of the procedure in Ahmadi et al. (2001), i.e. decomposing the interrelated activities into smaller blocks (i.e. sub-problems) with the objective of minimizing the sum of super-diagonal numbers among blocks. Instead, we use activity exchange (Procedure 5.1) to directly reduce the sum of super-diagonal numbers in a DSM. Based on the improved activity sequence identified by Procedure 5.1, Procedure 5.3 is then used to further improve the sequence. In other words, the proposed approach guarantees that the final activity sequence is better than the initial one. In practice, the initial solutions are provided by experienced project managers, and thus the initial activity sequence is often better than the randomly generated activity sequence. In such cases, using the procedure in Ahmadi et al. (2001) is inappropriate because the probability that the final activity sequence is worse than the original one could be quite high. The proposed approach can be used for scheduling these projects.

Moreover, the complexity of the first step in Ahmadi et al. (2001) is NP-hard, while the complexity of our Procedure 5.1 is $O(n^2)$. Thus, the procedure in Ahmadi et al. (2001) may not be applicable for large problem instances because the calculation time increases quickly with the number of activities. For instance, we try a random example of decomposing 25 activities into two groups where the maximum number of activities allowed in a block is set to 13, and find that the computation time for the first step in the procedure in Ahmadi et al. (2001) is more than 72 hours. For the same number of activities, the computation time for our Procedure 5.1 is several seconds.

### 5.4.2 Case Studies

Because feedbacks cause rework, it is accepted that in most cases, finding an activity sequence with minimum feedbacks can lead to considerable savings in both development time and costs of complex projects, see e.g. Steward (1981), Kusiak and Park (1990), Kusiak and Wang (1993), Tang et al. (2000), Ahmadi et al. (2001), Chen and Li (2003), Luh et al. (2009), Tang et al. (2009) for successful cases. Therefore, quite a few studies have examined this classic problem. Among them, Ahmadi et al. (2001), Chen and Li (2003), and Luh et al. (2009) proposed approaches for solving this problem, and also presented case studies to illustrate their approaches. Hence, we apply our approach to these three data sets, and benchmark our solutions with those presented in Ahmadi et al. (2001), Chen and Li (2003), and Luh et al. (2009). A short description of each case and our application results are presented below.

#### 5.4.2.1 Case 1: The Chemical Processing System

| | 5 | 8 | 17 | 4 | 18 | 11 | 1 | 10 | 19 |
|---|---|---|---|---|---|---|---|---|---|
| Structure Lifting Design 5 | ■ | | 0.224 | | | 0.136 | 0.286 | 0.161 | |
| Structural Documentation 8 | 0.351 | ■ | | 0.137 | 0.118 | 0.089 | | | |
| Foundation Load Design 17 | | | ■ | 0.160 | | | | | 0.485 |
| Shipping Design 4 | 0.451 | 0.310 | 0.165 | ■ | | | 0.495 | | |
| Insulation Structural Design 18 | | 0.239 | | | ■ | 0.071 | 0.177 | | |
| Seismic Design 11 | 0.200 | | | | 0.300 | ■ | | | |
| Operating Structure Design 1 | | 0.232 | | 0.654 | | 0.140 | ■ | | 0.253 |
| Wind Load Design 10 | | 0.194 | | | | | 0.117 | ■ | 0.119 |
| Structural Bill of Materials 19 | 0.128 | | | | 0.193 | | | 0.795 | ■ |

**Figure 5.5 Improved feasible solution by applying Procedure 5.1**

| | 5 | 8 | 18 | 10 | 19 | 11 | 17 | 4 | 1 |
|---|---|---|---|---|---|---|---|---|---|
| Structure Lifting Design 5 | ■ | | | 0.161 | | 0.136 | 0.224 | | 0.286 |
| Structural Documentation 8 | 0.351 | ■ | 0.118 | | | 0.089 | | 0.137 | |
| Insulation Structural Design 18 | | 0.239 | ■ | | | 0.071 | | | 0.177 |
| Wind Load Design 10 | | 0.194 | | ■ | 0.119 | | | | 0.117 |
| Structural Bill of Materials 19 | 0.128 | | 0.193 | 0.795 | ■ | | | | |
| Seismic Design 11 | 0.200 | | 0.300 | | | ■ | | | |
| Foundation Load Design 17 | | | | | 0.485 | | ■ | 0.160 | |
| Shipping Design 4 | 0.451 | 0.310 | | | | | 0.165 | ■ | 0.495 |
| Operating Structure Design 1 | | 0.232 | | | 0.253 | 0.140 | | 0.654 | ■ |

**Figure 5.6 Optimal solution by the Branch-and-Bound method**

Figure 5.2 shows the original NDSM of nine activities for a chemical processing system published in Chen and Li (2003). The information dependencies among the activities in the NDSM are quantified through the well-known technique of AHP (Chen and Li, 2003). We use this data set to illustrate Procedure 5.1 and the Brach-and-Bound method. By applying Procedure 5.1, we get the improved feasible solution shown in Figure 5.5, where the activity sequence is $\{5,8,17,4,18,11,1,10,19\}$. Note that in this activity sequence, activities 8 and 17, as well as activities 4 and 18, can be carried out in parallel since there is no information exchange between the activities. Based on this improved activity sequence, the total feedbacks are reduced by more than 20%, from original 3.837 to 2.911.

The optimal solution identified by Procedure 5.2 is shown in Figure 5.6, where the optimal activity sequence is $\{5,8,18,10,19,11,17,4,1\}$. Here the computation time is 970.8 minutes. Clearly, from Figure 5.6, we can conclude that there is no information exchange between activities 18 and 10, and so these two activities can be performed in parallel. Similarly, activities 19 and 11 can also be performed in parallel. Based on this optimal activity sequence, the total feedbacks are minimized at 2.29, and thus the optimal activity sequence can be used to improve the chemical processing system.

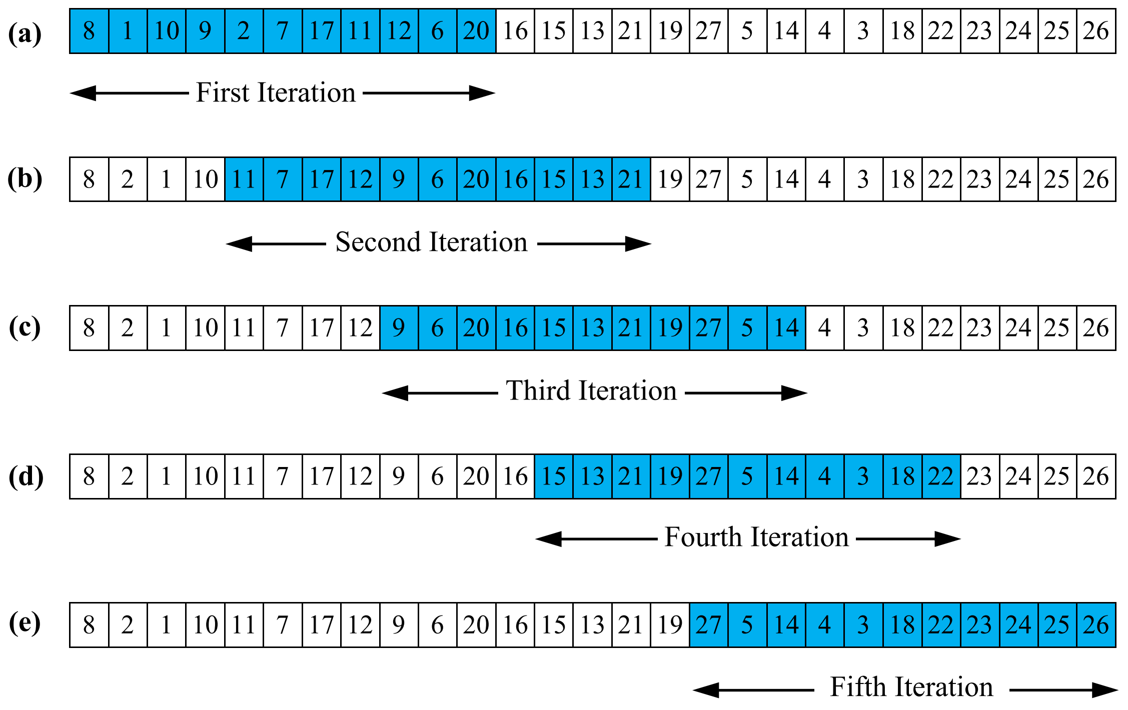
#### 5.4.2.2 Case 2: The Turbopump Concept Design

In this subsection, we apply the proposed heuristic decomposition approach to the data set published in Ahmadi et al. (2001). Figure 5.8 shows the initial NDSM for the concept design of the turbopump at Rocketdyne, which consists of 27 activities. The NDSM was constructed based on a questionnaire, and all the parameters were double checked by senior project managers (Ahmadi et al., 2001). Based on the proposed method, they get the final NDSM shown in Figure 5.9, where the total feedbacks are reduced to 5.35. As reported by Ahmadi et al. (2001), the new activity sequence in Figure 5.9, {1,10,9,2,7,8,17,11,12,6,16,20,21,19,15,13,4,3,27,14,18,22,5,23,24,25,26} had led to considerable savings in both development time and costs.

Note that in this data set, the dependency level $d_{i,j} = 1.0$ represents a hard dependency, in other words, activity *j* must be scheduled before activity *i* (Ahmadi et al., 2001). For example, Design Pump Housing (activity 4) must be scheduled before

CST Assess Pump Housing (activity 3), since the dependency level of activity 3 on activity 4, $d_{3,4} = 1.0$. To ensure this, we associate the hard dependencies with a large positive value. Figure 5.7(a) shows the improved initial solution by applying Procedure 5.1. As an example, we set the number of activities contained in each sub-problem $N = 11$, and the step length at each iteration $L = 4$. More specifically, in the first iteration, (1) activities from position 1 to position 11 form the sub-problem $S_b$, and activities from position 12 to position 27 constitute $S_2$; (2) the Branch-and-Bound method is then used to solve the sub-problem and identify the optimal activity sequence $S_b^*$, while the activity sequence in $S_2$ remains unchanged; (3) integrating $S_b^*$ and $S_2$ results in the improved activity sequence shown in Figure 5.7(b).

Based on the improved activity sequence, in the second iteration, (1) activities from position 1 to position 4 constitute $S_1$, activities from position 5 to position 15 form the sub-problem $S_b$, activities from position 16 to position 27 form $S_2$; (2) again, the Branch-and-Bound method is used to solve the sub-problem $S_b$ and identify the optimal activity sequence $S_b^*$, while the activity sequence in $S_1$ and $S_2$ remain unchanged; (3) combining $S_1$, $S_b^*$ and $S_2$ results in the improved activity sequence shown in Figure 5.7(c). Repeating these three steps in the third and fourth iteration, we get the improved activity sequence shown in Figure 5.7(d) and Figure 5.7(e), respectively. Finally, in the fifth iteration, activities from position 17 to position 27 constitute the sub-problem, and the final result is shown in Figure 5.10, where the total feedbacks are reduced to 5. The total computation time for the proposed approach is 770.4 minutes. Since the activity sequence identified by our approach, {8,2,1,10,11,7,17,12,9,6,20,16,15,13,21,19,27,5,14,4,3,18,22,23,24,25,26}, has smaller total feedbacks, it can be utilized to further improve the turbopump concept design. Note that here activities 11 and 7, activities 12, 9, 6, 20 and 16, activities 13 and 21, activities 5, 14 and 4, activities 3 and 18, as well as activities 22 and 23, can be performed in parallel since there is no information exchange among them.


(a) 8 1 10 9 2 7 17 11 12 6 20 16 15 13 21 19 27 5 14 4 3 18 22 23 24 25 26
First Iteration
(b) 8 2 1 10 11 7 17 12 9 6 20 16 15 13 21 19 27 5 14 4 3 18 22 23 24 25 26
Second Iteration
(c) 8 2 1 10 11 7 17 12 9 6 20 16 15 13 21 19 27 5 14 4 3 18 22 23 24 25 26
Third Iteration
(d) 8 2 1 10 11 7 17 12 9 6 20 16 15 13 21 19 27 5 14 4 3 18 22 23 24 25 26
Fourth Iteration
(e) 8 2 1 10 11 7 17 12 9 6 20 16 15 13 21 19 27 5 14 4 3 18 22 23 24 25 26
Fifth Iteration

(a) Initial solution by applying Procedure 5.1; (b) Improved solution after the first iteration;
(c) Improved solution after the second iteration; (d) Improved solution after the third iteration;
(e) Improved solution after the fourth iteration

**Figure 5.7 The decomposition strategy for the turbopump concept design**

#### 5.4.2.3 Case 3: The Power Line Communication

In this section, we apply the heuristic decomposition approach to the data set published in Luh et al. (2009). Figure 5.11 shows the initial NDSM for the development of Power Line Communication (PLC), which consists of 22 activities. The NDSM was constructed based on a survey and detailed interview with different departments managers (Luh et al., 2009). Here the positions of activity 1 and activity 2 can be fixed in the top of the NDSM since these two activities do not need information input from all other activities. By applying the proposed heuristic, they get the final result shown in Figure 5.12, where the total feedbacks are reduced from original 16.4 to 12.6. As reported by Luh et al. (2009), because iteration is a major driver for lengthy and costly product development of Power Line Communication, the new activity sequence with smaller total feedbacks in Figure 5.12, i.e., {1,2,6,8,7,3,5,4,10,9,16,11,12,13,14,15,22,19,17,18,20,21}, had considerably reduced the development time and costs.

In this example, we set the number of activities in each sub-problem $N = 8$, and the step length at each iteration $L = 4$. Applying our approach results in the final NDSM shown in Figure 5.13, which suggests that the activities should executed in the sequence of {1,2,6,8,7,3,4,10,16,9,5,11,12,13,14,15,22,19,17,18,20,21}. The total computation time is 1586.2 minutes. From Figure 5.13, we can conclude that activities 9 and 5, activities 13 and 14, as well as activities 15 and 22, can be executed in parallel because there is no information exchange between the activities. Based on the activity sequence identified by our approach, the total feedbacks are reduced to 12.5, and so it can be used to further improve the development process of Power Line Communication.

| | | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SSP Engine Balance | 1 | ■ | 0.15 | | | | | 0.1 | | | 0.1 | | | | | | | | | | | | | | | | | |
| CMT Make Preliminary Material Selection | 2 | | ■ | 0.1 | | | 0.1 | | 0.1 | 0.1 | | | 0.1 | | | | | | | | | | | | | | | |
| CST Assess Pump Housing | 3 | | | ■ | 1.0 | | | | | | | | | | | | | | | | | | | | | | | |
| Design Pump Housing | 4 | | 0.5 | 0.2 | ■ | | | | | | 1.0 | | | 1.0 | | | | | 0.1 | 1.0 | | | 0.1 | | | | | |
| CST Assess Turbine Housing | 5 | | | | | ■ | | | | | | | | | | | | | | | | | | | | | | 1.0 |
| CST Compare Design Annulus Area | 6 | | | | | | ■ | 1.0 | | | | | | | | | | | | | | | | | | | | |
| CAX Determine Optimum Turbine Staging | 7 | 1.0 | 0.1 | | | | 0.1 | ■ | 0.1 | | 1.0 | | | | | | | 0.2 | | | | | | | | | | 0.1 |
| CST Compare Design Pitchline Velocities | 8 | | | | | | | | ■ | | | | | | | | | | | | | | | | | | | |
| CST Compare Design Impeller Tip Speed | 9 | | | | | | | | | ■ | 1.0 | | | | | | | | | | | | | | | | | |
| CHX Determine Pumping Components | 10 | 1.0 | 0.1 | | | | | 0.2 | | 0.1 | ■ | 0.15 | | | | | | | | | | | | | | | | |
| CDE Design Pumping Elements | 11 | | 0.5 | | | | | | | | 1.0 | ■ | 0.3 | | 0.1 | | | | | | | | | | | | | |
| CST Evaluate Rotor Sizing | 12 | | | | | | | | | | | 1.0 | ■ | | | | | 1.0 | | | | | | | | | | |
| CDE Incorporate Bearing Dimensions | 13 | | | | | | | | | | | | | ■ | | 1.0 | | | | | | | | | | | | |
| CDE Design Rotor | 14 | | 0.2 | | | | | | | | | 1.0 | | 1.0 | ■ | | | 1.0 | 0.1 | 1.0 | | | 0.2 | | | 0.1 | | |
| CBR Determine Bearing Geometry | 15 | | | | 0.1 | | | | | | | | 1.0 | | 0.2 | ■ | 1.0 | | | | | | | | | 0.1 | | 0.1 |
| CDE Position Bearings and Selection | 16 | | 0.2 | | | | | 1.0 | | | 1.0 | | | 0.2 | | | ■ | | | | | | | | | | | |
| CDE Design Turbine | 17 | | 0.2 | | | | | 1.0 | | | | | 0.3 | | 0.1 | | | ■ | | | | | | | | | | |
| CDE Integrate Rotor and Structure Layout | 18 | | | | 1.0 | | | | | | | | | | 1.0 | | | | ■ | | | | | | | | 0.1 | 1.0 |
| CDE Incorporate Seal Dimensions | 19 | | | | | | | | | | | | | | | | | | | ■ | | 1.0 | | | | | | |
| CSL Define Seal System | 20 | | 0.2 | | 0.1 | | | 1.0 | | | 1.0 | | | | | | | | | 0.3 | ■ | | | | | | | |
| CSL Define Individual Sealing Elements | 21 | | | | | | | | | | | 0.1 | | | 0.2 | | | | | | 1.0 | ■ | 0.05 | | | | | 0.1 |
| CDE Develop Thrust Balance | 22 | | | | | | | | | | 0.15 | | | | | | | | 1.0 | | | | ■ | | | | | |
| CRD Build Finite Element Model | 23 | | 0.1 | | | 0.3 | | | | | | | | | 1.0 | | | | | | | | | ■ | | | | |
| CRD Define Linear Rotordynamic Behavior | 24 | | | | | | | 1.0 | | | 1.0 | | | | | 1.0 | | | | | | 1.0 | | 1.0 | ■ | | | |
| CRD Evaluate Design | 25 | | | | | | | | | | | | | | | | | | | | | | | | 1.0 | ■ | | |
| CDE Analyze Weight | 26 | | | | | | | | | | | | | | | | | | 1.0 | | | | | | | 0.2 | ■ | |
| Design Turbine Housing | 27 | | 0.5 | | | 0.1 | | 1.0 | | | | | | 1.0 | | | | | 0.2 | 1.0 | | | 0.1 | | | | | ■ |

**Figure 5.8 Original NDSM for turbopump concept design (from Ahmadi et al., 2001)**

| | | 1 | 10 | 9 | 2 | 7 | 8 | 17 | 11 | 12 | 6 | 16 | 20 | 21 | 19 | 15 | 13 | 4 | 3 | 27 | 14 | 18 | 22 | 5 | 23 | 24 | 25 | 26 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SSP Engine Balance | 1 | ■ | 0.1 | | 0.15 | 0.1 | | | | | | | | | | | | | | | | | | | | | | |
| CHX Determine Pumping Components | 10 | 1.0 | ■ | 0.1 | 0.1 | 0.2 | | | 0.15 | | | | | | | | | | | | | | | | | | | |
| CST Compare Design Impeller Tip Speed | 9 | | 1.0 | ■ | | | | | | | | | | | | | | | | | | | | | | | | |
| CMT Make Preliminary Material Selection | 2 | | | 0.1 | ■ | | 0.1 | | | 0.1 | 0.1 | | | | | | | | 0.1 | | | | | | | | | |
| CAX Determine Optimum Turbine Staging | 7 | 1.0 | 1.0 | | 0.1 | ■ | 0.1 | 0.2 | | | 0.1 | | | | | | | | | 0.1 | | | | | | | | |
| CST Compare Design Pitchline Velocities | 8 | | | | | | ■ | | | | | | | | | | | | | | | | | | | | | |
| CDE Design Turbine | 17 | | | | 0.2 | 1.0 | | ■ | | 0.3 | | | | | | | | | | | 0.1 | | | | | | | |
| CDE Design Pumping Elements | 11 | | 1.0 | | 0.5 | | | | ■ | 0.3 | | | | | | | | | | | 0.1 | | | | | | | |
| CST Evaluate Rotor Sizing | 12 | | | | | | | 1.0 | 1.0 | ■ | | | | | | | | | | | | | | | | | | |
| CST Compare Design Annulus Area | 6 | | | | | 1.0 | | | | | ■ | | | | | | | | | | | | | | | | | |
| CDE Position Bearings and Selection | 16 | | 1.0 | | 0.2 | 1.0 | | | | | | ■ | | | | | 0.2 | | | | | | | | | | | |
| CSL Define Seal System | 20 | | 1.0 | | 0.2 | 1.0 | | | | | | | ■ | | 0.3 | | | 0.1 | | | | | | | | | | |
| CSL Define Individual Sealing Elements | 21 | | | | | | | | 0.1 | | | | 1.0 | ■ | | | | | | 0.1 | 0.2 | | 0.05 | | | | | |
| CDE Incorporate Seal Dimensions | 19 | | | | | | | | | | | | | 1.0 | ■ | | | | | | | | | | | | | |
| CBR Determine Bearing Geometry | 15 | | | | | | | | | 1.0 | | 1.0 | | | | ■ | | 0.1 | | 0.1 | 0.2 | | | | | | 0.1 | |
| CDE Incorporate Bearing Dimensions | 13 | | | | | | | | | | | | | | | 1.0 | ■ | | | | | | | | | | | |
| Design Pump Housing | 4 | | 1.0 | | 0.5 | | | | | | | | | | 1.0 | | 1.0 | ■ | 0.2 | | | 0.1 | 0.1 | | | | | |
| CST Assess Pump Housing | 3 | | | | | | | | | | | | | | | | | 1.0 | ■ | | | | | | | | | |
| Design Turbine Housing | 27 | | | | 0.5 | 1.0 | | | | | | | | | 1.0 | | 1.0 | | | ■ | | 0.2 | 0.1 | 0.1 | | | | |
| CDE Design Rotor | 14 | | | | 0.2 | | | 1.0 | 1.0 | | | | | | 1.0 | | 1.0 | | | | ■ | 0.1 | 0.2 | | | | 0.1 | |
| CDE Integrate Rotor and Structure Layout | 18 | | | | | | | | | | | | | | | | | 1.0 | | 1.0 | 1.0 | ■ | | | | | | 0.1 |
| CDE Develop Thrust Balance | 22 | | 0.15 | | | | | | | | | | | | | | | | | | | 1.0 | ■ | | | | | |
| CST Assess Turbine Housing | 5 | | | | | | | | | | | | | | | | | | | 1.0 | | | | ■ | | | | |
| CRD Build Finite Element Model | 23 | | | | 0.1 | | | | | | | | | | | | | | | | 1.0 | | | 0.3 | ■ | | | |
| CRD Define Linear Rotordynamic Behavior | 24 | | 1.0 | | | 1.0 | | | | | | | | 1.0 | | 1.0 | | | | | | | | | 1.0 | ■ | | |
| CRD Evaluate Design | 25 | | | | | | | | | | | | | | | | | | | | | | | | | 1.0 | ■ | |
| CDE Analyze Weight | 26 | | | | | | | | | | | | | | | | | | | | | 1.0 | | | | | 0.2 | ■ |

**Figure 5.9 Final NDSM in Ahmadi et al. (2001) for turbopump concept design**

| | | 8 | 2 | 1 | 10 | 11 | 7 | 17 | 12 | 9 | 6 | 20 | 16 | 15 | 13 | 21 | 19 | 27 | 5 | 14 | 4 | 3 | 18 | 22 | 23 | 24 | 25 | 26 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CST Compare Design Pitchline Velocities | 8 | ■ | | | | | | | | | | | | | | | | | | | | | | | | | | |
| CMT Make Preliminary Material Selection | 2 | 0.1 | ■ | | | | | | 0.1 | 0.1 | 0.1 | | | | | | | | | | | 0.1 | | | | | | |
| SSP Engine Balance | 1 | | 0.15 | ■ | 0.1 | | 0.1 | | | | | | | | | | | | | | | | | | | | | |
| CHX Determine Pumping Components | 10 | | 0.1 | 1.0 | ■ | 0.15 | 0.2 | | | 0.1 | | | | | | | | | | | | | | | | | | |
| CDE Design Pumping Elements | 11 | | 0.5 | | 1.0 | ■ | | | 0.3 | | | | | | | | | | | 0.1 | | | | | | | | |
| CAX Determine Optimum Turbine Staging | 7 | 0.1 | 0.1 | 1.0 | 1.0 | | ■ | 0.2 | | | 0.1 | | | | | | | 0.1 | | | | | | | | | | |
| CDE Design Turbine | 17 | | 0.2 | | | | 1.0 | ■ | 0.3 | | | | | | | | | | | 0.1 | | | | | | | | |
| CST Evaluate Rotor Sizing | 12 | | | | | 1.0 | | 1.0 | ■ | | | | | | | | | | | | | | | | | | | |
| CST Compare Design Impeller Tip Speed | 9 | | | | 1.0 | | | | | ■ | | | | | | | | | | | | | | | | | | |
| CST Compare Design Annulus Area | 6 | | | | | | 1.0 | | | | ■ | | | | | | | | | | | | | | | | | |
| CSL Define Seal System | 20 | | 0.2 | | 1.0 | | 1.0 | | | | | ■ | | | | | 0.3 | | | | 0.1 | | | | | | | |
| CDE Position Bearings and Selection | 16 | | 0.2 | | 1.0 | | 1.0 | | | | | | ■ | | 0.2 | | | | | | | | | | | | | |
| CBR Determine Bearing Geometry | 15 | | | | | | | | 1.0 | | | | 1.0 | ■ | | | | 0.1 | | 0.2 | 0.1 | | | | | | 0.1 | |
| CDE Incorporate Bearing Dimensions | 13 | | | | | | | | | | | | | 1.0 | ■ | | | | | | | | | | | | | |
| CSL Define Individual Sealing Elements | 21 | | | | | 0.1 | | | | | | 1.0 | | | | ■ | | 0.1 | | 0.2 | | | | 0.05 | | | | |
| CDE Incorporate Seal Dimensions | 19 | | | | | | | | | | | | | | | 1.0 | ■ | | | | | | | | | | | |
| Design Turbine Housing | 27 | | 0.5 | | | | 1.0 | | | | | | | | 1.0 | | 1.0 | ■ | 0.1 | | | | 0.2 | 0.1 | | | | |
| CST Assess Turbine Housing | 5 | | | | | | | | | | | | | | | | | 1.0 | ■ | | | | | | | | | |
| CDE Design Rotor | 14 | | 0.2 | | | 1.0 | | 1.0 | | | | | | | 1.0 | | 1.0 | | | ■ | | | 0.1 | 0.2 | | | 0.1 | |
| Design Pump Housing | 4 | | 0.5 | | 1.0 | | | | | | | | | | 1.0 | | 1.0 | | | | ■ | 0.2 | 0.1 | 0.1 | | | | |
| CST Assess Pump Housing | 3 | | | | | | | | | | | | | | | | | | | | 1.0 | ■ | | | | | | |
| CDE Integrate Rotor and Structure Layout | 18 | | | | | | | | | | | | | | | | | 1.0 | | 1.0 | 1.0 | | ■ | | | | | 0.1 |
| CDE Develop Thrust Balance | 22 | | | | 0.15 | | | | | | | | | | | | | | | | | | 1.0 | ■ | | | | |
| CRD Build Finite Element Model | 23 | | 0.1 | | | | | | | | | | | | | | | | 0.3 | 1.0 | | | | | ■ | | | |
| CRD Define Linear Rotordynamic Behavior | 24 | | | | 1.0 | | 1.0 | | | | | | | 1.0 | | 1.0 | | | | | | | | | 1.0 | ■ | | |
| CRD Evaluate Design | 25 | | | | | | | | | | | | | | | | | | | | | | | | | 1.0 | ■ | |
| CDE Analyze Weight | 26 | | | | | | | | | | | | | | | | | | | | | | 1.0 | | | | 0.2 | ■ |

**Figure 5.10 Final NDSM for turbopump concept design by our approach**

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Product planning 1 | ■ | 1.0 | | | | | | | | | | | | | | | | | | | | |
| Market analysis 2 | 1.0 | ■ | | | | | | | | | | | | | | | | | | | | |
| Cost analysis 3 | 0.6 | | ■ | | 0.8 | | 0.5 | | | | | | | | | | | | | | | |
| Product spec. 4 | 0.4 | 0.5 | 0.8 | ■ | 0.8 | 0.3 | 0.5 | | | | | | | | | | | | | | | |
| Sales & profit 5 | 0.6 | 0.4 | 0.8 | 0.8 | ■ | | 0.2 | | | | | | | | | | | | | | | |
| Concept design 6 | 0.9 | 0.5 | | | | ■ | | 0.9 | | | | | | | | | | | | | | |
| Concept evaluation 7 | | | | | | 0.9 | ■ | 0.9 | | | | | | | | | | | | | | |
| Industrial design 8 | 0.9 | 0.9 | | | | 0.9 | 0.9 | ■ | | | | | | | | | | | | | | |
| Mechanical design 9 | | | | 0.5 | | | | | ■ | 0.9 | 0.9 | | | | | 0.9 | | | | | | |
| Components layout 10 | | | | 0.5 | | | | | 0.9 | ■ | | | | | | 0.8 | | | | | | |
| Electronic design 11 | | | | 0.5 | | | | | 0.9 | 0.8 | ■ | 0.9 | | | | 0.4 | | | | | | |
| IC design 12 | | | | | | | | | 0.4 | 0.4 | 0.9 | ■ | 0.9 | | | | | | | | | |
| Software design 13 | | | | | | | | | | | 0.6 | 0.9 | ■ | | | | | | | | | |
| Prototype making 14 | | | | | | | | 0.2 | 0.5 | 0.4 | 0.6 | | | ■ | 0.8 | | | | | | | |
| Mold design 15 | | | | | | 0.4 | 0.3 | 0.1 | 0.5 | 0.4 | | | | 0.8 | ■ | | | | | | | |
| Packing design 16 | | | | | | | | 0.4 | 0.8 | 0.9 | | | | | | ■ | | | | | | |
| Design pilot run 17 | | | | | | | | | 0.4 | | | | | | 0.7 | 0.4 | ■ | | 0.7 | | | 0.7 |
| Production pilot run 18 | | | | | | | | | | | | | | | 0.7 | 0.3 | 0.5 | ■ | 0.7 | | | 0.8 |
| Production tooling 19 | | | | | | | 0.2 | | 0.2 | | | | 0.2 | | 0.7 | 0.3 | | 0.7 | ■ | | | 0.8 |
| Customer authorize 20 | | | | 0.3 | | | | | | | | | | | | | 0.5 | 0.5 | | ■ | | 0.3 |
| Mass production 21 | | | | | | | | | | | | | | | | | | 0.4 | | 0.5 | ■ | |
| Inspection criteria 22 | | | | 0.2 | | | | | 0.3 | | | | 0.3 | | | | 0.7 | | 0.7 | | | ■ |

**Figure 5.11 Original NDSM for PLC design (from Luh et al., 2009)**

| | 1 | 2 | 6 | 8 | 7 | 3 | 5 | 4 | 10 | 9 | 16 | 11 | 12 | 13 | 14 | 15 | 22 | 19 | 17 | 18 | 20 | 21 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Product planning 1 | ■ | 1.0 | | | | | | | | | | | | | | | | | | | | |
| Market analysis 2 | 1.0 | ■ | | | | | | | | | | | | | | | | | | | | |
| Concept design 6 | 0.9 | 0.5 | ■ | 0.9 | | | | | | | | | | | | | | | | | | |
| Industrial design 8 | 0.9 | 0.9 | 0.9 | ■ | 0.9 | | | | | | | | | | | | | | | | | |
| Concept evaluation 7 | | | 0.9 | 0.9 | ■ | | | | | | | | | | | | | | | | | |
| Cost analysis 3 | 0.6 | | | | 0.5 | ■ | 0.8 | | | | | | | | | | | | | | | |
| Sales & profit 5 | 0.6 | 0.4 | | | 0.2 | 0.8 | ■ | 0.8 | | | | | | | | | | | | | | |
| Product spec. 4 | 0.4 | 0.5 | 0.3 | | 0.5 | 0.8 | 0.8 | ■ | | | | | | | | | | | | | | |
| Components layout 10 | | | | | | | | 0.5 | ■ | 0.9 | 0.8 | | | | | | | | | | | |
| Mechanical design 9 | | | | | | | | 0.5 | 0.9 | ■ | 0.9 | 0.9 | | | | | | | | | | |
| Packing design 16 | | | | 0.4 | | | | | 0.9 | 0.8 | ■ | | | | | | | | | | | |
| Electronic design 11 | | | | | | | | 0.5 | 0.8 | 0.9 | 0.4 | ■ | 0.9 | | | | | | | | | |
| IC design 12 | | | | | | | | | 0.4 | 0.4 | | 0.9 | ■ | 0.9 | | | | | | | | |
| Software design 13 | | | | | | | | | | | | 0.6 | 0.9 | ■ | | | | | | | | |
| Prototype making 14 | | | | 0.2 | | | | | 0.4 | 0.5 | | 0.6 | | | ■ | 0.8 | | | | | | |
| Mold design 15 | | | 0.4 | 0.1 | 0.3 | | | | 0.4 | 0.5 | | | | | 0.8 | ■ | | | | | | |
| Inspection criteria 22 | | | | | | | | 0.2 | | 0.3 | | | | 0.3 | | | ■ | 0.7 | 0.7 | | | |
| Production tooling 19 | | | | | 0.2 | | | | | 0.2 | 0.3 | | | 0.2 | | 0.7 | 0.8 | ■ | | 0.7 | | |
| Design pilot run 17 | | | | | | | | | | 0.4 | 0.4 | | | | | 0.7 | 0.7 | 0.7 | ■ | | | |
| Production pilot run 18 | | | | | | | | | | | 0.3 | | | | | 0.7 | 0.8 | 0.7 | 0.5 | ■ | | |
| Customer authorize 20 | | | | | | | | 0.3 | | | | | | | | | 0.3 | | 0.5 | 0.5 | ■ | |
| Mass production 21 | | | | | | | | | | | | | | | | | | | | 0.4 | 0.5 | ■ |

**Figure 5.12 Final NDSM for PLC design in Luh et al. (2009)**

| | | 1 | 2 | 6 | 8 | 7 | 3 | 4 | 10 | 16 | 9 | 5 | 11 | 12 | 13 | 14 | 15 | 22 | 19 | 17 | 18 | 20 | 21 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Product planning | 1 | ■ | 1.0 | | | | | | | | | | | | | | | | | | | | |
| Market analysis | 2 | 1.0 | ■ | | | | | | | | | | | | | | | | | | | | |
| Concept design | 6 | 0.9 | 0.5 | ■ | 0.9 | | | | | | | | | | | | | | | | | | |
| Industrial design | 8 | 0.9 | 0.9 | 0.9 | ■ | 0.9 | | | | | | | | | | | | | | | | | |
| Concept evaluation | 7 | | | 0.9 | 0.9 | ■ | | | | | | | | | | | | | | | | | |
| Cost analysis | 3 | 0.6 | | | | 0.5 | ■ | | | | | 0.8 | | | | | | | | | | | |
| Product spec. | 4 | 0.4 | 0.5 | 0.3 | | 0.5 | 0.8 | ■ | | | | 0.8 | | | | | | | | | | | |
| Components layout | 10 | | | | | | | 0.5 | ■ | 0.8 | 0.9 | | | | | | | | | | | | |
| Packing design | 16 | | | | 0.4 | | | | 0.9 | ■ | 0.8 | | | | | | | | | | | | |
| Mechanical design | 9 | | | | | | | 0.5 | 0.9 | 0.9 | ■ | | 0.9 | | | | | | | | | | |
| Sales & profit | 5 | 0.6 | 0.4 | | | 0.2 | 0.8 | 0.8 | | | | ■ | | | | | | | | | | | |
| Electronic design | 11 | | | | | | | 0.5 | 0.8 | 0.4 | 0.9 | | ■ | 0.9 | | | | | | | | | |
| IC design | 12 | | | | | | | | 0.4 | | 0.4 | | 0.9 | ■ | 0.9 | | | | | | | | |
| Software design | 13 | | | | | | | | | | | | 0.6 | 0.9 | ■ | | | | | | | | |
| Prototype making | 14 | | | | 0.2 | | | | 0.4 | | 0.5 | | 0.6 | | | ■ | 0.8 | | | | | | |
| Mold design | 15 | | | 0.4 | 0.1 | 0.3 | | | 0.4 | | 0.5 | | | | | 0.8 | ■ | | | | | | |
| Inspection criteria | 22 | | | | | | | 0.2 | | | 0.3 | | | | 0.3 | | | ■ | 0.7 | 0.7 | | | |
| Production tooling | 19 | | | | | 0.2 | | | | 0.3 | 0.2 | | | | 0.2 | | 0.7 | 0.8 | ■ | | 0.7 | | |
| Design pilot run | 17 | | | | | | | | | 0.4 | 0.4 | | | | | | 0.7 | 0.7 | 0.7 | ■ | | | |
| Production pilot run | 18 | | | | | | | | | 0.3 | | | | | | | 0.7 | 0.8 | 0.7 | 0.5 | ■ | | |
| Customer authorize | 20 | | | | | | | 0.3 | | | | | | | | | | 0.3 | | 0.5 | 0.5 | ■ | |
| Mass production | 21 | | | | | | | | | | | | | | | | | | | | 0.4 | 0.5 | ■ |

**Figure 5.13 Final NDSM for PLC design by our approach**

## 5.5 Conclusion

In this chapter, we have presented a new decomposition approach for large-scale DSM sequencing problem with the objective of minimizing feedbacks. The contribution of this work is threefold. First, we establish two simple rules (Theorem 5.1 and 5.2) for direct feedback reduction. Based on the results, a heuristic is then proposed for improving feasible solutions through activity exchange. Second, we show that the 0-1 QIP formulation of the DSM sequencing problem can be transferred to an equivalent 0-1 LIP problem, which can then be solved by a Branch-and-Bound method (Theorem 5.3 and 5.4). Third, we present a new decomposition approach to extend this Branch-and-Bound method for solving large-scale problems. The proposed decomposition approach has more flexibility than previous ones.

More importantly, based on the computational experiments in section 5.4.1, we show that in many cases, the probability that previous decomposition-based approach in Ahmadi et al. (2001) leads to a sequence that is worse than the original one is quite high. Thanks to the well construction of sub-problems, the proposed approach guarantees that the final activity sequence is better than the initial one. In practice, the initial solutions are provided by experienced project managers, and thus the initial activity sequence is often better than the randomly generated activity sequence, and thus using the approach in Ahmadi et al. (2001) is inappropriate. The proposed approach can be used for scheduling these projects.

We have also applied the proposed solution strategy to three real data sets, the conceptual design of the turbopump at Rocketdyne (Ahmadi et al., 2001), the development of a chemical processing system (Chen and Li, 2003), and the development of power line communication (Luh et al., 2009). It is shown that in all three cases, compared to the solutions presented in previous studies, applying our approach results in better solutions with smaller feedbacks.

# CHAPTER 6

# A NOVEL APPROACH TO LARGE-SCALE DSM SEQUENCING PROBLEM

## 6.1 Introduction

As was discussed in Chapter 5, DSM provides a compact representation of iterative NPD process by showing information dependencies in a square matrix. To accelerate NPD processes, DSM suggests to re-sequencing the activities such that iterative behaviors are minimized in the matrix. In recent years, there has been a growing interest in applying the DSM approach for planning product development projects (Karniel and Reich, 2009). However, a major difficulty lies in that DSM sequencing problem is NP-complete (McCulley and Bloebaum, 1996; Ahmadi et al., 2001; Meier et al., 2007).

To solve large-scale problem, one line of research focuses on meta-heuristic methods. Another line focuses on decomposition based approaches. More specifically, the overall project is first decomposed into smaller sub-problems which are easier to solve, and sub-problem solutions are then merged into a solution of the overall problem. Examples of such studies include McCulley and Bloebaum (1996), Rogers (1996, 1999), and Ahmadi et al. (2001). This chapter also follows previous decomposition based approaches. In particular, we extend the activity exchange rules (Theorem 5.1 and 5.2) to a group of activities, and find that through the fold operation, a block has similar properties to a single activity. Based on these findings, a novel decomposition approach is presented for solving large-scale DSM sequencing problem.

In the next section, a mathematical formulation is presented for the DSM sequencing problem. In Section 6.3, we prove several properties of the solution, and propose a novel approach for solving the problem. In Section 6.4, we apply the proposed approach to the turbopump concept design (Ahmadi et al., 2001), and to the development power line communication (Luh et al., 2009). Solutions provided in Chapter 5 of this thesis are used to benchmark the effectiveness of the approach in this chapter. Section 6.5 concludes this study. Proofs of all the results in this chapter can be found in Appendix D.

## 6.2 Problem Formulation

Again consider an NPD project consisting of *n* design activities. Let $d_{i,j}$ ($d_{i,j} \geq 0$) denote the information dependence level of activity *i* on activity *j*, where $i = 1, 2, \dots, n$ and $j = 1, 2, \dots, n$ are indexes for activities. Let $m = 1, 2, \dots, n$ denote the index for activity positions. Defining the decision variables (for $i = 1, 2, \dots, n$; $m = 1, 2, \dots, n$):

$$x_{im} = \begin{cases} 1 & \text{if activity } i \text{ is assigned to the } m^{\text{th}} \text{ position,} \\ 0 & \text{Otherwise.} \end{cases}$$

| Activities | 1 | 2 | … | $i$ | … | $n$ |
|---|---|---|---|---|---|---|
| 1 | 0 | $d_{1,2}$ | … | $d_{1,i}$ | … | $d_{1,n}$ |
| 2 | $d_{2,1}$ | 0 | … | $d_{2,i}$ | … | $d_{2,n}$ |
| … | … | … | 0 | … | … | … |
| $i$ | $d_{i,1}$ | $d_{i,2}$ | … | 0 | … | $d_{i,n}$ |
| … | … | … | … | … | 0 | … |
| $n$ | $d_{n,1}$ | $d_{n,2}$ | … | $d_{n,i}$ | … | 0 |

**Figure 6.1 NDSM representation of the optimization problem**

As illustrated in Figure 6.1, the DSM sequencing problem is to find an activity sequence that minimize feedbacks, which can be formulated as follows:

$$\left.\begin{aligned}
&Min\ f(x)=\sum_{i=1}^{n}\sum_{\substack{j=1\\ j\neq i}}^{n}\sum_{m=1}^{n}\left(x_{im}\times\sum_{h=m+1}^{n}x_{jh}\right)d_{i,j}\\
&\text{Subject to:}\\
&\sum_{i=1}^{n}x_{im}=1,\quad \text{for } m=1,2,\dots,n\\
&\sum_{m=1}^{n}x_{im}=1,\quad \text{for } i=1,2,\dots,n\\
&x_{im}\in\{0,1\},\quad \text{for } i=1,2,\dots,n \text{ and } m=1,2,\dots,n
\end{aligned}\right\}\qquad (6.1)$$

## 6.3 The Proposed Approach

### 6.3.1 Preliminaries

This section describes the theoretical basis of the proposed approach for solving problem (6.1). We first introduce following definitions:

**Definition 6.1 (block).** *A block is a smaller NDSM that contains a series of consecutive activities. For instance, Figure 6.2 shows a block $B_J$ which contains activities from j to $i-1$.*

**Definition 6.2 (fold).** *Fold is the operations of treating the activities in a block as a single activity, summing up the activities' feedbacks and inputs. For example, let $J=(j,\dots,i-1)$ denote the activities that belong to block $B_J$. Folding the block $B_J$ in Figure 6.2(b) into a single activity J', we get the NDSM shown in Figure 6.3.*

| Activities | $1$ | $\dots$ | $j$ | $j+1$ | $\dots$ | $i-1$ | $\dots$ | $n$ |
|---|---|---|---|---|---|---|---|---|
| $1$ | 0 | $\dots$ | $d_{1,j}$ | $d_{1,j+1}$ | $\dots$ | $d_{1,i-1}$ | $\dots$ | $d_{1,n}$ |
| $\dots$ | $\dots$ | 0 | $\dots$ | $\dots$ | $\dots$ | $\dots$ | $\dots$ | $\dots$ |
| $j$ | $d_{j,1}$ | $\dots$ | 0 | $d_{j,j+1}$ | $\dots$ | $d_{j,i-1}$ | $\dots$ | $d_{j,n}$ |
| $j+1$ | $d_{j+1,1}$ | $\dots$ | $d_{j+1,j}$ | 0 | $\dots$ | $d_{j+1,i-1}$ | $\dots$ | $d_{j+1,n}$ |
| $\dots$ | $\dots$ | $\dots$ | $\dots$ | $\dots$ | 0 | $\dots$ | $\dots$ | $\dots$ |
| $i-1$ | $d_{i-1,1}$ | $\dots$ | $d_{i-1,j}$ | $d_{i-1,j+1}$ | $\dots$ | 0 | $\dots$ | $d_{i-1,n}$ |
| $\dots$ | $\dots$ | $\dots$ | $\dots$ | $\dots$ | $\dots$ | $\dots$ | 0 | $\dots$ |
| $n$ | $d_{n,1}$ | $\dots$ | $d_{n,j}$ | $d_{n,j+1}$ | $\dots$ | $d_{n,i-1}$ | $\dots$ | 0 |

**(a)** An $n \times n$ NDSM

| Activities | $j$ | $j+1$ | $\dots$ | $i-1$ |
|---|---|---|---|---|
| $j$ | 0 | $d_{j,j+1}$ | $\dots$ | $d_{j,i-1}$ |
| $j+1$ | $d_{j+1,j}$ | 0 | $\dots$ | $d_{j+1,i-1}$ |
| $\dots$ | $\dots$ | $\dots$ | 0 | $\dots$ |
| $i-1$ | $d_{i-1,j}$ | $d_{i-1,j+1}$ | $\dots$ | 0 |

**(b)** A block $B_J$

**Figure 6.2 Definition of a block**

| Activities | $1$ | $\dots$ | $j-1$ | $J'$ | $i$ | $\dots$ | $n$ |
|---|---|---|---|---|---|---|---|
| $1$ | 0 | $\dots$ | $d_{1,j-1}$ | $\sum_{k\in J} d_{1,k}$ | $d_{1,i}$ | $\dots$ | $d_{1,n}$ |
| $\dots$ | $\dots$ | 0 | $\dots$ | $\dots$ | $\dots$ | $\dots$ | $\dots$ |
| $j-1$ | $d_{j-1,1}$ | $\dots$ | 0 | $\sum_{k\in J} d_{j-1,k}$ | $d_{j-1,i}$ | $\dots$ | $d_{j-1,n}$ |
| $J'$ | $\sum_{k\in J} d_{k,1}$ | $\dots$ | $\sum_{k\in J} d_{k,j-1}$ | 0 | $\sum_{k\in J} d_{k,i}$ | $\dots$ | $\sum_{k\in J} d_{k,n}$ |
| $i$ | $d_{i,1}$ | $\dots$ | $d_{i,j-1}$ | $\sum_{k\in J} d_{i,k}$ | 0 | $\dots$ | $d_{i,n}$ |
| $\dots$ | $\dots$ | $\dots$ | $\dots$ | $\dots$ | $\dots$ | 0 | $\dots$ |
| $n$ | $d_{n,1}$ | $\dots$ | $d_{n,j-1}$ | $\sum_{k\in J} d_{n,k}$ | $d_{n,i}$ | $\dots$ | 0 |

**Figure 6.3 Resulting NDSM by folding block $B_J$**

Assume that in a $n \times n$ NDSM, activities are executed in the sequence of $\{1, \dots j, \dots, i-1, i, \dots, n\}$, let $J = (j, \dots, i-1)$ be the activities that belong to block $B_J$:

**THEOREM 6.1 (block-activity exchange).** *If* $\sum_{k\in J} d_{k,i} > \sum_{k\in J} d_{i,k}$ , *then through exchanging block* $B_J$ *and activity* $i$, *the feedbacks in the NDSM can be reduced by*

$\sum_{k\in J}\left(d_{k,i}-d_{i,k}\right).$

Proofs of all the results in this chapter are provided in Appendix D.

Theorem 6.1 shows that if the sum of feedbacks from activity $i$ to block $B_J$ are larger than the sum of inputs from block $B_J$ to activity $i$, i.e., if $\sum_{k\in J} d_{k,i} > \sum_{k\in J} d_{i,k}$, then the sequence of block $B_J$ and activity $i$ should be exchanged.

In a $n\times n$ NDSM, suppose that activities are executed in the order of $\{1,...j-1, j,..., i-1, i,...h-1, h, h+1,...,n\}$, let $J=(j,...,i-1)$ be the activities that belong to block $B_J$, and $I=(i,...,h-1)$ denote the activities from $i$ to $h-1$. Define:

$$\theta_J^h=\sum_{k\in J}\sum_{r\in I}\left(d_{k,r}-d_{r,k}\right)+\sum_{k\in J}\left(d_{k,h}-d_{h,k}\right)+\sum_{r\in I}\left(d_{r,h}-d_{h,r}\right) \tag{6.2}$$

**Theorem 6.2 (block-activity exchange).** *In the NDSM, if $\theta_J^h>0$, then exchanging the sequence of block $B_J$ and activity $h$ results in a feedback reduction of $\theta_J^h$.*

Theorem 6.2 establishes the rule of exchanging a block with a non-adjacent activity for feedback reduction. It shows that in a NDSM, if $\theta_J^h>0$, then the sequence of block $B_J$ and activity $h$ should be exchanged. In what follows, we will establish the rule of exchanging two blocks of activities in a NDSM. The results are summarized in Theorem 6.3 and 6.4.

Suppose that in a $n\times n$ NDSM, activities are executed in the sequence of $\{1,...,j-1, j,...,i-1, i,...,h-1, h,...,n\}$. Let $J=(j,...,i-1)$ denote the activities that belong to block $B_J$, and $I=(i,...,h-1)$ denote the activities that belong to block $B_I$:

**THEOREM 6.3 (block-block exchange).** *If* $\sum_{k\in J}\sum_{r\in I} d_{k,r} > \sum_{k\in J}\sum_{r\in I} d_{r,k}$ *, then through exchanging block* $B_J$ *and block* $B_I$*, the feedbacks in the NDSM can be reduced by* $\sum_{k\in J}\sum_{r\in I}\left(d_{k,r} - d_{r,k}\right)$.

Theorem 6.3 reveals that, in a NDSM, if the feedbacks from block $B_I$ to block $B_J$ are larger than the inputs from block $B_J$ to $B_I$, i.e., if $\sum_{k\in J}\sum_{r\in I} d_{k,r} > \sum_{k\in J}\sum_{r\in I} d_{r,k}$ , then the sequence of block $B_J$ and block $B_I$ should be exchanged. Note that if we fold the activities in block $B_I$ into a single activity, then the result in Theorem 6.3 is equivalent to that in Theorem 6.1. Let us use a real NDSM published in Chen et al. (2004) to further illustrate these. The original DSM of the burn-in system, which is systematically constructed based on the information contained in QFD, consists of twelve activities. For easy presentation, we select nine activities whose dependencies relations are shown in Figure 6.4(a).

Let block $B_J$ contains activities (3,4,5), and block $B_I$ contains activities 6 and 7, mathematically, $J = (3,4,5)$ and $I = (6,7)$. The sum of feedbacks from block $B_I$ to $B_J$, which is highlighted in grey in Figure 6.4(a), is $\sum_{k\in J}\sum_{r\in I} d_{k,r} = 1.46$. The sum of inputs from block $B_J$ to block $B_I$, which is highlighted in blue in Figure 4(a), is $\sum_{k\in J}\sum_{r\in I} d_{r,k} = 0.89$. Based on Theorem 6.3, exchanging blocks $B_J$ and $B_I$ would lead to a feedback reduction of 0.57. The resulting NDSM is shown in Figure 6.4(b). If we fold activities 6 and 7 in block $B_I$ into a single activity 6', by definition 6.2, we can get the folded NDSM in Figure 6.4(c). Clearly, the sum of feedbacks from activity 6' to block $B_J$, highlighted in grey in Figure 6.4(c), is $\sum_{k\in J} d_{k,6'} = 1.46$. The sum of inputs from $B_J$ to activity 6', which is highlighted in blue in Figure 6.4(c), is $\sum_{k \in J} d_{6',k} = 0.89$. Then according to Theorem 6.1, exchanging block $B_J$ and activity 6' would lead to a feedback reduction of 0.57. The resulting NDSM is presented in Figure 6.4(d).

| Activities | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 0 | 0.34 | 0 | 0 | 0 | 0.29 | 0.04 | 0.33 | 0 |
| 2 | 0 | 0 | 0.24 | 0 | 0.02 | 0 | 0 | 0.04 | 0 |
| 3 | 0.15 | 0 | 0 | 0 | 0.14 | 0.02 | 0.55 | 0.26 | 0 |
| 4 | 0.34 | 0.35 | 0.04 | 0 | 0 | 0.53 | 0 | 0 | 0 |
| 5 | 0.34 | 0.27 | 0.19 | 0.2 | 0 | 0.16 | 0.2 | 0 | 0 |
| 6 | 0 | 0.05 | 0.14 | 0.3 | 0 | 0 | 0 | 0 | 0 |
| 7 | 0.2 | 0.02 | 0.41 | 0 | 0.04 | 0 | 0 | 0 | 0.12 |
| 8 | 0 | 0 | 0.42 | 0 | 0 | 0 | 0 | 0 | 0.18 |
| 9 | 0 | 0 | 0 | 0 | 0.11 | 0 | 0.21 | 0.24 | 0 |

**(a)** Original NDSM (from Chen et al., 2004)

| Activities | 1 | 2 | 6 | 7 | 3 | 4 | 5 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 0 | 0.34 | 0.29 | 0.04 | 0 | 0 | 0 | 0.33 | 0 |
| 2 | 0 | 0 | 0 | 0 | 0.24 | 0 | 0.02 | 0.04 | 0 |
| 6 | 0 | 0.05 | 0 | 0 | 0.14 | 0.3 | 0 | 0 | 0 |
| 7 | 0.2 | 0.02 | 0 | 0 | 0.41 | 0 | 0.04 | 0 | 0.12 |
| 3 | 0.15 | 0 | 0.02 | 0.55 | 0 | 0 | 0.14 | 0.26 | 0 |
| 4 | 0.34 | 0.35 | 0.53 | 0 | 0.04 | 0 | 0 | 0 | 0 |
| 5 | 0.34 | 0.27 | 0.16 | 0.2 | 0.19 | 0.2 | 0 | 0 | 0 |
| 8 | 0 | 0 | 0 | 0 | 0.42 | 0 | 0 | 0 | 0.18 |
| 9 | 0 | 0 | 0 | 0.21 | 0 | 0 | 0.11 | 0.24 | 0 |

**(b)** Improved NDSM based on Theorem 6.3

| Activities | 1 | 2 | 3 | 4 | 5 | 6' | 8 | 9 |
|---|---|---|---|---|---|---|---|---|
| 1 | 0 | 0.34 | 0 | 0 | 0 | 0.33 | 0.33 | 0 |
| 2 | 0 | 0 | 0.24 | 0 | 0.02 | 0 | 0.04 | 0 |
| 3 | 0.15 | 0 | 0 | 0 | 0.14 | 0.57 | 0.26 | 0 |
| 4 | 0.34 | 0.35 | 0.04 | 0 | 0 | 0.53 | 0 | 0 |
| 5 | 0.34 | 0.27 | 0.19 | 0.2 | 0 | 0.36 | 0 | 0 |
| 6' | 0.2 | 0.07 | 0.55 | 0.3 | 0.04 | 0 | 0 | 0.12 |
| 8 | 0 | 0 | 0.42 | 0 | 0 | 0 | 0 | 0.18 |
| 9 | 0 | 0 | 0 | 0 | 0.11 | 0.21 | 0.24 | 0 |

**(c)** Folded NDSM

| Activities | 1 | 2 | 6' | 3 | 4 | 5 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|
| 1 | 0 | 0.34 | 0.33 | 0 | 0 | 0 | 0.33 | 0 |
| 2 | 0 | 0 | 0 | 0.24 | 0 | 0.02 | 0.04 | 0 |
| 6' | 0.2 | 0.07 | 0 | 0.55 | 0.3 | 0.04 | 0 | 0.12 |
| 3 | 0.15 | 0 | 0.57 | 0 | 0 | 0.14 | 0.26 | 0 |
| 4 | 0.34 | 0.35 | 0.53 | 0.04 | 0 | 0 | 0 | 0 |
| 5 | 0.34 | 0.27 | 0.36 | 0.19 | 0.2 | 0 | 0 | 0 |
| 8 | 0 | 0 | 0 | 0.42 | 0 | 0 | 0 | 0.18 |
| 9 | 0 | 0 | 0.21 | 0 | 0 | 0.11 | 0.24 | 0 |

**(d)** Improved folded NDSM based on Theorem 6.1

**Figure 6.4 Illustration of Theorem 6.3 and Theorem 6.1: a practical example**

Suppose that in a $n \times n$ NDSM, activities are executed in the sequence of $\{1,...j-1, j,..., i-1, i,...,h-1, h,...,k-1, k...,n\}$. Let $J = (j,...,i-1)$ be the activities that belong to block $B_J$, and $H = (h,...,k-1)$ be the activities that belong to block $B_H$. Let $I = (i,...,h-1)$ denote the activities from $i$ to $h-1$. Define:

$$\omega_J^H = \sum_{r \in J} \sum_{p \in I} \left(d_{r,p} - d_{p,r}\right) + \sum_{r \in J} \sum_{p \in H} \left(d_{r,p} - d_{p,r}\right) + \sum_{r \in I} \sum_{p \in H} \left(d_{r,p} - d_{p,r}\right) \tag{6.3}$$

**THEOREM 6.4 (block-block exchange).** *In a NDSM, if $\omega_J^H > 0$, then exchanging block $B_J$ and block $B_H$ leads to a feedback reduction of $\omega_J^H$.*

Theorem 6.4 further establishes the rule of exchanging a block with a non-adjacent block for feedback reduction. It reveals that if $\omega_J^H > 0$, then the sequence of block $B_J$ and block $B_H$ should be exchanged. Similarly, if block $B_H$ is folded into a single activity, then the result in Theorem 6.4 is equivalent to that in Theorem 6.2.

To illustrate the results of Theorem 6.2 and Theorem 6.4, consider the original DSM shown in Figure 6.5(a). Let block $B_J$ contains activities 4 and 5, and block $B_H$ contains activities 8 and 9, mathematically, $J=(4,5)$, $I=(6,7)$, and $H=(8,9)$. From (6.3), we can get: $\omega_J^H = \sum_{r\in J}\sum_{p\in I}\left(d_{r,p}-d_{p,r}\right)+\sum_{r\in J}\sum_{p\in H}\left(d_{r,p}-d_{p,r}\right)+\sum_{r\in I}\sum_{p\in H}\left(d_{r,p}-d_{p,r}\right)=0.35$. To make our statement more legible, in Figure 6.5(a), the positive items in $\omega_J^H$ are highlighted in grey, while the negative items are highlighted in blue. According to Theorem 6.4, the sequence of block $B_J$ and block $B_H$ should be exchanged. The resulting NDSM is presented in Figure 6.5(b) where the feedbacks are reduced by 0.35. Folding activities 8 and 9 in block $B_H$ into a single activity 8', we get the DSM shown in Figure 6.5(c). Inspection of block $B_J$ and activity 8' shows that $\theta_J^{8'} = \sum_{k\in J}\sum_{r\in I}\left(d_{k,r}-d_{r,k}\right)+\sum_{k\in J}\left(d_{k,8'}-d_{8',k}\right)+\sum_{r\in I}\left(d_{r,8'}-d_{8',r}\right)=0.35$, where in Figure 6.5(c) the positive items are highlighted in grey, and the negative items are highlighted in blue. Based on Theorem 6.2, the sequence of block $B_J$ and activity 8' should be exchanged. The resulting NDSM is shown in Figure 6.5(d) where the feedbacks are reduced by 0.35. These results imply that through the fold operation, a block has similar properties to a single activity.

| Activities | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 0 | 0.34 | 0 | 0 | 0 | 0.29 | 0.04 | 0.33 | 0 |
| 2 | 0 | 0 | 0.24 | 0 | 0.02 | 0 | 0 | 0.04 | 0 |
| 3 | 0.15 | 0 | 0 | 0 | 0.14 | 0.02 | 0.55 | 0.26 | 0 |
| 4 | 0.34 | 0.35 | 0.04 | 0 | 0 | 0.53 | 0 | 0 | 0 |
| 5 | 0.34 | 0.27 | 0.19 | 0.2 | 0 | 0.16 | 0.2 | 0 | 0 |
| 6 | 0 | 0.05 | 0.14 | 0.3 | 0 | 0 | 0 | 0 | 0 |
| 7 | 0.2 | 0.02 | 0.41 | 0 | 0.04 | 0 | 0 | 0 | 0.12 |
| 8 | 0 | 0 | 0.42 | 0 | 0 | 0 | 0 | 0 | 0.18 |
| 9 | 0 | 0 | 0 | 0 | 0.11 | 0 | 0.21 | 0.24 | 0 |

**(a)** Original NDSM (from Chen et al., 2004)

| Activities | 1 | 2 | 3 | 8 | 9 | 6 | 7 | 4 | 5 |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 0 | 0.34 | 0 | 0.33 | 0 | 0.29 | 0.04 | 0 | 0 |
| 2 | 0 | 0 | 0.24 | 0.04 | 0 | 0 | 0 | 0 | 0.02 |
| 3 | 0.15 | 0 | 0 | 0.26 | 0 | 0.02 | 0.55 | 0 | 0.14 |
| 8 | 0 | 0 | 0.42 | 0 | 0.18 | 0 | 0 | 0 | 0 |
| 9 | 0 | 0 | 0 | 0.24 | 0 | 0 | 0.21 | 0 | 0.11 |
| 6 | 0 | 0.05 | 0.14 | 0 | 0 | 0 | 0 | 0.3 | 0 |
| 7 | 0.2 | 0.02 | 0.41 | 0 | 0.12 | 0 | 0 | 0 | 0.04 |
| 4 | 0.34 | 0.35 | 0.04 | 0 | 0 | 0.53 | 0 | 0 | 0 |
| 5 | 0.34 | 0.27 | 0.19 | 0 | 0 | 0.16 | 0.2 | 0.2 |  |

**(b)** Improved NDSM based on Theorem 6.4

| Activities | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8' |
|---|---|---|---|---|---|---|---|---|
| 1 | 0 | 0.34 | 0 | 0 | 0 | 0.29 | 0.04 | 0.33 |
| 2 | 0 | 0 | 0.24 | 0 | 0.02 | 0 | 0 | 0.04 |
| 3 | 0.15 | 0 | 0 | 0 | 0.14 | 0.02 | 0.55 | 0.26 |
| 4 | 0.34 | 0.35 | 0.04 | 0 | 0 | 0.53 | 0 | 0 |
| 5 | 0.34 | 0.27 | 0.19 | 0.2 | 0 | 0.16 | 0.2 | 0 |
| 6 | 0 | 0.05 | 0.14 | 0.3 | 0 | 0 | 0 | 0 |
| 7 | 0.2 | 0.02 | 0.41 | 0 | 0.04 | 0 | 0 | 0.12 |
| 8' | 0 | 0 | 0.42 | 0 | 0.11 | 0 | 0.21 | 0 |

**(c)** Folded DSM

| Activities | 1 | 2 | 3 | 8' | 6 | 7 | 4 | 5 |
|---|---|---|---|---|---|---|---|---|
| 1 | 0 | 0.34 | 0 | 0.33 | 0.29 | 0.04 | 0 | 0 |
| 2 | 0 | 0 | 0.24 | 0.04 | 0 | 0 | 0 | 0.02 |
| 3 | 0.15 | 0 | 0 | 0.26 | 0.02 | 0.55 | 0 | 0.14 |
| 8' | 0 | 0 | 0.42 | 0 | 0 | 0.21 | 0 | 0.11 |
| 6 | 0 | 0.05 | 0.14 | 0 | 0 | 0 | 0.3 | 0 |
| 7 | 0.2 | 0.02 | 0.41 | 0.12 | 0 | 0 | 0 | 0.04 |
| 4 | 0.34 | 0.35 | 0.04 | 0 | 0.53 | 0 | 0 | 0 |
| 5 | 0.34 | 0.27 | 0.19 | 0 | 0.16 | 0.2 | 0.2 | 0 |

**(d)** Improved folded DSM based on Theorem 6.2

**Figure 6.5 Illustration of Theorem 6.4 and Theorem 6.2: a practical example**

## 6.3.2 The Solution Strategy

In this section, we will propose a novel solution strategy for large-scale DSM sequencing problem. The proposed solution strategy consists of two heuristic procedures. The first heuristic Procedure 6.1 is to improve a feasible solution by using exchange. Based on the solution identified, Procedure 6.2 can be utilized to solve large-scale DSM sequencing problem.

**Procedure 6.1**

*Step 0* (Initiation). Given an initial activity sequence, calculate the total feedbacks $f$, and store it as the best sequence. Set the first activity as the mobile-activity $j$.

*Step 1.* Set $J = (j)$.

*Step 2.* For each activity *h* that follows activity *j*, calculate $\theta_J^h$ by equation (6.2): if $\theta_J^h > 0$, exchange block $B_J$ with activity *h*, save the result as the new best sequence, and set $f = f - \theta_J^h$; otherwise, add activity *h* to *J*.

*Step 3.* Set the next activity as the new mobile-activity $j$, and go back to *Step 1* until all the activities have been set as the mobile-activity.

As stated earlier, decomposition method can provide an efficient way to solve large-scale DSM sequencing problem. The basic idea of such method is to break the original problem into small sub-problems that are easier to solve, and then combine the sub-problem solutions into a solution to the original problem. As we discussed in previous section, through the fold operation, a block has similar properties to a single activity. Based on these ideas, we develop the following Procedure 6.2. Let $N_k$ be the number of activities contained in a sub-problem *k*.

**Procedure 6.2**

*Step 1* (Construct sub-problems). Based on the feasible solution identified by Procedure 6.1, select activities from position 1 to position $N_1$ to obtain block $B_1$ and the first sub-problem. Fold the activities in block $B_1$ into a single activity 1', and update activity dependencies. Select activity 1' and activities from position $N_1 + 1$ to position $N_1 + N_2 - 1$ to obtain block $B_2$ and the second sub-problem. Fold the activities in block $B_2$ into a single activity 2', and update activity dependencies. Repeat until the last activity n is reached. Step 2 (Solve sub-problems). For each sub-problem, compute the optimal solution by the branch-and-bound method.

*Step 2* (Solve sub-problems). The sub-problems are solved to optimality by commercial solvers, leading to improved sub-problem sequences $S_k^*$ ($k = 1,2,...,K$).

*Step 3*. The improved sub-problem sequences are concatenated into an improved sequence of the overall problem. In particular, in the improved sub-problem sequence $S_k^*$, replace activity *k'* with the improved sub-problem sequence $S_{k-1}^*$. Repeat until activity 1' is replaced with the improved sub-problem sequence $S_1^*$.

In *Step 1* of above procedure, when the sub-problems are constructed and solved, the sum of super-diagonal numbers that locates outside each sub-problem (i.e. block) will not be affected. In *Step 3*, when the improved sub-problem sequences are concatenated, we can get an improved sequence of the overall problem. The amount of improvement of the objective values in the overall problem equals to the sum of improvements of the objective values of the sub-problems. Thus, Procedure 6.2 guarantees that the final activity sequence is better than the initial one. Note that in *Step 2*, the sub-problems are solved in parallel.

To illustrate Procedure 6.2, consider the DSM in Figure 6.5(b). Set $N_k = 5$ , we get the first sub-problem in Figure 6.6(a). Folding the activities in the first sub-problem into a single activity 1', we obtain the second sub-problem in Figure 6.6(b). We then solve the two sub-problems in parallel and obtain the optimal solutions of $\{3,8,9,2,1\}$ and $\{7,6,1',4,5\}$. Based on the optimal sub-problem solutions, the sum of super-diagonal numbers in sub-problem 1 is reduced from 1.39 to 0.59, and the sum of super-diagonal numbers in sub-problem 2 is reduced from 1.72 to 1.55. Finally, by combining these two solutions, we get the solution of the overall problem shown in

Figure 6.6(c), where the sum of super-diagonal numbers is reduced from 3.11 to 2.14. Clearly, the amount of improvement of the objective values in the overall problem equals to the sum of improvements of the objective values of the sub-problems. In the next section, we will show that based on above procedure, relatively good solutions can be easily obtained.

| Activities | 1 | 2 | 3 | 8 | 9 |
|---|---|---|---|---|---|
| 1 | ■ | 0.34 | 0 | 0.33 | 0 |
| 2 | 0 | ■ | 0.24 | 0.04 | 0 |
| 3 | 0.15 | 0 | ■ | 0.26 | 0 |
| 8 | 0 | 0 | 0.42 | ■ | 0.18 |
| 9 | 0 | 0 | 0 | 0.24 | ■ |

**(a)** Sub-problem 1

| Activities | 1' | 6 | 7 | 4 | 5 |
|---|---|---|---|---|---|
| 1' | ■ | 0.31 | 0.8 | 0 | 0.27 |
| 6 | 0.19 | ■ | 0 | 0.3 | 0 |
| 7 | 0.75 | 0 | ■ | 0 | 0.04 |
| 4 | 0.73 | 0.53 | 0 | ■ | 0 |
| 5 | 0.8 | 0.16 | 0.2 | 0.2 | ■ |

**(b)** Sub-problem 2

| Activities | 7 | 6 | 3 | 8 | 9 | 2 | 1 | 4 | 5 |
|---|---|---|---|---|---|---|---|---|---|
| 7 | ■ | 0 | 0.41 | 0 | 0.12 | 0.02 | 0.2 | 0 | 0.04 |
| 6 | 0 | ■ | 0.14 | 0 | 0 | 0.05 | 0 | 0.3 | 0 |
| 3 | 0.55 | 0.02 | ■ | 0.26 | 0 | 0 | 0.15 | 0 | 0.14 |
| 8 | 0 | 0 | 0.42 | ■ | 0.18 | 0 | 0 | 0 | 0 |
| 9 | 0.21 | 0 | 0 | 0.24 | ■ | 0 | 0 | 0 | 0.11 |
| 2 | 0 | 0 | 0.24 | 0.04 | 0 | ■ | 0 | 0 | 0.02 |
| 1 | 0.04 | 0.29 | 0 | 0.33 | 0 | 0.34 | ■ | 0 | 0 |
| 4 | 0 | 0.53 | 0.04 | 0 | 0 | 0.35 | 0.34 | ■ | 0 |
| 5 | 0.2 | 0.16 | 0.19 | 0 | 0 | 0.27 | 0.34 | 0.2 | ■ |

**(c)** Final solution by applying Procedure 2

**Figure 6.6 Illustration of Procedure 6.2**

## 6.4 Computational Results

In this section, we first apply the proposed Procedure 6.2 to the turbopump concept design (Ahmadi et al., 2001), and to the development power line communication (Luh et al., 2009). Solutions provided in Chapter 5 of this thesis are used to benchmark the effectiveness of Procedure 6.2. After that we generate six groups of random problems in order to further examine the performance of the proposed procedure. All the experiments are performed on a Pentium Dual-Core

2.1GHz Compaq laptop with 2G memory.

### 6.4.1 Application Results

**Case 1.** The NDSM for the turbopump concept design has been shown in Figure 5.8. To compare the application results, we use the same initial solution as has been shown in Figure 5.7(a), and set the same value of $N = 11$.

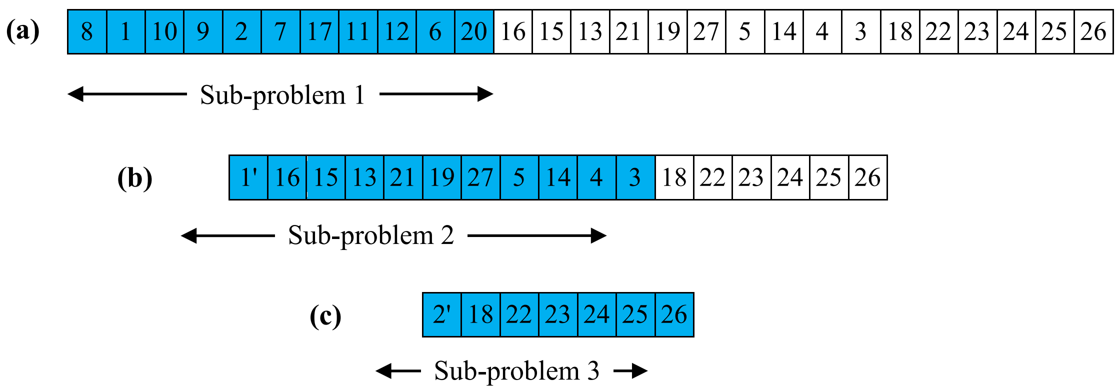


**Figure 6.7 The solution strategy for the turbopump concept design**

We then apply Procedure 6.2. As illustrated in Figure 6.7, the activities from position 1 to position 11, i.e. activities $(8,1,10,9,2,7,17,11,12,6,20)$, constitute block $B_1$ and sub-problem 1. Then, fold block $B_1$ into a single activity 1', and choose activity 1' and activities from position 12 to position 21 to get block $B_2$ and sub-problem 2. Similarly, fold block $B_2$ into a single activity 2', and select activity 2' and the remaining activities to obtain sub-problem 3. The Branch-and-Bound method is then used to solve the sub-problems. The optimal sub-problem solutions are then merged into a solution of the overall problem. Based on these steps, we get the same result as has shown in Figure 5.10. The computation time for Procedure 6.2 is 319.5 minutes.

**Case 2.** The data set for the development power line communication has been shown in Figure 5.11. Similarly, we choose the same initial solution, and set the same value of $N = 8$. By applying Procedure 6.2 in this chapter, we also get the same result as has been illustrated in Figure 5.13. Here the computation time is 804.3 minutes.

### 6.4.2 Numerical Results

We perform six groups of random problems to examine how different problem structures (such as modularity, size and density) affect the performance of our proposed approach and the solution. The Procedure 6.2 was coded in Matlab. The matrices that are tested include $n = 25$ and $n = 50$ activities. The information dependency among activities, $d_{i,j}$, are uniformly distributed in the interval $[0,1)$. Let $\alpha = \frac{D}{n \times (n-1)}$ be the density level in a DSM, where $D$ is the number of non-zero $d_{i,j}$ in a DSM. Three levels of density, i.e., $\alpha = 33\%$, $\alpha = 67\%$, and $\alpha = 100\%$ are investigated.

We generate 10 random instances for each group. For each random instance, the sizes of a sub-problem we test are $N_k = 3\ \forall k$, $N_k = 4\ \forall k$, $N_k = 5\ \forall k$, $N_k = 6\ \forall k$, $N_k = \lfloor 200/D_k \rfloor$, where $D_k$ is the number of non-zero $d_{i,j}$ in a sub-problem *k*. The reasons that we set $N_k$ equal to the maximum integer that less than $\lfloor 200/D_k \rfloor$ are twofold. First, such setting allows $N_k$ varying between sub-problems. Second, from our computational experience, the computation time for solving a sub-problem to optimality is mainly determined by the number of inequality constraints in the optimization problem (1)-(5), i.e. $N_k \times D_k$. To be able to report our computational findings within reasonable time, we set $N_k \times D_k \le 200$, and so $N_k = \lfloor 200/D_k \rfloor$. Note that when $\alpha = 100\%$, $\lfloor 200/D_k \rfloor$ equals to 6.

To report our computational results, let *fv* be the sum of super-diagonal numbers obtained from the proposed approach, $P=(Ifv-fv)/Ifv$ be the percentages of reduction of the sum of super-diagonal numbers, where *Ifv* is the initial values of the sum of super-diagonal numbers. The computational results for each group of trials are shown in Tables 6.1-6.2.

**Table 6.1 Computation results of the proposed approach ($n=25$)**

| **Density $\alpha$** | $N_k$ | **Average Computation time (minutes)** | **Average *fv*** | **Average *P*** |
|---|---|---|---|---|
| 33% | 3 | 0.001 | 28.01 | 42.16% |
| | 4 | 0.006 | 27.98 | 42.23% |
| | 5 | 0.108 | 27.81 | 42.56% |
| | 6 | 3.385 | 27.65 | 42.91% |
| | $\lfloor 200/D_k \rfloor$ | 158.627 | 27.41 | 43.38% |
| 67% | 3 | 0.014 | 73.66 | 25.49% |
| | 4 | 0.016 | 73.64 | 25.51% |
| | 5 | 0.201 | 73.51 | 25.64% |
| | 6 | 4.808 | 73.44 | 25.72% |
| | $\lfloor 200/D_k \rfloor$ | 77.044 | 73.38 | 25.77% |
| 100% | 3 | 0.001 | 127.35 | 15.29% |
| | 4 | 0.015 | 127.22 | 15.38% |
| | 5 | 0.487 | 127.02 | 15.51% |
| | 6 | 29.176 | 126.88 | 15.61% |

**Table 6.2 Computation results of the proposed approach ($n=50$)**

| **Density $\alpha$** | $N_k$ | **Average Computation time (minutes)** | **Average *fv*** | **Average *P*** |
|---|---|---|---|---|
| 33% | 3 | 0.051 | 134.19 | 33.02% |
| | 4 | 0.057 | 133.89 | 33.16% |
| | 5 | 0.11 | 133.2 | 33.5% |
| | 6 | 1.5 | 133.34 | 33.43% |
| | $\lfloor 200/D_k \rfloor$ | 551.717 | 132.6 | 33.79% |
| 67% | 3 | 0.056 | 326.23 | 20.71% |
| | 4 | 0.062 | 325.78 | 20.82% |
| | 5 | 0.306 | 325.69 | 20.84% |
| | 6 | 23.88 | 325.34 | 20.92% |
| | $\lfloor 200/D_k \rfloor$ | 31.381 | 325.21 | 20.95% |
| 100% | 3 | 0.054 | 544.54 | 12.05% |
| | 4 | 0.069 | 544.43 | 12.06% |
| | 5 | 0.532 | 544.02 | 12.13% |
| | 6 | 47.217 | 543.57 | 12.2% |

Overall, Procedure 6.2 performs quite well. For instance, as shown in Table 6.1, for $n = 25$, $\alpha = 33\%$, and $N_k = \lfloor 200/D_k \rfloor$, the average reduction of the sum of super-diagonal numbers is 43.38%. Generally, the amount of reduction of the sum of super-diagonal numbers decreases with the density level and with the size of the matrix. For instance, when the size of the matrix is increased to 50 (for $\alpha = 33\%$ , and $N_k = \lfloor 200/D_k \rfloor$), the average reduction of the sum of super-diagonal numbers is decreased to 33.79%. In many cases, increasing the size of a sub-problem will lead to better final solutions, but not always. For instance, for $n = 50$, $\alpha = 33\%$, when $N_k$ is increased from 5 to 6, the average reduction of the sum of super-diagonal numbers is decreased from 33.5% to 33.43%. This is so because increasing the size of a sub-problem decreases the total number of sub-problems. Another advantage of Procedure 6.2 is that the impact of the size of the matrix on the computation time is quite small. This is so because the complexity of our Procedure 6.1 is O($n^2$), and all the sub-problems in Procedure 6.2 can be solved in parallel.

## 6.5 Conclusion

In this chapter, we have presented a novel approach to large-scale DSM sequencing problem. The contribution of the work is threefold. First, it establishes block-activity exchange rules (Theorem 6.1 and 6.2), and block-block exchange rules (Theorem 6.3 and 6.4), for direct feedback reduction. Second, a heuristic is proposed for improving feasible solutions by using exchange. Third, the results of theorems imply that through the fold operation, a block has similar properties to a single activity. In other words, we may treat the activities in a block as a single activity. Based on this idea, a new decomposition procedure is presented for solving large-scale DSM sequencing problem. To investigate its utility in solving real world problems, the proposed solution strategy has been applied to the turbopump conceptual design (Ahmadi et al., 2001), and to the development of power line communication (Luh et al., 2009).

We also perform six groups of random examples and show that good solutions can be easily obtained by Procedure 6.2. Compared to the decomposition approach presented in Chapter 5, one advantage of the approach in this chapter is that it can solve the sub-problems independently and in parallel.

There are two major limitations for the application of the methodologies developed in this chapter and Chapter 5. First, the present models only examine the activity sequencing problem with the objective of minimizing feedbacks, because it is simple and of practical importance. Moreover, it is also a common practice in previous activity sequencing models. As has been mentioned earlier, except for the objective of minimizing feedbacks, some other considerations have also been incorporated in sequencing design activities in a DSM. Therefore, our models may be extended to explore the activity sequencing problem with some other objectives.

Second, the studies presented in Chapter 5 and 6 of this thesis, assume that the information dependencies among activities can be estimated with reasonable accuracy. This assumption has been proved to hold in dozens of development projects, see e.g. Eppinger et al. (1994), Eppinger (2001), Browning and Eppinger (2002), Batallas and Yassine (2006), Yu et al. (2007), Sosa (2008), Tang et al. (2009), Collins et al. (2009), Voss and Hsuan (2009). However, there are still some projects where precise information of activity dependencies is not available (Chen et al., 2004; Karniel and Reich, 2009; Luh et al., 2009). To address this issue, in Chapter 7, we present a fuzzy approach for solving DSM sequencing problem with imprecise activity dependencies.

# CHAPTER 7

# A FUZZY APPROACH TO DSM SEQUENCING PROBLEM

## 7.1 Introduction

DSM provides a concise representation of an NPD process by showing information dependencies in a square matrix (Cho and Eppinger, 2005). In recent years, DSM has been regarded as an effective tool for modeling and improving NPD processes (Browning, 2001; Karniel and Reich, 2009; To et al., 2009). To accelerate NPD processes, the DSM approach suggests to re-sequencing the activities such that iterative behaviors are minimized in the matrix.

Quite a few studies have examined the activity sequencing problem in a DSM. For example, Rogers (1989) developed an expert system called DeMAID. The DeMAID was further extended to DeMAID/GA by incorporating GA to organize the activities (see e.g. Altus et al., 1996; McCulley and Bloebaum, 1996; Rogers, 1996). Kusiak and Wang (1993) presented an algorithm for ordering design activities in a DSM. Extensions of their work can be found in Kusiak et al. (1994), Kusiak et al. (1995), Tang et al. (2000), Zakarian and Kusiak (2001). Recently, there has been a growing interest in ordering design activities in a numerical DSM. Examples of such studies include Smith and Eppinger (1997a, 1997b), Ahmadi et al. (2001), Browning and Eppinger (2002), Chen and Li (2003), Abdelsalam and Bao (2006), Banerjee et al. (2007), Meier et al. (2007), Yu et al., (2007), Lancaster and Cheng (2008). These works have clearly shed light on the analysis of activity sequencing problem, and are useful to guide the practice when activity dependencies can be precisely estimated. However, all of these models, including the ones presented in Chapter 5 and 6 of this thesis, assume that activity dependencies can be accurately estimated.

As reported by Chen et al. (2004), as well as Karniel and Reich (2009), in many NPD processes, precise information of activity dependencies is not available, especially when the activities have never been performed before. To address this issue, one natural approach is to treat uncertain activity dependencies as random variables with specified probability distributions. However, for NPD processes, we are often lack of enough data to derive the probability distributions for activity dependencies. Fortunately, in such cases, fuzzy set theory can help us tackle uncertain activity dependencies. Moreover, compared to probability distributions, fuzzy sets are often easier to compute (Zimmermann, 1996; Dubois et al., 2003a; Wang, 2004; Liberatore, 2008).

Recently, Luh et al. (2009) proposed the concept of fuzzy DSM, and developed a heuristic to reorder design activities so as to minimize feedbacks. In this study we also resort to fuzzy set theory for tackling uncertain parameters. More specifically, our study complements the work of Luh et al. (2009) by developing a mathematical model for activity sequencing problem with a fuzzy representation of activity dependencies. The model can be utilized to predict the most likely, pessimistic and optimistic values of feedbacks. Since the problem is NP-complete, we have also developed a new hybrid approach, which embed an exact algorithm within a framework of local search heuristic, for solving large-scale problems. The proposed approach is applied to a real data set published in Eppinger (2001). Application result shows that the approach can help decision makers better manage NPD processes with uncertainty.

In the next section, we review concepts of fuzzy set theory used in this research, and formulate the activity sequencing problem with imprecise activity dependencies. Section 7.3 presents a methodology for solving the problem. In Section 7.4, we illustrate the methodology with a case study of the powertrain development at General Motors (Eppinger, 2001). Section 7.5 concludes this research.

## 7.2 Problem Formulation

We first introduce the following notation:

| | |
|---|---|
| $n$ | Total number of activities |
| $i, j$ | Indexes for activities, $i = 1, 2, \ldots, n$; $j = 1, 2, \ldots, n$ |
| $\tilde{d}_{i,j}$ | Fuzzy information dependence of activity *i* on activity *j* |
| $m, h$ | Indexes for activity positions, $m = 1, 2, \ldots, n$; $h = 1, 2, \ldots, n$ |

**Decision Variables:**

$$x_{im} = \begin{cases} 1 & \text{if activity } i \text{ is assigned to the } m^{\text{th}} \text{ position,} \\ 0 & \text{Otherwise.} \end{cases}$$

$$z_{i,j} = \begin{cases} 1 & \text{if activity } i \text{ is assigned to precede activity } j, \\ 0 & \text{Otherwise.} \end{cases}$$

As shown in Figure 7.1, we consider the NPD process consisting of *n* activities, numbered from 1 to *n*. Given uncertain activity dependencies defined by fuzzy sets, the objective is to find an activity sequence with minimum fuzzy total feedbacks, $\sum_{i=1}^{n} \sum_{j=i+1}^{n} \tilde{d}_{i,j}$.

| Activities | 1 | 2 | $\cdots$ | $i$ | $\cdots$ | $n$ |
|---|---|---|---|---|---|---|
| 1 | ■ | $\tilde{d}_{1,2}$ | $\cdots$ | $\tilde{d}_{1,i}$ | $\cdots$ | $\tilde{d}_{1,n}$ |
| 2 | $\tilde{d}_{2,1}$ | ■ | $\cdots$ | $\tilde{d}_{2,i}$ | $\cdots$ | $\tilde{d}_{2,n}$ |
| $\cdots$ | $\cdots$ | $\cdots$ | ■ | $\cdots$ | $\cdots$ | $\cdots$ |
| $i$ | $\tilde{d}_{i,1}$ | $\tilde{d}_{i,2}$ | $\cdots$ | ■ | $\cdots$ | $\tilde{d}_{i,n}$ |
| $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | ■ | $\cdots$ |
| $n$ | $\tilde{d}_{n,1}$ | $\tilde{d}_{n,2}$ | $\cdots$ | $\tilde{d}_{n,i}$ | $\cdots$ | ■ |

**Figure 7.1 Representation of the optimization problem**

## 7.2.1 Fuzzy Set Background

We use fuzzy triangular numbers to represent the uncertain activity dependencies. As shown in Figure 7.2, a fuzzy triangular number $\tilde{d}_{i,j}$ is characterized by a triplet ($d^1_{i,j}$, $d^2_{i,j}$, $d^3_{i,j}$), and its membership function $\mu_{\tilde{d}_{i,j}}(x)$ is defined as:

$$\mu_{\tilde{d}_{i,j}}(x) = \begin{cases} \dfrac{x - d^1_{i,j}}{d^2_{i,j} - d^1_{i,j}} & \text{if } d^1_{i,j} \le x \le d^2_{i,j}, \\[2ex] \dfrac{d^3_{i,j} - x}{d^3_{i,j} - d^2_{i,j}} & \text{if } d^2_{i,j} \le x \le d^3_{i,j}, \\[2ex] 0 & \text{otherwise.} \end{cases} \tag{7.1}$$

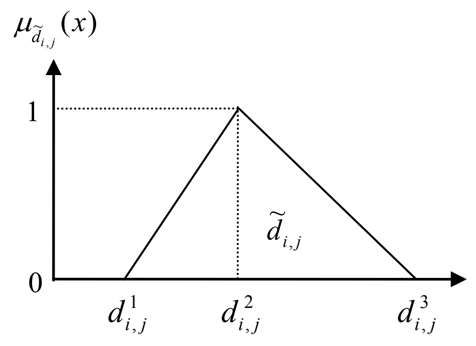


**Figure 7.2 Illustration of fuzzy triangular number $\tilde{d}_{i,j}$**

The above membership function specifies the degree of belief that the activity dependence value is *x*. For example, the vague and imprecision statement that information dependence of activity *i* on activity *j* is "low", might be represented by $(0,0.2,0.3)$. For $x = 0.2$, $\mu_{\tilde{d}_{i,j}}(x) = 1$, which means that most likely, the information dependence of activity *i* on activity *j* is 0.2. For $x = 0.1$, $\mu_{\tilde{d}_{i,j}}(x) = 0.5$, which means that the degree of belief that the information dependence of activity *i* on activity *j* equals 0.1, is 0.5.

Let $\tilde{A}$ and $\tilde{B}$ be two fuzzy triangular numbers characterized by $(a_1, a_2, a_3)$ and $(b_1, b_2, b_3)$, respectively. In the activity sequencing problem, the main arithmetic operations involved are fuzzy addition and fuzzy multiplication, which are defined as follows:

$$\tilde{A} \oplus \tilde{B} = (a_1 + b_1, a_2 + b_2, a_3 + b_3) \tag{7.2}$$

$$\tilde{A} \otimes \tilde{B} = (a_1 \times b_1, a_2 \times b_2, a_3 \times b_3) \tag{7.3}$$

Quite a few defuzzification methods can be used to rank fuzzy numbers. As reported by Dubois et al. (2003a), one of them, proposed by Yager (1981), turned out to be the most natural and convincing. In addition, Fortemps and Roubens (1996) proved that the method could also be derived using the area compensation method. These observations motivated us to use the method by Yager (1981) to compare the fuzzy total feedbacks of various activity sequences. The method calculates the integral of the mean of the $\alpha$-cut of a fuzzy number $\tilde{M}$, i.e.

$$D(\tilde{M}) = \frac{1}{2}\int_0^1 (m_\alpha^L + m_\alpha^R)\, d\alpha \tag{7.4}$$

Where $[m_\alpha^L, m_\alpha^R]$ is the $\alpha$-cut of $\tilde{M}$. More specifically, the $\alpha$-cut of $\tilde{M}$ is crisp subset of elements whose degree of membership is not less than $\alpha$ ($0 < \alpha \le 1$). For the two fuzzy triangular numbers $\widetilde{A}$ and $\widetilde{B}$ that are characterized by $(a_1, a_2, a_3)$ and $(b_1, b_2, b_3)$, it can be verified that definition (7.4) satisfies following desirable properties:

**P1.** $D\left(\widetilde{A} \oplus \widetilde{B}\right) = D(\widetilde{A}) + D(\widetilde{B})$ (7.5)

**P2.** $D(r\widetilde{A}) = rD(\widetilde{A})$, for any real $r$ (7.6)

**P3.** $D\left(r_1\widetilde{A} \oplus r_2\widetilde{B}\right) = r_1 D(\widetilde{A}) + r_2 D(\widetilde{B})$, for any real $r_1$ and $r_2$ (7.7)

The proofs of above properties are provided in Appendix E.

### 7.2.2 The Mathematical Model

As mentioned earlier, the objective is to find an activity sequence with minimum fuzzy total feedbacks, which can be formulated as:

$$Min \sum_{i=1}^{n} \sum_{\substack{j=1 \\ j \neq i}}^{n} \widetilde{d}_{i,j} z_{i,j} \tag{7.8}$$

$$\text{s.t.} \quad z_{i,j} \geq x_{im} - \sum_{h=1}^{m-1} x_{jh}, \quad \text{for } m = 1, 2, \ldots, n \tag{7.9}$$

$$\sum_{i=1}^{n} x_{im} = 1, \quad \text{for } m = 1, 2, \ldots, n \tag{7.10}$$

$$\sum_{m=1}^{n} x_{im} = 1, \quad \text{for } i = 1, 2, \ldots, n \tag{7.11}$$

$$x_{im}, z_{i,j} \in \{0, 1\}, \quad \text{for } i = 1, 2, \ldots, n; \; j = 1, 2, \ldots, n; m = 1, 2, \ldots, n \tag{7.12}$$

## 7.3 The Solution Approach

Previous studies (McCulley and Bloebaum, 1996; Ahmadi et al., 2001) have reported that the activity sequencing problem with deterministic activity dependencies is NP-complete. Therefore, it is difficult to solve the problem in (7.8)-(7.12). For small activity sequencing problem, we may apply exact methods, such as the Branch-and-Bound method (Ahmadi et al., 2001), to find global optimal solution. However, it is known that exact methods are computationally infeasible for large problems. Therefore, some heuristic methods have been developed, such as Genetic Algorithm (Altus et al., 1996; McCulley and Bloebaum, 1996; Whitfield et al., 2003; Meier et al., 2007), Simulated Annealing (Abdelsalam and Bao, 2006), and Evolutionary algorithm Lancaster and Cheng, 2008). While these heuristic methods can obtain reasonably solutions in a relatively short time, generally, they cannot guarantee the optimality of the solution, and may suffer from convergence problems.

As reported by Talbi (2002) and Jourdan et al. (2009), in recent years, there is a growing interest in combining exact and heuristic methods, termed as hybrid optimization approaches, for solving NP problems. Such solution strategies can take advantage of both types of methods, and have been successfully applied to many real-world problems, such as the job-shop scheduling problem, the resource-constrained project scheduling problem, the travelling salesman problem and so on. In the activity sequencing problem, Ahmadi et al. (2001) proposed one such hybrid optimization approach, which was successfully applied to the turbopump development process at Rocketdyne with nearly 350 activities. However, the final activity sequence identified by their approach may be a sequence that is worse than the initial one.

Motivated by these issues, in this section we introduce a new hybrid approach for solving the activity sequencing problem in (7.8)-(7.12). We first introduce following notation:

$S$ Activity set that store the sequence of activities from position 1 to position $n$

$S_0$ Activity set that store the sequence of activities from position 1 to position $M-1$

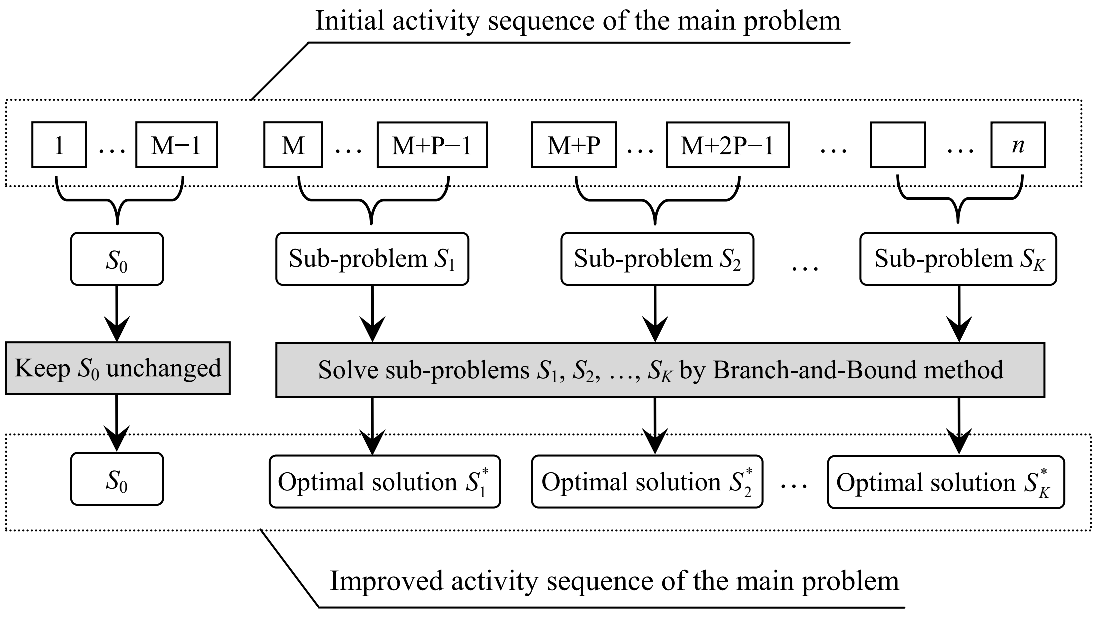
$P$ Number of activities contained in a sub-problem, where $P < n$

$S_k$ Activity set that store the sequence of activities in a sub-problem $k$ ($k = 1,...,K$)

$S_k^*$ Activity set that stores the optimal activity sequence of the sub-problems

$L$ The iteration step length, where $L$ is a positive integer, and $L < P$

$I_{\max}$ Maximum number of iterations



Initial activity sequence of the main problem
1
M−1
M
M+P−1
M+P
M+2P−1
n
S0
Sub-problem S1
Sub-problem S2
Sub-problem SK
Keep S0 unchanged
Solve sub-problems S1, S2, …, SK by Branch-and-Bound method
S0
Optimal solution S1*
Optimal solution S2*
Optimal solution SK*
Improved activity sequence of the main problem

**Figure 7.3 An iteration of the proposed approach**

As illustrated in Figure 7.3, an iteration of the proposed approach consists of three steps: (1) based on an initial activity sequence, select the activities from position $M$ to position $n$, and separate them into several sub-problems such that each sub-problem includes a number of $P$ consecutive activities; (2) solve the sub-problems by the Branch-and-Bound method (note that here the sub-problems are solved in parallel); (3) combine the sub-problem solutions into a solution of the main problem.

After an iteration of above three steps, we can get an improved activity sequence of the main problem. We then use this improved activity sequence to restart, and iteratively change the value of *M* such that new sub-problems, which are different from those of the previous iteration, can be derived. The process is repeated until the maximum number of iterations is reached. We now describe the hybrid approach in details as follows.

**Procedure 7.1**

*Step 0*. Initialize *S*, calculate the corresponding objective value and save the result as $F_{\min}$. Initialize *P* and *L*. Set $M = 1$, $I = 1$.

While $I \le I_{\max}$, do:

*Step 1*. If $M \ge P$, reset $M = 1$.

*Step 2*. Based on *S*, select activities from position 1 to position $(M-1)$ to form $S_0$; choose activities from position $M$ to position $(M+P-1)$ to form sub-problem $S_1$, and activities from position $(M+P)$ to position $(M+2P-1)$ to form sub-problem $S_2$, repeat until the last activity *n* is reached.

*Step 3*. Solve the sub-problems $S_k$ independently by the Branch-and-Bound method, and save the corresponding results as $S_k^*$.

*Step 4*. Combine $S_0$ and optimal sub-problem solutions $S_k^*$, save the result as *S*. Update $F_{\min}$.

*Step 5*. $M = M + L$, $I = I + 1$.

In the above procedure, parameter *P* defines the size of sub-problems. As discussed in Chapter 5 and Chapter 6, two extreme cases need to be avoided in choosing the size of sub-problems: (1) if *P* is set to be too small, then solving the sub-problems by commercial solvers may not improve the solution of the main problem; (2) if *P* is too large, then it will be computationally infeasible. After an iteration of Step 2 to Step 4, we use the improved schedule *S* to restart Step 1. To get new sub-problems that are different from those of the previous iteration, we iteratively change the value of *M* and *L*. Parameter $I_{\max}$ determines the total number of iterations. Generally, the higher the value of $I_{\max}$, the better the final solution. However, the calculation time also increases linearly with the value of $I_{\max}$.

## 7.4 Case Study

| Activities | $A_1$ | $A_2$ | $A_3$ | $A_4$ | $A_5$ | $A_6$ | $A_7$ | $A_8$ | $A_9$ | $A_{10}$ | $A_{11}$ | $A_{12}$ | $A_{13}$ | $A_{14}$ | $A_{15}$ | $A_{16}$ | $A_{17}$ | $A_{18}$ | $A_{19}$ | $A_{20}$ | $A_{21}$ | $A_{22}$ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Engine Block $A_1$ | ■ | H. | L. | H. | L. | H. | H. | H. | H. | L. |  | H. |  |  |  | L. | L. |  | H. |  | H. | M. |
| Crankshaft $A_2$ | H. | ■ | H. | H. | H. | H. | L. | L. | L. |  |  | M. |  |  |  |  |  |  | M. | L. |  | M. |
| Flywheel $A_3$ | L. | H. | ■ |  |  | L. |  |  |  |  |  |  |  |  |  |  |  |  |  |  | M. | M. |
| Pistons $A_4$ | H. | M. | L. | ■ | M. | M. | H. | M. | M. | L. | M. |  |  |  |  |  |  |  | L. |  |  | H. |
| Connecting Rods $A_5$ | M. | H. |  | H. | ■ | M. | L. |  | L. |  |  |  |  |  |  |  |  |  |  |  |  | M. |
| Lubrication $A_6$ | H. | M. | L. | M. | L. | ■ | L. | M. | L. |  |  | L. |  |  |  | L. |  |  | L. |  | M. | H. |
| Cylinder Heads $A_7$ | H. | L. |  | M. |  | L. | ■ | H. | H. | H. |  | H. |  |  |  |  |  |  | M. |  | L. | H. |
| Camshaft/Valve Train $A_8$ | H. | L. |  | L. |  | M. | H. | ■ | L. | L. |  |  |  |  |  |  |  |  | L. |  | L. | M. |
| Water Pump/Cooling $A_9$ | H. |  |  | M. |  | M. | H. | M. | ■ | M. | L. | H. | L. |  | L. |  | L. |  |  |  | L. | M. |
| Intake Manifold $A_{10}$ | M. |  |  |  |  | L. | H. | L. | H. | ■ |  | H. |  |  |  |  |  |  | M. | M. |  | H. |
| Fuel System $A_{11}$ |  |  |  |  |  |  |  |  | L. |  | ■ | M. | L. | L. | L. |  | L. | L. |  |  | M. | M. |
| Accessory Drive $A_{12}$ | H. | M. |  |  |  | L. | H. | L. | H. | H. | H. | ■ | M. | H. | H. | M. | M. | L. | L. | L. | L. | M. |
| Air Cleaner $A_{13}$ |  |  |  |  |  |  |  |  |  |  | L. | H. | ■ | M. | H. | L. |  |  |  |  |  |  |
| AIR $A_{14}$ | L. |  |  |  |  |  |  |  | L. |  |  | H. | M. | ■ | L. | H. | L. |  |  | M. | L. | M. |
| Throttle Body $A_{15}$ |  |  |  |  |  |  |  |  | M. |  | M. | M. | H. | L. | ■ |  | M. | H. | L. | H. |  | M. |
| Exhaust $A_{16}$ | L. |  |  |  |  | L. |  |  | L. |  | M. | L. | L. | H. |  | ■ | H. |  | M. | M. | L. | M. |
| EGR $A_{17}$ | L. |  |  |  |  |  |  |  | L. |  | L. | L. |  | L. | L. | H. | ■ |  | L. | H. | L. | M. |
| EVAP $A_{18}$ |  |  |  |  |  |  |  |  |  |  | M. |  |  |  | H. |  |  | ■ |  | M. | L. |  |
| Ignition $A_{19}$ | H. | H. | H. | L. |  | L. | H. | H. | L. | M. | H. | M. |  |  | L. | H. | L. |  | ■ | H. | H. | H. |
| ECM $A_{20}$ | L. | M. | L. |  |  | L. | L. | L. | H. | M. | H. | L. |  | M. | M. | L. | M. | H. | H. | ■ | H. | M. |
| Electrical System $A_{21}$ | H. | L. | M. | L. |  | M. | L. | M. | L. | L. | M. | L. |  | L. |  | H. | L. | L. | H. | H. | ■ | H. |
| Engine Assembly $A_{22}$ | H. | H. | M. | H. | M. | H. | H. | M. | M. | H. | H. | M. |  | M. | M. | M. | M. | L. | H. | M. | H. | ■ |

**Figure 7.4 Original DSM for the powertrain development (from Eppinger, 2001)**

In order to demonstrate the utility of the proposed methodology, we apply it to a real DSM of the powertrain development at General Motors (Eppinger, 2001). Figure

7.4 shows the original DSM, which consists of 22 activities, numbered from $A_1$ to $A_{22}$. Here "L.", "M." and "H." denote the "low", "medium" and "high" activity dependencies, respectively.

### 7.4.1 Parameter Setting

The imprecise activity dependencies in Figure 7.4 are represented by fuzzy triangular numbers, which are defined as follows:

Low: (0, 0.2, 0.4).

Medium: (0.3, 0.5, 0.7).

High: (0.6, 0.8, 1).

To apply Procedure 7.1, we first set the parameter values. As an example, the number of activities contained in a sub-problem $P$ is fixed at 7, the iteration step length $L$ is set to be 2, and the maximum number of iterations $I_{\max}$ is set to be 6. Based on the parameter values, the solution strategy for the powertrain development is illustrated in Figure 7.5.

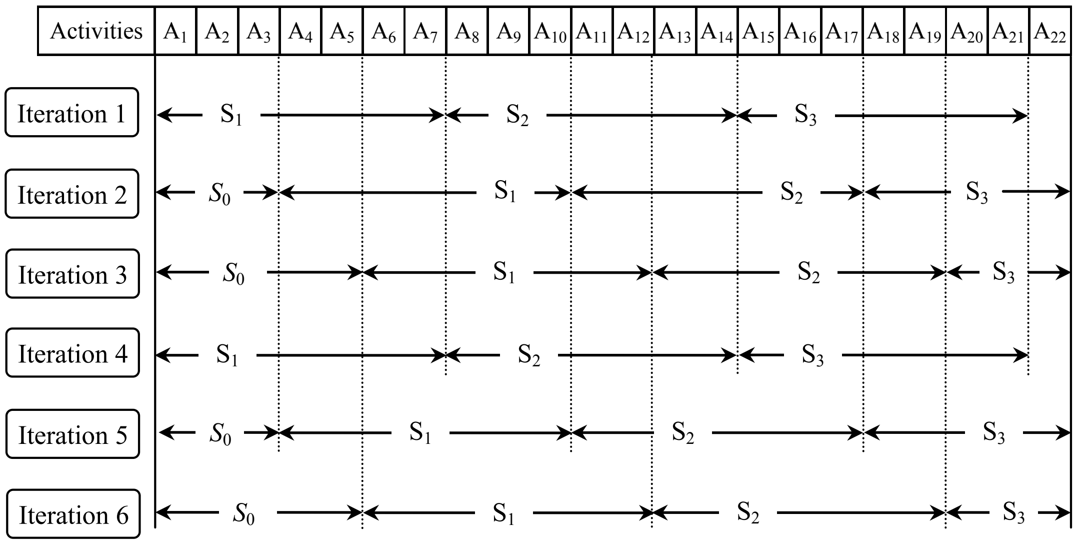

Activities
A1 A2 A3 A4 A5 A6 A7 A8 A9 A10 A11 A12 A13 A14 A15 A16 A17 A18 A19 A20 A21 A22
Iteration 1: S1, S2, S3
Iteration 2: S0, S1, S2, S3
Iteration 3: S0, S1, S2, S3
Iteration 4: S1, S2, S3
Iteration 5: S0, S1, S2, S3
Iteration 6: S0, S1, S2, S3

**Figure 7.5 The solution strategy for the powertrain development**

### 7.4.2 Application Result

Figure 7.6 shows the improved solution by applying Procedure 7.1, where the total fuzzy feedbacks is (30, 53.4, 76.8). Based on the criterion (7.4), the total feedbacks are reduced by about 10%.

| Activities | | $A_{11}$ | $A_{18}$ | $A_6$ | $A_7$ | $A_8$ | $A_1$ | $A_3$ | $A_9$ | $A_{10}$ | $A_{14}$ | $A_4$ | $A_2$ | $A_{20}$ | $A_{17}$ | $A_{16}$ | $A_{15}$ | $A_{12}$ | $A_5$ | $A_{19}$ | $A_{21}$ | $A_{22}$ | $A_{13}$ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Fuel System | $A_{11}$ | ■ | L. | | | | | | L. | | L. | | | | L. | | L. | M. | | | M. | M. | L. |
| EVAP | $A_{18}$ | M. | ■ | | | | | | | | | | | M. | | | H. | | | | L. | | |
| Lubrication | $A_6$ | | | ■ | L. | M. | H. | L. | L. | | | M. | M. | | | L. | | L. | L. | L. | M. | H. | |
| Cylinder Heads | $A_7$ | | | L. | ■ | H. | H. | | H. | H. | | M. | L. | | | | | H. | | M. | L. | H. | |
| Camshaft/Valve Train | $A_8$ | | | M. | H. | ■ | H. | | L. | L. | | L. | L. | | | | | | | L. | L. | M. | |
| Engine Block | $A_1$ | | | H. | H. | H. | ■ | L. | H. | L. | | H. | H. | | L. | L. | | H. | L. | H. | H. | M. | |
| Flywheel | $A_3$ | | | L. | | | L. | ■ | | | | | H. | | | | | | | | M. | M. | |
| Water Pump/Cooling | $A_9$ | L. | | M. | H. | M. | H. | | ■ | M. | | M. | | | L. | | L. | H. | | | L. | M. | L. |
| Intake Manifold | $A_{10}$ | | | L. | H. | L. | M. | | H. | ■ | | | | M. | | | | H. | | M. | | H. | |
| AIR | $A_{14}$ | | | | | | L. | | L. | | ■ | | | M. | L. | H. | L. | H. | | | L. | M. | M. |
| Pistons | $A_4$ | M. | | M. | H. | M. | H. | L. | M. | L. | | ■ | M. | | | | | | M. | L. | | H. | |
| Crankshaft | $A_2$ | | | H. | L. | L. | H. | H. | L. | | | H. | ■ | L. | | | | M. | H. | M. | | M. | |
| ECM | $A_{20}$ | H. | H. | L. | L. | L. | L. | L. | H. | M. | M. | | M. | ■ | M. | L. | M. | L. | | H. | H. | M. | |
| EGR | $A_{17}$ | L. | | | | | L. | | L. | | L. | | | H. | ■ | H. | L. | L. | | L. | L. | M. | |
| Exhaust | $A_{16}$ | M. | | L. | | | L. | | L. | | H. | | | M. | H. | ■ | | L. | | M. | L. | M. | L. |
| Throttle Body | $A_{15}$ | M. | H. | | | | | | M. | | L. | | | H. | M. | | ■ | M. | | L. | | M. | H. |
| Accessory Drive | $A_{12}$ | H. | L. | L. | H. | L. | H. | | H. | H. | H. | | M. | L. | M. | M. | H. | ■ | | L. | L. | M. | M. |
| Connecting Rods | $A_5$ | | | M. | L. | | M. | | L. | | | H. | H. | | | | | | ■ | | | M. | |
| Ignition | $A_{19}$ | H. | | L. | H. | H. | H. | H. | L. | M. | | L. | H. | H. | L. | H. | L. | M. | | ■ | H. | H. | |
| Electrical System | $A_{21}$ | M. | L. | M. | L. | M. | H. | M. | L. | L. | L. | L. | L. | H. | L. | H. | | L. | | H. | ■ | H. | |
| Engine Assembly | $A_{22}$ | H. | L. | H. | H. | M. | H. | M. | M. | H. | M. | H. | H. | M. | M. | M. | M. | M. | M. | H. | H. | ■ | |
| Air Cleaner | $A_{13}$ | L. | | | | | | | | | M. | | | | | L. | H. | H. | | | | | ■ |

**Figure 7.6 Solution for the powertrain development by our approach**

## 7.5 Conclusion

We have presented a mathematical model for activity sequencing problem with a fuzzy representation of activity dependencies. The model can be used to decide an optimal activity sequence with minimum feedbacks, as well as to predict the most likely, pessimistic and optimistic values of feedbacks. We have also developed a new hybrid approach, which embed an exact algorithm within a framework of local search heuristic, for solving large-scale problems. The proposed approach has been applied to a real data set published in Eppinger (2001). Application result shows that the approach can help decision makers better manage product development processes with uncertain activity dependencies.

Future study should consider testing of the proposed scheduling methodology on additional project management problems. These problem sets could include activity dependencies represented by fuzzy trapezoidal numbers, since these are sometimes used in practice to represent fuzzy numbers. We may also extend the proposed scheduling methodology to handle multiple development projects or explore some other approaches for addressing the uncertainties.

# CHAPTER 8

# CONCLUSIONS AND FUTURE STUDY

This thesis contributes to some methodological issues for scheduling tests in overlapped NPD process and for sequencing design activities with iteration loops. In this chapter, we will summarize the main results of our study as described in previous chapters, and discuss some possible future research.

## 8.1 Summary of Results

### 8.1.1 Optimal Scheduling of Tests in Overlapped NPD Process

To model and structure NPD processes, an important decision problem is how to optimally schedule various tests along the NPD process so as to maximize the product development performance. Past studies, which are developed to determine the optimal scheduling of tests, often focused on single-stage testing of sequential design process. Meanwhile, overlapping has become a common mode of product development (Terwiesch et al., 2002; Yassine et al., 2008; Roemer and Ahmadi, 2010). We therefore present two analytical models for the optimal scheduling of tests in overlapped NPD process.

When the testing set-up time is relatively small, the analytical model in Chapter 3 can help management decide when to stop testing at each stage, and when to start downstream development (e.g. mold fabrication). Analysis of the model also yields several useful management insights (Proposition 3.1-3.5, Corollaries 3.1-3.3). For instance, Proposition 3.2 reveals that the total cost is first convex then concave increasing with respect to upstream testing duration. In practice, some companies may intend to shift from sequential to overlapped process. Then, Proposition 3.4 can be used as a guideline for these companies to adjust their existing testing and overlapping policies coordinately. Proposition 3.5 indicates that our methodology can also be used to derive optimal testing and overlapping strategies that achieve the required product quality at minimum cost, or optimal testing and overlapping strategies whose product quality is best for a given budget. We have also applied the methodology to the handset design process at our case study company.

When the testing set-up time is long, the analytical model in Chapter 4 can help decision makers determine the optimal number of tests needed at each stage, together with the optimal overlapping policies. The work in this chapter yields several useful results (Proposition 4.1-4.4). For instance, Proposition 4.1 reveals that it is optimal to start downstream stages directly after the completion of an upstream test, which is useful to guide the practice. Proposition 4.2 and 4.3 discuss the impact of different model parameters, such as the testing set-up time and unit problem-solving time, on the optimal solution, which can help the management adjust testing and overlapping strategies for NPD projects with different characteristics. Finally, our model was applied to improve the refrigerator development process at our case study company.

### 8.1.2 Approaches for DSM Sequencing Problem

Generally, in test scheduling models, the NPD process is viewed as consisting of a series of development stages. In such cases, it’s no need to consider the sequencing problem since the execution sequence of these stages (such as the concept design and the detail design) is known. However, when the NPD process is further broken into a number of smaller activities, then, a key and challenging issue, i.e. the planned timing and sequence of activities, arises because clear precedence constraints among design activities do not exist and are rarely known in advance (Eppinger et al., 1994; Ahmadi et al., 2001; Jun and Suh, 2008; Karniel and Reich, 2009).

Unlike the manufacturing process, the NPD process often involves a number of activities with cyclic information flow among them (Cho and Eppinger, 2005; Karniel and Reich, 2009). Traditional network-based scheduling techniques, such as CPM and PERT, cannot effectively model cyclic information and iteration, limiting their capability of planning for NPD projects (Eppinger et al. 1994; Ahmadi et al., 2001; Karniel and Reich, 2009). To address this shortfall, one known method is DSM, which has spawned many research efforts on sequencing design activities with the objective of minimizing feedbacks. It is known that the problem is NP-complete, and thus is difficult to solve. While previous decomposition method provides an efficient way to solve large-scale problems, a major disadvantage of these methods is that the resulting overall activity sequence may be a sequence that is worse than the initial one. To address this issue, we therefore propose two new decomposition approaches for solving large problem.

In Chapter 5, we first propose two simple rules (Theorem 5.1 and 5.2) for feedback reduction. A heuristic is then proposed for improving feasible solutions through activity exchange. After that, a new decomposition approach is presented for solving large DSM sequencing problem. The proposed decomposition approach outperforms previous ones in its flexibility and well construction of sub-problems. We have also applied the proposed solution strategy to three real data sets. Compared to the solutions presented in previous studies, applying our approach results in better solutions with smaller feedbacks.

In Chapter 6, we further establish rules of block-activity exchange (Theorem 6.1 and 6.2) and block-block exchange (Theorem 6.3 and 6.4), for feedback reduction. We find that based on the fold operation, a block has similar properties to a single activity. Based on these findings, a novel decomposition approach is presented for large DSM sequencing problems. We also perform six groups of random examples and show that good solutions can be easily obtained by Procedure 6.2. Compared to the decomposition approach presented in Chapter 5, one advantage of the approach in this chapter is that it can solve the sub-problems independently and in parallel.

Finally, in some situations, activity dependencies may not be precisely estimated. In Chapter 7, we resort to fuzzy set theory for tackling uncertain parameters, and present a fuzzy approach to DSM sequencing problem. The model can be utilized to predict the most likely, pessimistic and optimistic values of feedbacks, and can help managers better manage NPD processes with uncertainty. To demonstrate its utility, the methodology has been applied to the prowerchain development at General Motors (Eppinger, 2001).

## 8.2 Possible Future Research

In spite of the contributions described in previous section, we would like to point out some limitations for application of the results in this thesis. Future research may be carried out in the areas discussed below.

In Chapter 3 and Chapter 4, we present two analytical models for the optimal scheduling of tests in overlapped development processes. As mentioned earlier, the work in Chapter 4 complements the work in Chapter 3 by taking testing set-up time into consideration, and by examining the testing and overlapping polices for the whole NPD process rather than two development stages. In the future, we may further extend the work in Chapter 4 as follows.

Firstly, our models can only be built and analyzed when historical data is available. For derivative projects which account for the majority of product development projects (Leifer et al., 2000; Rizova, 2006; Jun and Suh, 2008), these data are typically available (see e.g. Krishnan et al., 1997; Thomke and Fujimoto, 2000; Gerk and Qassim, 2008; Ni et al., 2008; Love et al., 2009). However, for totally new projects, we can only build a model based on estimated data which may lead to inaccurate results. Secondly, in spite of our best effort, we were able to do only a small number of case studies. In the data collection process, we find that many firms are unwilling to share their product development experiences for security reasons. Even for the two firms that have participated in our study, the data collection was quite challenging and time-consuming since the companies did not know how past project data could be used in guiding future projects. In the future, it is meaningful to test the broader application of the models by applying them to other development projects and processes. Thirdly, the present models are deterministic and therefore do not directly address risk, although we have addressed many facets of risk through sensitivity analysis of the optimal solution to parameters. Consequently, we may extend our models by taking into account the estimation errors of model parameters into decision making. Fourthly, we assume the opportunity cost of time is constant. However, although the opportunity cost should be constant in short run, it may change in long run. Hence, it may be worthwhile to investigate in detail how will the opportunity cost of time affect the product development performance, as well as the optimal testing and overlapping polices. Finally, to keep the model simple, we assume that the overall penalty cost is a linear function of the number of remaining design faults, and overall rework cost is a linear function of the amount of rework. Even with such linear assumptions, our model is complex and is difficult to solve. Moreover, the linear functions are appropriate approximations of many real situations, as demonstrated by many studies (see e.g. Kogan and Raz, 2002; Roemer and Ahmadi, 2004; Ji et al., 2005; Wang and Yan, 2005; Gerk and Qassim, 2008; Lin et al., 2009). Future research can relax these assumptions and discuss the corresponding testing and overlapping policies.

In Chapter 5 and Chapter 6, we propose several simple rules for reducing feedbacks and present two decomposition-based approaches for solving DSM sequencing problem. There are two major limitations for the application of the methodologies developed in these two chapters. Firstly, the present studies only examine the activity sequencing problem with the objective of minimizing feedbacks, because it is simple and of practical importance. Moreover, it is also a common practice in previous activity sequencing models. As has been mentioned earlier, except for the objective of minimizing feedbacks, some other considerations have also been incorporated in sequencing design activities in a DSM. Therefore, our models may be extended to explore the activity sequencing problem with some other objectives. Second, the studies presented in Chapter 5 and 6 of this thesis, assume that the information dependencies among activities can be estimated with reasonable accuracy. This assumption has been proved to hold in dozens of development projects, see e.g. Eppinger et al. (1994), Eppinger (2001), Browning and Eppinger (2002), Batallas and Yassine (2006), Yu et al. (2007), Sosa (2008), Tang et al. (2009), Collins et al. (2009), Voss and Hsuan (2009). However, there are still some projects where precise information of activity dependencies is not available (Chen et al., 2004; Karniel and Reich, 2009; Luh et al., 2009). Hence, the present studies may not be applicable for these projects.

In Chapter 7, we propose a fuzzy approach to incorporate imprecise parameters, and present a scheduling methodology for activity sequencing problem with a fuzzy representation of activity dependencies. Future study should consider testing of the proposed scheduling methodology on additional project management problems. These problem sets could include activity dependencies represented by fuzzy trapezoidal numbers, since these are sometimes used in practice to represent fuzzy numbers. Moreover, the present model considers only one project within a company. However, spending resource for testing and downstream rework for one project inevitably causes delay in other projects. Therefore, we may also extend the proposed model to handle multiple development projects since it has become increasingly important (Roemer and Ahmadi, 2010; Song and Kusiak, 2010), or explore some other approaches for addressing the uncertainties.

# BIBLIOGRAPHY

Abdelsalam, H.M.E., Bao, H.P., 2006. A simulation-based optimization framework for product development cycle time reduction. IEEE Transactions on Engineering Management 53 (1), 69-85.

Abdelsalam, H.M.E., Bao, H.P., 2007. Re-sequencing of design processes with activity stochastic time and cost: An optimization-simulation approach. Journal of Mechanical Design 129 (2), 150-157.

Ahmadi, R., Roemer, T.A., Wang, R.H., 2001. Structuring product development processes. European Journal of Operational Research 130 (3), 539-558.

Ahmadi, R., Wang, H., 1999. Managing development risk in product design processes. Operations Research 47 (2), 235-246.

Altus, S.S., Kroo, I.M., Gage, P.J., 1996. A genetic algorithm for scheduling and decomposition of multidisciplinary design problems. Journal of Mechanical Design 118 (4), 486–489.

Amrit, C., van Hillegersberg, J., 2008. Detecting coordination problems in collaborative software development environments. Information Systems Management 25 (1), 57-70.

Anderson, E.G., Joglekar, N.R., 2005. A hierarchical product development planning framework. Production and Operations Management 14(3), 344-361.

Atuahene-Gima, K., Evangelista, F., 2000, Cross-Functional influence in new product development: An exploratory study of marketing and R&D perspectives. Management Science 46 (10), 1269-1284.

Badiru, A.B, 1993. Quantitative Models for Project Planning, Scheduling, and Control. Quorum Books, Westport, Conn.

Baldwin, C.Y., Clark, K.B., 2000. Design rules: Volume 1, the power of modularity. MIT Press, Cambridge, Massachusetts.

Banerjee, A., Carrillo, J., Paul, A., 2007. Projects with sequential iteration: Models and complexity. IIE Transactions 39 (5), 453-463.

Bartels, J.H., Zimmermann, J., 2009. Scheduling tests in automotive R&D projects. European Journal of Operational Research 193 (3), 805-819.

Bashir, H.A., AlZebdeh, K., Abdo, J., 2009. An Eigenvalue Based Approach for Assessing the Decomposability of Interdependent Design Project Tasks. Concurrent Engineering: Research and Applications 17 (1), 35-42.

Batallas, D.A., Yassine, A.A., 2006. Information leaders in product development organizational networks: Social network analysis of the design structure matrix. IEEE Transactions on Engineering Management 53 (4), 570-582.

Belhe, U., Kusiak, A., 1996. Modeling relationships among design activities. Journal of Mechanical Design 118 (4), 454-460.

Bhaskaran, S.R., Krishnan, V., 2009. Effort, revenue, and cost sharing mechanisms for collaborative new product development. Management Science, 55 (7), 1152-1169.

Bhuiyan, F., 2001. Dynamic models of concurrent engineering processes and performance. Ph.D. dissertation, Department of Mechanical Engineering, McGill University, Montreal.

Bhuiyan, N., Gerwin, D., Thomson, V., 2004. Simulation of the new product development process for performance improvement. Management Science 50 (12), 1690-1703.

Bhuiyan, N., Thomson, V., Gerwin, D., 2006. Implementing concurrent engineering. Research Technology Management 49 (1), 38-43.

Biazzo, S., 2009. Flexibility, structuration, and simultaneity in new product development. Journal of Product Innovation Management 26 (3), 336-353.

Black, L.J., Repenning, N.R., 2001. Why firefighting is never enough: Preserving high-quality product development. System Dynamics Review 17 (1), 33-62.

Boehm, B.W., 1981. Software engineering economics. Englewood Cliffs, Prentice-Hall.

Browning, T.R., 2001. Applying the Design Structure Matrix to system decomposition and integration problems: a review and new directions. IEEE Transactions on Engineering Management 48 (3), 292-306.

Browning, T.R., Eppinger, S.D., 2002. Modeling impacts of process architecture on cost and schedule risk in product development. IEEE Transactions on Engineering Management 49 (4), 428-442.

Browning, T.R., Ramasesh, R. V., 2007. A survey of activity network-based process models for managing product development projects. Production and Operations Management 16 (2), 217-240.

Browning, T.R., 2009. The many views of a process: Toward a process architecture framework for product development processes. Systems Engineering 12 (1), 69-90.

Browning, T.R., Health, R.D., 2009. Reconceptualizing the effects of lean on production costs with evidence from the F-22 program. Journal of Operations Management 27 (1), 23-44.

Burkard, R.E., Dell'Amico, M., Martello, S., 2009. Assignment Problems. Society for Industrial and Applied Mathematics, Philadelphia.

Cantamessa, M., Villa, A., 2000. Product and process design effort allocation in concurrent engineering. International Journal of Production Research 38 (14), 3131-3147.

Carrascosa, M., Eppinger, S.D., Whitney, D.E., 1998. Using the design structure matrix to estimate product development time. ASME Design Engineering Technical Conferences, Atlanta, GA.

Carrillo, J.E., Franza, R.M., 2006. Investing in product development and production capabilities: The crucial linkage between time-to-market and ramp-up time. European Journal of Operational Research 171 (2), 536-556.

Chakravarty, A.K., 2001. Overlapping design and build cycles in product development. European Journal of Operational Research 134 (2), 392-424.

Chakravarty, A.K., 2003. Optimizing prototyping effort with an evolving product design. IIE Transactions 35 (10), 921-932.

Chanas, .S, Dubois, D., Zielinski, P., 2002. On the sure criticality of tasks in activity networks with imprecise durations. IEEE Transactions on Systems Man and Cybernetics 32 (4), 393-407.

Chao, R.O., Kavadias, S., Gaimon, C., 2009. Revenue driven resource allocation: Funding authority, incentives, and new product development portfolio management. Management Science 55 (9), 1556-1569.

Chen, C.H., Khoo, L.P., Jiao, L., 2004. Information deduction approach through quality function deployment for the quantification of the dependency between design tasks. International Journal of Production Research 42 (21), 4623-4637.

Chen, L., Ding, Z.D., Li, S., 2005. A formal two-phase method for decomposition of complex design problems. Journal of Mechanical Design 127 (2), 184-195.

Chen, L., Li, S. 2005. Analysis of decomposability and complexity for design problems in the context of decomposition. Journal of Mechanical Design 127 (4), 545-557.

Chen, L., Macwan, A., Li, S., 2007. Model-based rapid redesign using decomposition patterns.  Journal of Mechanical Design 129 (3), 283-294.

Chen, S.J., Li, L., 2003. Decomposition of interdependent task group for concurrent engineering. Computers and Industrial Engineering 44 (3), 435-459.

Chen, S.J., Huang, E., 2007. A systematic approach for supply chain improvement using design structure matrix. Journal of Intelligent Manufacturing 18 (2), 285-299.

Chin, K.S., Zu, X., Mok, C.K., Tam, H. Y., 2006. Integrated Integration Definition Language 0 (IDEF) and coloured Petri nets (CPN) modelling and simulation tool: a study on mould-making processes. International Journal of Production Research 44 (16), 3179-3205.

Cho, S., Eppinger, S.D., 2005. A simulation-based process model for managing complex design projects. IEEE Transactions on Engineering Management 52 (3), 316-328.

Clark, K.B., Fujimoto, T., 1991. Product Development Performance Strategy, Organization and Management in the World Auto Industry. Harvard Business School Press, Boston, MA.

Clarkson, P.J., Simons, C., Eckert, C., 2004. Predicting change propagation in complex design. Journal of Mechanical Design 126 (5), 788-797.

Cohen, M.A., Eliashberg, J., Ho, T-H., 1996. New product development: the performance and time-to-market tradeoff. Management Science 42 (2), 173-186.

Collins, S.T., Yassine, A.A., Borgatti, S.P., 2009. Evaluating product development systems using network analysis. Systems Engineering 12 (1), 55-68.

Cooper, K.G., 1980. Naval ship production: A claim settled and a framework built. Interface 10 (6), 20-36.

Cooper, K.G., 1993a. The rework cycle: Benchmarks for the project manager. Project Management Journal 24 (1), 17-22.

Cooper, K.G., 1993b. The rework cycle: How projects are mismanaged. PMNETwork, February, 5-7.

Cooper, K.G., 1993c. The rework cycle: How it really works…and reworks…. PMNETwork, February, 25-28.

Cooper, R.G., 1994. Third generation of new product processes. Journal of Production Innovation Management 11 (1), 3-14.

Cooper, R.G., 2001. Winning at new products: Accelerating the process from idea to launch (3rd ed.). Perseus Publishing, Cambridge, Massachusetts.

Cooper, R.G., Kleinschmidt, E.J., 2007. Winning businesses in product development: The critical success factors. Research-Technology Management 50 (3), 52-66.

Cooper, R.G., Edgett, S.J., 2008. Maximizing productivity in product innovation. Research-Technology Management 51 (2), 47-58.

Cusumano, M.A., Selby, R.W., 1995. Microsoft secrets: how the world's most powerful software company creates technology, shapes markets, and manages people. Free Press, New York.

Dahan, E., Srinivasan, V., 2000. The predictive power of internet-based product concept testing using visual depiction and animation. Journal of Product Innovation Management 17 (2), 99-109.

Dahan, E., Mendelson, H., 2001. An extreme-value model of concept testing. Management Science 47 (1), 102-116.

Dahan, E., Hauser, J.R., 2002. The virtual customer. Journal of Product Innovation Management 19 (5), 332-353.

Dai, Y.S., Xie, M., Poh, K.L., Yang, B., 2003. Optimal testing-resource allocation with genetic algorithm for modular software systems. Journal of Systems and Software 66 (1), 47–55.

Datar, S., Jordan, C., Kekre, S., Rajiv, S., Srinivasan, K., 1997. New product development structures and time-to-market. Management Science 43 (4), 452-464.

De Weerd-Nederhof, P.C., Wouters, M.J.F., Teuns, S.J.A., Hissel, P.H., 2007. The architecture improvement method: cost management and systemic learning about strategic product architectures. R&D Management, 37 (5), 425-439.

Denker, D., Steward, D.V., Browning, T.R, 2001. Planning concurrency and managing iteration in projects. Project Management Journal 32(3), 31-38.

Dubois, D., Fargier, H., Fortemps, P., 2003a. Fuzzy scheduling: Modelling flexible constraints vs. coping with incomplete knowledge. European Journal of Operational Research 147 (2), 231-252.

Dubois, D., Fargier, H., Galvagnon, V., 2003b. On latest starting times and floats in activity networks with ill-known durations. European Journal of Operational Research 147 (2), 266-280.

Eisenhardt, K.M., Tabrizi, B.N., 1995. Accelerating adaptive processes: Product innovation in the global computer industry. Administrative Science Quarterly 40 (1), 84-110.

Eppinger, S.D., Whitney, D.E., Smith, R.P., Gebala, D.A., 1994. A model-based method for organizing tasks in product development. Research in Engineering Design 6 (1), 1-13.

Eppinger, S.D., 2001. Innovation at the speed of information. Harvard Business Review 79 (1), 149-158.

Erat, S., Kavadias, S., 2008. Sequential testing of product designs: Implications for learning. Management Science 54 (5), 956-968.

Fixson, S.K., 2005. Product architecture assessment: a tool to link product, process, and supply chain design decisions. Journal of Operations Management 23 (3-4), 345-369.

Ford, D.N., Sterman, J.D., 1998. Dynamic modeling of product development processes. System Dynamics Review 14 (1), 31-68.

Ford, D.N., Sterman, J.D., 2003a. Overcoming the 90% syndrome: Iteration management in concurrent development projects. Concurrent Engineering: Research and Applications 11 (3), 177-186.

Ford, D.N., Sterman, J.D., 2003b. The liar's club: Concealing rework in concurrent development. Concurrent Engineering: Research and Applications 11 (3), 211-219.

Fortemps, P., Roubens, M., 1996. Ranking and defuzzification methods based on area compensation. Fuzzy Sets and Systems 82 (3), 319-330.

Fulkerson, D.R., 1961. A network flow computation for project cost curves. Management Science 7 (2), 167-178.

Gebala, D.A., Eppinger, S.D., 1991. Methods for analyzing design procedures, Proceedings of ASME 3rd International Conference on Design Theory and Methodology, 227-233.

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

Gerwin, D., Barrowman, N.J., 2002. An evaluation of research on integrated product development. Management Science 48 (7), 938-953.

Gil, N., Tommelein, I.D., Beckman, S., 2004. Postponing design processes in unpredictable environments. Research in Engineering Design 15 (3), 139–154.

Gil, N., Beckman, S., Tommelein, I.D., 2008. Upstream problem solving under uncertainty and ambiguity: Evidence from airport expansion projects. IEEE Transactions on Engineering Management 55 (3), 508-522.

Gil, N., 2009. Project Safeguards: Operationalizing Option-Like Strategic Thinking in Infrastructure Development. IEEE Transactions on Engineering Management, 56 (2), 257-270.

Golenko-Ginzburg, D., Gonik, A., 1996. On-line control model for cost simulation network projects. Journal of the Operational Research Society 47 (2), 266-283.

Grupp, H., Maital, S., 2001. Managing New Product Development and Innovation: A Microeconomic Toolbox. Edward Elgar, Cheltenham, UK.

Ha, A.Y., Porteus, E.L., 1995. Optimal timing of reviews in concurrent design for manufacturability. Management Science 41 (9), 1431-1447.

Haberle, K.R., Burke, R.J., Graves, R.J., 2000. A note on measuring parallelism in concurrent engineering. International Journal of Production Research 38 (8), 1947-1952.

Harter, D.E., Krishnan M.S., Slaughter, S.A., 2000. Effects of process maturity on quality, cycle time, and effort in software product development. Management Science 46 (4), 451-466.

Harter, D.E., Slaughter, S.A., 2003. Quality improvement and infrastructure activity costs in software development: A longitudinal analysis. Management Science 49(6), 784-800.

Hauser, J., Tellis, G.J., Griffin A., 2006. Research on innovation: A review and agenda for marketing science. Marketing Science 25 (6), 687-717.

He, Z.W., Xu, Y., 2008. Multi-mode project payment scheduling problems with bonus-penalty structure. European Journal of Operational Research 189 (3), 1191-1207.

Helms, R., 2002. Product data management as enabler for concurrent engineering. Ph.D. dissertation, Department of Technology Management, Technical University Eindhoven, The Netherlands.

Helms, R., 2004. Framework for releasing preliminary information in product development. Advanced Engineering Informatics 18 (4), 231-240.

Helo, P.T., 2006. Product configuration analysis with design structure matrix. Industrial Management and Data Systems 106 (7), 997-1011.

Hillier, F.S., Lieberman, G.J., 2001. Introduction to operations research (7th ed.). McGraw-Hill, Boston.

Hou, R.H., Kuo, S.Y., Chang, Y.P., 1997. Optimal release times for software systems with scheduled delivery time based on the HGDM. IEEE Transactions on Computers 46 (2), 216-221.

Huang, C.Y., Lyu, M.R., 2005. Optimal release time for software systems considering cost, testing-effort, and test efficiency. IEEE Transactions on Reliability 54 (4), 583-591.

Ji, Y.H., Mookerjee, V.S., Sethi, S.P., 2005. Optimal software development: A control theoretic approach. Information Systems Research 16(3), 292-306.

Joglekar, N.R., Yassine, A.A., Eppinger, S.D., Whitney, D. E., 2001. Performance of coupled product development activities with a deadline. Management Science 47 (12), 1605-1620.

Joglekar, N.R., Ford, D.N., 2005. Product development resource allocation with foresight. European Journal of Operational Research 160 (1), 72-87.

Jourdan, L., Basseur, M., Talbi, E.G., 2009. Hybridizing exact methods and metaheuristics: A taxonomy. European Journal of Operational Research 199 (3), 620-629.

Jun, H.B., Ahn, H.S., Suh, H.W., 2005. On identifying and estimating the cycle time of product development process. IEEE Transactions on Engineering Management 52 (3), 336-349.

Jun, H.B., Suh, H.W., 2008. A modeling framework for product development process considering its characteristics. IEEE Transactions on Engineering Management 55(1), 103-119.

Kang, C., Hong, Y.S., 2009. Evaluation of acceleration effect of dynamic sequencing of design process in a multiproject environment. Journal of Mechanical Design 131(2), article number: 021008.

Kapur, P.K., Bardhan, A.K., Yadavalli, V.S.S., 2007. On allocation of resources during testing phase of a modular software. International Journal of Systems Science 38 (6), 493-499.

Karimian, P., Herrmann, J.W., 2009. Separating design optimization problems into decision-based design processes. Journal of Mechanical Design 131 (1), Article Number: 011007.

Karniel, A., Reich, Y. 2009. From DSM-based planning to design process simulation: A review of process scheme logic verification issues. IEEE Transactions on Engineering Management, 56 (4), 636-649.

Ke, H., Liu, B.D., 2007. Project scheduling problem with mixed uncertainty of randomness and fuzziness. European Journal of Operational Research 183 (1), 135-147.

Kelley, J.E., 1961. Critical-path planning and scheduling: mathematical basis. Operations Research 9, 296-320.

Kogan, K., Raz, T., 2002. Optimal allocation of inspection effort over a finite planning horizon. IIE Transactions 34 (6), 515-527.

Kolisch, R., Hartmann, S., 2006. Experimental investigation of heuristics for resource-constrained project scheduling: An update. European Journal of Operational Research 174 (1), 23-37.

Kouskouras, K.G., Georgiou, A.C., 2007. A discrete event simulation model in the case of managing a software project. European Journal of Operational Research 181(1), 374-389.

Krishnan, V., 1996. Managing the simultaneous execution of coupled phases in concurrent product development. IEEE Transactions on Engineering Management 43 (2), 210-217.

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

Krishnan, V., Ulrich, K.T., 2001. Product development decisions: A review of the literature. Management Science 47 (1), 1-21.

Kusiak, A., and Wang, J., 1993. Efficient organizing of design activities. International Journal of Production Research 31(4), 753-769.

Kusiak, A., Larson, T.N., Wang, J.R., 1994. Reengineering of design and manufacturing processes. Computers and Industrial Engineering 26 (3), 521-536.

Kusiak, A., Wang, J., He, D.W., Feng, C.X., 1995. A structured approach for analysis of design processes. IEEE Transactions on Components, Packaging and Manufacturing technology-Part A 18 (3), 664-673.

Lancaster, J., Ozbayrak, M., 2007. Evolutionary algorithms applied to project scheduling problems - a survey of the state-of-the-art. International Journal of Production Research 45 (2), 425-450.

Lancaster, J., Cheng, K., 2008. A fitness differential adaptive parameter controlled evolutionary algorithm with application to the design structure matrix. International Journal of Production Research 46 (18), 5043-5057.

Lee, H., Suh, H.W., 2006. Workflow structuring and reengineering method for design process. Computers and Industrial Engineering 51 (4), 698-714.

Leifer, R., McDermott, C.M., O'Connor, G.C., Peters, L.S., Rice, M., Veryzer, R.W., 2000. Radical Innovation: How Mature Companies Can Outsmart Upstarts. Harvard Business School Press, Cambridge, Massachusetts.

Levardy, V., Browning, T.R., 2009. An adaptive process model to support product development project management. IEEE Transactions on Engineering Management, 56 (4), 600-620.

Li, D., Sun, X.L., 2006. Nonlinear integer programming. Springer, New York.

Liberatore, M.J., 2008. Critical path analysis with fuzzy activity times. IEEE Transactions on Engineering Management 55 (2), 329-337.

Liker, J.K., Sobek, D.K., Ward, A.C., Cristiano, J.J., 1996. Involving suppliers in product development in the United States and Japan: evidence for set-based concurrent engineering. IEEE Transactions on Engineering Management 43 (2), 165-178.

Lin, J., Chai, K.H., Wong, Y.S., Brombacher, A.C., 2008. A dynamic model for managing overlapped iterative product development. European Journal of Operational Research 185 (1), 378-392.

Lin, J., Chai, K.H., Brombacher, A.C., Wong, Y.S., 2009. Optimal overlapping and functional interaction in product development. European Journal of Operational Research 196 (3), 1158-1169.

Lin, J., Qian, Y.J., Cui, W.T., Miao, Z.L., 2010. Overlapping and communication policies in product development. European Journal of Operational Research 201 (3), 737-750.

Loch, C.H., Terwiesch, C., 1998. Communication and uncertainty in concurrent engineering. Management Science 44 (8), 1032-1048.

Loch, C.H., Terwiesch, C., Thomke, S., 2001. Parallel and sequential testing of design alternatives. Management Science 47 (5), 663-678.

Loch, C., Mihm, J., Huchzermeier, A., 2003. Concurrent engineering and design oscillations in complex engineering projects. Concurrent Engineering: Research and Applications 11 (3), 187-199.

Loch, C.H., Terwiesch, C., 2005. Rush and be wrong or wait and be late? A model of information in collaborative processes. Production and Operations Management 14(3), 331-343.

Lootsma, F.A., 1989. Stochastic and fuzzy PERT. European Journal of Operational Research 43 (2) 174-183.

Lopes, C.V., Bajracharya, S.K., 2006. Assessing aspect modularizations using design structure matrix and net option value. Transactions on Aspect-Oriented Software Development I, 3880, 1-35.

Love, P.E.D., Edwards, D.J., Irani, Z., 2008. Forensic project management: An exploratory examination of the causal behavior of design-induced rework. IEEE Transactions on Engineering Management 55 (2), 234-247.

Love, P.E.D., Edwards, D.J., Irani, Z., Walker, D.H.T., 2009. Project pathogens: the anatomy of omission errors in construction and resource engineering project. IEEE Transactions on Engineering Management 56 (3), 425-435.

Luh, D.B., Ko, Y.T., Ma, C.H., 2009. A dynamic planning approach for new product development. Concurrent Engineering: Research and Applications, 17 (1), 43-59.

MacCormack, A., Verganti, R., Iansiti, M., 2001. Developing products on "Internet Time": The anatomy of a flexible development process. Management Science 47(1), 133-150.

MacCormack, A., Rusnak, J., Baldwin, C.Y., 2006. Exploring the structure of complex software designs: An empirical study of open source and proprietary code. Management Science 52 (7), 1015-1030.

Maheswari, J.U., Varghese, K., Sridharan, T., 2006. Application of dependency structure matrix for activity sequencing in concurrent engineering projects. Journal of Construction Engineering and Management 132 (5), 482-490.

McCulley, C., Bloebaum, C.L., 1996. A genetic tool for optimal design sequencing in complex engineering systems. Structural Optimization 12 (2-3), 186-201.

Meier, C., Yassine, A.A., Browning, T.R., 2007. Design process sequencing with competent genetic algorithms. Journal of Mechanical Design 129 (6), 566-585.

Meixell, M.J., Nunez, M., Talalayevsky, A., 2006. Activity structures in a project-based environment: A coordination theory perspective. IEEE Transactions on Engineering Management 53 (2), 285-296.

Mihm, J., Loch, C., Huchzermeier, A. 2003, Problem-Solving oscillations in complex engineering projects. Management Science 46(6), 733-750.

Mika, M., Waligora, G., Weglarz, J., 2005. Simulated annealing and tabu search for multi-mode resource-constrained project scheduling with positive discounted cash flows and different payment models. European Journal of Operational Research, 164(3), 639-668.

Minderhoud, S., Fraser, P., 2005. Shifting paradigms of product development in fast and dynamic markets. Reliability Engineering and System Safety 88 (2), 127-135.

Mitchell, V.L., Nault, B.R., 2007. Cooperative planning, uncertainty, and managerial control in concurrent design. Management Science 53 (3), 375-389.

Morelli, M.D., Eppinger S.D., Gulati R.K., 1995. Predicting technical communications in product development organizations. IEEE Transactions on Engineering Management 42 (3), 215-222.

Muhuri, P.K., Shukla, K.K., 2008. Real-time task scheduling with fuzzy uncertainty in processing times and deadlines. Applied Soft Computing 8 (1), 1-13.

Nair, A., Boulton, W.R., 2008. Innovation-oriented operations strategy typology and stage-based model. International Journal of Operations and Production 28 (7-8), 748-711.

Nambisan, S., 2002. Designing virtual customer environments for new product development: Toward a theory. Academy of Management Review 27 (3), 392-413.

Nasution, S.H., 1994. Fuzzy critical path method. IEEE Transactions on Systems Man and Cybernetics 24 (1), 48-57.

Novak, S., Stern, S., 2008. How does outsourcing affect performance dynamics? Evidence from the automobile industry. Management Science, 54 (12), 1963-1979.

Ni, M., Luh, P.B., Moser, B., 2008. An optimization-based approach for design project scheduling. IEEE Transactions on Automation Science and Engineering 5(3), 3940406.

O'Sullivan, A., 2003. Dispersed collaboration in a multi-firm, multi-team product-development project. Journal of Engineering and Technology Management 20 (1-2), 93-116.

Osborne, S.M., 1993. Product Development Cycle Time Characterization through Modeling of Process Change. Unpublished master's thesis, Sloan School of Management, MIT, Cambridge, Massachusetts.

Patrashkova-Volzdoska, R., McComb, S.A., Green, S.G., Compton, W.D., 2003. Examining a curvilinear relationship between communication frequency and team performance in cross-functional project teams. IEEE Transactions on Engineering Management 50 (3), 262-269.

Pektas, S.T., Pultar, M., 2006. Modelling detailed information flows in building design with the parameter-based design structure matrix. Design Studies 27 (1), 99-122.

Petrovic, S., Fayad, C., Petrovic, D., 2008. Sensitivity analysis of a fuzzy multiobjective scheduling problem. International Journal of Production Research 46 (12), 3327-3344.

Pham, H., Zhang, X.M., 2003. NHPP software reliability and cost models with testing coverage. European Journal of Operational Research 145 (2), 443–454.

Pich, M.T., Loch, C.H., Meyer, A.D., 2002. On uncertainty, ambiguity, and complexity in project management. Management Science 48 (8) 1008-1023.

Pollack-Johnson, B., Liberatore, M.J., 2006. Incorporating quality considerations into project time/cost tradeoff analysis and decision making. IEEE Transactions on Engineering Management 53 (4), 534-542.

Qian, Y.J., Goh, T.N., 2007. Development projects scheduling and Design Structure Matrix. Proceeding of 2007 IEEE International Conference on Industrial Engineering and Engineering Management. Singapore.

Qian, Y.J., Xie, M., Goh, T.N., 2007. A model for streamlining overlapped iterative design processes. Proceeding of the Asia-Pacific Systems Engineering Conference. Singapore.

Qian, Y.J., Xie, M., Goh, T.N., Lin, J., 2010. Optimal testing strategies in overlapped design process. European Journal of Operational Research 206 (1), 131-143.

Rizova, P., 2006. Are you networked for successful innovation? MIT Sloan Management Review 47 (3), 49-55.

Rodrigues, A., Bowers, J., 1996. System dynamics in project management: a comparative analysis with traditional methods. System Dynamics Review 12 (2), 121-139.

Roemer, T.A., Ahmadi, R., Wang, R.H., 2000. Time-cost tradeoffs in overlapped product development. Operations Research 48 (6), 858-865.

Roemer, T.A., Ahmadi, R., 2004. Concurrent crashing and overlapping in product development. Operations Research 52 (4), 606-622.

Roemer, T.A., Ahmadi, R., 2010. Models for concurrent product and process design. European Journal of Operational Research, 203 (3), 601-613.

Rogers, J.L., 1989. A knowledge-based tool for multilevel decomposition of a complex design problem. NASA TP-2903.

Rogers, J.L., 1996. Integrating a genetic algorithm into a knowledge-based system for ordering complex design processes. NASA TM-110247.

Rogers, J.L., 1999. Tools and techniques for decomposing and managing complex design projects. Journal of Aircraft 36 (1), 266-274.

Rosenau, M.D., Griffin, A., Castellion, G., Anschuetz, N., (Eds.), 1996. The PDMA handbook of new product development. John Wiley, New York.

Rosenau, M.D., Githens, G. D., 2005. Successful project management: a step-by-step approach with practical examples. John Wiley, New York.

Serich, S., 2005. Prototype stopping rules in software development projects, IEEE Transactions on Engineering Management, 52 (4): 478-485.

Shane, S.A., Ulrich, K.T., 2004. Technological innovation, product development, and entrepreneurship in management science. Management Science 50 (2), 133-144.

Sharman, D., Yassine A., 2004. Characterizing complex product architectures. Systems Engineering 7 (1), 35-60.

Sharman, D.M., Yassine, A.A., 2007. Architectural valuation using the design structure matrix and real options theory. Concurrent Engineering: Research and Applications 15 (2), 157-173.

Shooman, M.L., 1983. Software engineering: design, reliability, and management, McGraw-Hill, New York.

Slaughter, S.A., Harter, D.E., Krishnan, M.S., 1998. Evaluating the cost of software quality. Communications of the ACM 41(8), 67-73.

Smith, P.G., Reinertsen, D.G., 1998. Developing Products in Half the Time, 2nd ed. Van Nostrand Reinhold, New York.

Smith, R.P., Eppinger, S.D., 1997a. A predictive model of sequential iteration in engineering design. Management Science 43 (8), 1104-1120.

Smith, R.P., Eppinger, S.D., 1997b. Identifying controlling features of engineering design iteration. Management Science 43 (3), 276-293.

Sobek, D.K., Ward, A.C., Liker, J.K., 1999. Toyota's principles of set-based concurrent engineering. Sloan Management Review 40 (2), 67-83.

Song, Z. Kusiak, A., 2010. Mining Pareto-optimal modules for delayed product differentiation. European Journal of Operational Research 201 (1), 123-128.

Sosa, M.E., Eppinger, S.D., Rowles, C.M., 2004. The misalignment of product architecture and organizational structure in complex product development. Management Science 50 (12), 1674-1689.

Sosa, M.E., Eppinger, S.D., Rowles, C.M., 2007a. Are your engineers talking to one another when they should? Harvard Business Review 85 (11), 133-142.

Sosa, M.E., Eppinger, S.D., Rowles, C.M., 2007b. A network approach to define modularity of components in complex products. Journal of Mechanical Design 129 (11), 1118-1129.

Sosa, M.E., 2008. A structured approach to predicting and managing technical interactions in software development. Research in Engineering Design 19(1), 47-70.

Sterman, J.D., 2004. Business Dynamics: Systems Thinking and Modeling for a Complex World. Irwin/McGraw-Hill, Boston.

Steward, D.V., 1981. The design structure system: A method for managing the design of complex systems. IEEE Transactions on Engineering Management 49 (4), 428-442.

Swink, M.L., Sandvig, C., Mabert, V.A., 1996. Customizing concurrent engineering processes: five case studies. Journal of Production Innovation Management 13 (3), 229-244.

Talbi, E.G., 2002. A taxonomy of hybrid metaheuristics. Journal of Heuristics 8 (5), 541-564.

Tamura, Y., Yamada, S., 2006. A flexible stochastic differential equation model in distributed development environment. European Journal of Operational Research 168 (1), 143-152.

Tang, D.B., Zheng, L., Li, Z.Z., Zhang, S.Q., 2000. Re-engineering of the design process for concurrent engineering. Computers and Industrial Engineering 38 (4), 479-491.

Tang, D.B., Zhu, R.M., Dai, S., Zhang, G.J., 2009. Enhancing axiomatic design with design structure matrix. Concurrent Engineering: Research and Applications 17 (2), 129-137.

Terwiesch, C., Loch, C.H., 1999. Measuring the effectiveness of overlapping development activities. Management Science 45 (4), 455-465.

Terwiesch, C., Loch, C.H., De Meyer, A., 2002. Exchanging preliminary information in concurrent engineering: Alternative coordination strategies. Organization Science 13 (4), 402-419.

Terwiesch, C., Xu, Y., 2008. Innovation contests, open innovation, and multiagent problem solving. Management Science 54 (9), 1529-1543.

Thomke, S., 1998. Simulation, learning, and R&D performance: Evidence from automotive development. Research Policy 27 (1), 55-74.

Thomke, S., Fujimoto, T., 2000. The effect of "Front-Loading" problem-solving on product development performance. Journal of Production Innovation Management 17 (2), 128-142.

Thomke, S., Bell, D.E., 2001. Sequential testing in product development. Management Science 47 (2), 308-323.

Thomke, S., 2003. Experimentation Matters: Unlocking the Potential of New Technologies for Innovation. Harvard Business School Press, Boston.

Thomke, S., 2007. Managing Product and Service Department: Text and Cases. McGraw-Hill, Boston.

To, C.K.M., Fung, H.K., Harwood, R.J., Ho, K.C., 2009. Coordinating dispersed product development processes: A contingency perspective of project design and modelling. International Journal of Production Economics 120 (2), 570-584.

Ulrich, K.T., Eppinger S.D., 2008. Product Design and Development (4th ed.). McGraw-Hill, Boston.

Veenstra, V.S., Halman, J.I.M., Voordijk, J.T., 2006. A methodology for developing product platforms in the specific setting of the housebuilding industry. Research in Engineering Design 17 (3), 157-173.

Voss, C.A., Hsuan, J., 2009. Service Architecture and Modularity. Decision Sciences 40 (3), 541-569.

Wagner, S.M., Hoegl, M., 2006. Involving suppliers in product development: insights from R&D directors and project managers. Industrial Marketing Management 35 (8), 936-943.

Waligora G., 2008. Discrete-continuous project scheduling with discounted cash flows - A tabu search approach. Computers and Operations Research 35(7), 2141-2153.

Wang, J., 1999. A fuzzy set approach to activity scheduling for product development. Journal of the Operational Research Society 50 (12), 1217-1228.

Wang, J., 2002. A fuzzy project scheduling approach to minimize schedule risk for product development. Fuzzy Sets and Systems 127 (2), 99-116.

Wang, J., 2004. A fuzzy robust scheduling approach for product development projects. European Journal of Operational Research 152 (1), 180-194.

Wang, Z., Yan, H.S., 2005. Optimizing the Concurrency for a group of design activities. IEEE Transactions on Engineering Management 52 (1), 102-118.

Wheelwright, S.C., Clark, K.B., 1992. Revolutionizing Product Development. The Free Press, New York.

Whitfield, R.I., Duffy, A.H.B., Coates, G., Hills, B. 2003. Efficient process optimization. Concurrent Engineering: Research and Applications 11 (2), 83-92.

Williams, T., Ackermann, F., Eden, C., 2003. Structuring a delay and disruption claim: An application of cause-mapping and system dynamics. European Journal of Operational Research 148 (1), 192-204.

Williams, T., 2005. Assessing and moving on from the dominant project management discourse in the light of project overruns. IEEE Transactions on Engineering Management 52 (4), 497-508.

Wu, L.F., De Matta, R., Lowe, T.J., 2009. Updating a modular product: How to set time to market and component quality. IEEE Transactions on Engineering Management 56 (2), 298-311.

Wu, Y.P., Hu, Q.P., Xie, M., Ng, S.H., 2007. Modeling and analysis of software fault detection and correction process by considering time dependency. IEEE Transactions on Reliability 56 (4), 629-642.

Xie, M., Yang, B., 2001. Optimal testing-time allocation for modular systems. International Journal of Quality and Reliability Management 18 (8), 854-863.

Xie, M., Yang, B., 2003. A Study of the Effect of Imperfect Debugging on Software Development Cost. IEEE Transactions on Software Engineering 29 (5), 471-473.

Xie, M., Hu, Q.P., Wu, Y.P., 2007. A study of the modeling and analysis of software fault-detection and fault-correction processes. Quality and Reliability Engineering 23 (4), 459-470.

Yager, R.R., 1981. A procedure for ordering fuzzy subsets of the unit interval. Information Sciences 24 (2), 143-161.

Yamada, S., Ichimori, T., Nishiwaki, M., 1995. Optimal allocation policies for testing-resource based on a software reliability growth model. Mathematical and Computer Modelling 22 (10-12), 295-301.

Yan, H.S., Wang, Z., Jiang, M., 2002. A quantitative approach to the process modeling and planning in concurrent engineering. Concurrent Engineering: Research and Applications 10 (2), 97-111.

Yan, H.S., Wang, Z., Jiao, X.C., 2003. Modeling, scheduling and simulation of product development process by extended stochastic high-level evaluation Petri nets. Robotics and Computer-Integrated Manufacturing 19 (4), 329-342.

Yan, H.S., Xu, D., 2007. An approach to estimating product design time based on fuzzy v-support vector machine. IEEE Transactions on Neural Networks 18(3), 721-731.

Yang, B., Hu, H., Jia, L., 2008. A study of uncertainty in software cost and its impact on optimal software release time. IEEE Transactions on Software Engineering 34(6), 813-825.

Yassine A.A., Falkenburg, D., Chelst, K., 1999a. Engineering design management: an information structure approach. International Journal of Production Research 37(13), 2957-2975.

Yassine, A.A., Chelst, K.R., Falkenburg, D.R., 1999b. A decision analytic framework for evaluating concurrent engineering. IEEE Transactions on Engineering Management 46 (2), 144–157.

Yassine, A., Joglekar, N., Braha, D., Eppinger, S., Whitney, D., 2003. Information hiding in product development: the design churn effect. Research in Engineering Design 14 (3), 145-161.

Yassine, A.A., 2007. Investigating product development process reliability and robustness using simulation. Journal of Engineering Design 18 (6), 545-561.

Yassine, A.A., Sreenivas, R.S., Zhu, J., 2008. Managing the exchange of information in product development. European Journal of Operational Research 184 (1), 311-326.

Yu, T.L., Yassine, A.A., Goldberg, D.E., 2007. An information theoretic method for developing modular architectures using genetic algorithms. Research in Engineering Design 18 (2), 91-109.

Zakarian, A., Kusiak, A., 2001. Process analysis and reengineering. Computers and Industrial Engineering 41 (2), 135-150.

Zhang, H.P., Qiu, W.H., Zhang, H.F., 2006. An approach to measuring coupled tasks strength and sequencing of coupled tasks in new product development. Concurrent Engineering: Research and Applications 14 (4), 305-311.

Zhang, X.M., Ding, H., 2007. Interval finite element method for complex eigenvalues of closed-loop systems with uncertain parameters. Structural Engineering and Mechanics 26 (2) 163-178.

Zhou, K.Z., Yim, C.K., Tse, D.K., 2005. The effects of strategic orientations on technology- and market-based breakthrough innovations. Journal of Marketing, 69 (2), 42-60.

Zhuo, L., San, W.Y., Seng, L.K., 2008. Integrated approach to modularize the conceptual product family architecture. International Journal of Advanced Manufacturing 36 (1-2), 83-96.

Zielinski, P., 2005. On computing the latest starting times and floats of activities in a network with imprecise durations. Fuzzy Sets and Systems 150 (1), 53-76.

Zimmermann, H.-J., 1996. Fuzzy set theory and its applications (3rd ed.). Kluwer Academic Publishers, Boston.

# APPENDIX A

# PROOFS OF CHAPTER 3

**Proof of Proposition 3.1**

**(a)** The first and second partial derivatives of (3.9) in $t_s$ are:

$$\frac{\partial C}{\partial t_s} = \beta + ku_1(c_r + \beta)\left(e^{-b_1 t_1} - e^{-b_1 t_s}\right) \tag{3.20}$$

$$\frac{\partial^2 C}{\partial t_s^2} = ku_1 b_1(c_r + \beta)e^{-b_1 t_s} \tag{3.21}$$

Let $\theta_1 = \dfrac{\beta}{ku_1(c_r + \beta)}$. Clearly, $\partial^2 C/\partial t_s^2 > 0$ for all feasible values of $t_s$. Hence, if $t_1^*$ is given, and if $\theta_1 \geq 1 - e^{-b_1 t_1^*}$, then $\partial C/\partial t_s \geq 0$ for all $t_s$. In such case, $t_s^* = 0$. On the other hand, if $\theta_1 < 1 - e^{-b_1 t_1^*}$, then setting (3.20) to zero yields:

$$t_s^* = -\frac{1}{b_1}\ln\left(\theta_1 + e^{-b_1 t_1^*}\right) \tag{3.22}$$

**(b)** Since the maximum value of $\left(1 - e^{-b_1 t_1^*}\right)$ is 1, we can get the result in Proposition 3.1(b).

**(c)** The first and second partial derivatives of (3.9) with respect to $t_2$ are:

$$\frac{\partial C}{\partial t_2} = \beta + c_2 - b_2 c_q e^{-b_2 t_2}\left[a_2 + ku_1(w_2 - a_2)e^{-b_1 t_1}\right] \tag{3.23}$$

$$\frac{\partial^2 C}{\partial t_2^2} = b_2^2 c_q e^{-b_2 t_2}\left[a_2 + ku_1(w_2 - a_2)e^{-b_1 t_1}\right] \tag{3.24}$$

Let $\theta_2 = \dfrac{c_2 + \beta}{c_q b_2 k u_1 (w_2 - a_2)}$, and $\theta_3 = \dfrac{a_2}{k u_1 (w_2 - a_2)}$. Obviously, $\partial^2 C / \partial t_2^2 > 0$ for all $t_2$. If $t_1^*$ is given, and if $\theta_2 - \theta_3 \geq e^{-b_1 t_1^*}$, then $\partial C / \partial t_2 \geq 0$ for all $t_2$, leading to $t_2^* = 0$. Otherwise, setting (3.23) to zero yields:

$$t_2^* = \frac{1}{b_2} \ln\left( \frac{\theta_3 + e^{-b_1 t_1^*}}{\theta_2} \right) \tag{3.25}$$

**(d)** The maximum value of $e^{-b_1 t_1^*}$ is 1. Hence if $\theta_2 - \theta_3 \geq 1$, $t_2^* = 0$.

**Proof of Corollary 3.1**

When $t_2^* \neq 0$, by substituting (3.25) into (3.7), we get $N(t_1^*, t_2^*) = \dfrac{c_2 + \beta}{c_q b_2}$.

**Proof of Proposition 3.2**

**(a)** Obviously, $\left.\dfrac{\partial \Phi(t_1)}{\partial t_1}\right|_{t_s^* = 0} = 1$, and $\left.\dfrac{\partial \Phi(t_1)}{\partial t_1}\right|_{t_s^* \neq 0} = \dfrac{\theta_1}{\theta_1 + e^{-b_1 t_1}}$. Thus, $\dfrac{\partial \Phi(t_1)}{\partial t_1} > 0 \ \ \forall\, t_1$.

**(b)** If $t_s^* \neq 0$, $t_2^* \neq 0$, the first partial derivative of (3.14) with respect to $t_1$ is:

$$\left.\frac{\partial C}{\partial t_1}\right|_{t_s^* \neq 0, t_2^* \neq 0} = (c_r + \beta) k u_1 e^{-b_1 t_1} \left[ b_1 t_1 + \ln\left(\theta_1 + e^{-b_1 t_1}\right) \right] - \frac{(c_2 + \beta) b_1 e^{-b_1 t_1}}{b_2 (\theta_3 + e^{-b_1 t_1})} + c_1 \tag{3.26}$$

As above, we get following equations:

$$\left.\frac{\partial C}{\partial t_1}\right|_{t_s^* \neq 0, t_2^* = 0} = k u_1 e^{-b_1 t_1} \left\{ (c_r + \beta)\left[ b_1 t_1 + \ln\left(\theta_1 + e^{-b_1 t_1}\right) \right] - c_q b_1 (w_2 - a_2) \right\} + c_1 \tag{3.27}$$

$$\left.\frac{\partial C}{\partial t_1}\right|_{t_s^* = 0, t_2^* \neq 0} = (c_r + \beta) k u_1 b_1 t_1 e^{-b_1 t_1} - \frac{(c_2 + \beta) b_1 e^{-b_1 t_1}}{b_2 (\theta_3 + e^{-b_1 t_1})} + c_1 \tag{3.28}$$

$$\left.\frac{\partial C}{\partial t_1}\right|_{t_s^* = 0, t_2^* = 0} = k u_1 b_1 e^{-b_1 t_1} \left[ (c_r + \beta) t_1 - c_q (w_2 - a_2) \right] + c_1 \tag{3.29}$$

The second partial derivatives of (3.14) with respect to $t_1$ for all four situations are:

$$\left.\frac{\partial^2 C}{\partial t_1^2}\right|_{t_s^*\neq 0, t_2^*\neq 0} = b_1 e^{-b_1 t_1}\left\{ku_1(c_r+\beta)\left[\frac{\theta_1}{\theta_1+e^{-b_1 t_1}} - b_1 t_1 - \ln\left(\theta_1 + e^{-b_1 t_1}\right)\right] + \frac{b_1\theta_3(c_2+\beta)}{b_2\left(\theta_3+e^{-b_1 t_1}\right)^2}\right\} \quad (3.30)$$

$$\left.\frac{\partial^2 C}{\partial t_1^2}\right|_{t_s^*\neq 0, t_2^*=0} = ku_1 b_1 e^{-b_1 t_1}\left\{(c_r+\beta)\left[\frac{\theta_1}{\theta_1+e^{-b_1 t_1}} - b_1 t_1 - \ln\left(\theta_1 + e^{-b_1 t_1}\right)\right] + c_q b_1(w_2-a_2)\right\} \quad (3.31)$$

$$\left.\frac{\partial^2 C}{\partial t_1^2}\right|_{t_s^*=0, t_2^*\neq 0} = b_1 e^{-b_1 t_1}\left[ku_1(c_r+\beta)(1-b_1 t_1) + \frac{b_1\theta_3(c_2+\beta)}{b_2\left(\theta_3+e^{-b_1 t_1}\right)^2}\right] \quad (3.32)$$

$$\left.\frac{\partial^2 C}{\partial t_1^2}\right|_{t_s^*=0, t_2^*=0} = ku_1 b_1 e^{-b_1 t_1}\left[(c_r+\beta)(1-b_1 t_1) + c_q b_1(w_2-a_2)\right] \quad (3.33)$$

Observe from (3.30) to (3.33), the second derivatives for all four cases decrease from a positive value to a negative global minimum value, then increases monotonically with $t_1$, and finally they approach zero when $t_1$ approaches positive infinity. The first derivatives for all four cases strictly increase to a global maximum value, then decrease to $c_1$ (positive) when $t_1$ approaches positive infinity. Consequently, the total cost in (3.14) is first convex then concave increasing with respect to $t_1$.

Let $\theta_4 = \dfrac{c_1}{c_q k u_1(w_2-a_2)}$. Clearly, if $\left.\dfrac{\partial C}{\partial t_1}\right|_{t_1=0} \geq 0$, i.e. $\dfrac{\theta_4}{b_1} \geq \min\left[1, \dfrac{\theta_2}{\theta_3+1}\right]$, then $\partial C/\partial t_1 \geq 0$ for all $t_1$, resulting in $t_1^* = 0$.

**Proof of Proposition 3.3**

**(a)** When $t_1^* + \dfrac{1}{b_1}\ln\left(\theta_1 + e^{-b_1 t_1^*}\right) \leq T_2$, and $t_s^* \neq 0$, $t_2^* \neq 0$, $t_1^* \neq 0$, by the implicit function theorem: $\dfrac{\partial t_1^*}{\partial u_1} = -\left.\dfrac{\partial^2 C}{\partial t_1 \partial u_1}\right|_{t_1^*} \Big/ \left.\dfrac{\partial^2 C}{\partial t_1^2}\right|_{t_1^*}$, and $\dfrac{\partial t_1^*}{\partial k} = -\left.\dfrac{\partial^2 C}{\partial t_1 \partial k}\right|_{t_1^*} \Big/ \left.\dfrac{\partial^2 C}{\partial t_1^2}\right|_{t_1^*}$, we can get:

$$\frac{\partial t_1^*}{\partial u_1}=\frac{e^{-b_1t_1^*}\left\{k(c_r+\beta)\left[\frac{\theta_1}{\theta_1+e^{-b_1t_1^*}}-b_1t_1^*-\ln\left(\theta_1+e^{-b_1t_1^*}\right)\right]+\frac{b_1\theta_3(c_2+\beta)}{u_1b_2\left(\theta_3+e^{-b_1t_1^*}\right)^2}\right\}}{b_1e^{-b_1t_1^*}\left\{ku_1(c_r+\beta)\left[\frac{\theta_1}{\theta_1+e^{-b_1t_1^*}}-b_1t_1^*-\ln\left(\theta_1+e^{-b_1t_1^*}\right)\right]+\frac{b_1\theta_3(c_2+\beta)}{b_2\left(\theta_3+e^{-b_1t_1^*}\right)^2}\right\}}=\frac{1}{u_1b_1} \quad (3.34)$$

$$\frac{\partial t_1^*}{\partial k}=\frac{e^{-b_1t_1^*}\left\{u_1(c_r+\beta)\left[\frac{\theta_1}{\theta_1+e^{-b_1t_1^*}}-b_1t_1^*-\ln\left(\theta_1+e^{-b_1t_1^*}\right)\right]+\frac{b_1\theta_3(c_2+\beta)}{kb_2\left(\theta_3+e^{-b_1t_1^*}\right)^2}\right\}}{b_1e^{-b_1t_1^*}\left\{ku_1(c_r+\beta)\left[\frac{\theta_1}{\theta_1+e^{-b_1t_1^*}}-b_1t_1^*-\ln\left(\theta_1+e^{-b_1t_1^*}\right)\right]+\frac{b_1\theta_3(c_2+\beta)}{b_2\left(\theta_3+e^{-b_1t_1^*}\right)^2}\right\}}=\frac{1}{kb_1} \quad (3.35)$$

Therefore, $t_1^*$ increases with $u_1$ and $k$. Similarly, we have:

$$\frac{\partial t_1^*}{\partial c_r}=ku_1e^{-b_1t_1^*}\left[\frac{\theta_1}{\theta_1+e^{-b_1t_1^*}}-b_1t_1^*-\ln\left(\theta_1+e^{-b_1t_1^*}\right)\right]\Bigg/\left.\frac{\partial^2 C}{\partial t_1^2}\right|_{t_1^*} \quad (3.36)$$

It can be verified that: $\frac{\theta_1}{\theta_1+e^{-b_1t_1^*}}-b_1t_1^*-\ln\left(\theta_1+e^{-b_1t_1^*}\right)<0,\ \forall t_1^*$. Thus, $\frac{\partial t_1^*}{\partial c_r}<0$.

**(b)** When $t_s^*\neq 0$, $t_2^*\neq 0$, $t_1^*\neq 0$ and $t_1^*+\frac{1}{b_1}\ln\left(\theta_1+e^{-b_1t_1^*}\right)\leq T_2$, the mixed partial:

$$\left.\frac{\partial^2 C}{\partial t_1\partial b_1}\right|_{t_1^*}=ku_1t_1^*(c_r+\beta)e^{-b_1t_1^*}\left[\frac{\theta_1}{\theta_1+e^{-b_1t_1^*}}-b_1t_1^*-\ln\left(\theta_1+e^{-b_1t_1^*}\right)\right]+\frac{(c_2+\beta)(b_1\theta_3t_1^*-\theta_3-e^{-b_1t_1^*})e^{-b_1t_1^*}}{b_2(\theta_3+e^{-b_1t_1^*})^2}$$

(3.37)

When $b_1\theta_3t_1^*-\theta_3-e^{-b_1t_1^*}\leq 0$, the above expression is negative. Moreover, (3.37) strictly increases with $b_1$ when $b_1\theta_3t_1^*-\theta_3-e^{-b_1t_1^*}>0$. Therefore, there exists a unique value of $b_1^0$ such that: if $b_1<b_1^0$, then the optimal upstream testing duration $t_1^*$ increases with $b_1$; if $b_1>b_1^0$, $t_1^*$ decreases with $b_1$.

**(c)** When $t_s^*\neq 0$, $t_2^*\neq 0$, $t_1^*\neq 0$ and $t_1^*+\frac{1}{b_1}\ln\left(\theta_1+e^{-b_1t_1^*}\right)\leq T_2$, the mixed partial:

$$\left.\frac{\partial^2 C}{\partial t_1 \partial \beta}\right|_{t_1^*} = e^{-b_1 t_1^*}\left\{ k u_1 \left[ b_1 t_1^* + \ln\left(\theta_1 + e^{-b_1 t_1^*}\right) + \frac{c_r \theta_1}{\beta(\theta_1 + e^{-b_1 t_1^*})}\right] - \frac{b_1}{b_2(\theta_3 + e^{-b_1 t_1^*})}\right\}$$

$$= \frac{-1}{c_r + \beta}\left[ c_1 - \frac{c_r e^{-b_1 t_1^*}}{(\theta_1 + e^{-b_1 t_1^*})} + \frac{(c_r - c_2) b_1 e^{-b_1 t_1^*}}{b_2(\theta_3 + e^{-b_1 t_1^*})}\right] \tag{3.38}$$

Let $\Omega(\beta) = c_1 - \frac{c_r e^{-b_1 t_1^*}}{(\theta_1 + e^{-b_1 t_1^*})} + \frac{(c_r - c_2) b_1 e^{-b_1 t_1^*}}{b_2(\theta_3 + e^{-b_1 t_1^*})}$. It can be verified that $\Omega(\beta)$ strictly increases in $\beta$. If $c_r < c_2 < c_1$, then $\Omega(\beta)$ is positive, and so $t_1^*$ strictly increases with $\beta$. Otherwise, there exists a critical $\beta^0$ such that: if $\beta < \beta^0$, $t_1^*$ decreases with $\beta$; if $\beta > \beta^0$, $t_1^*$ increases with $\beta$.

**Proof of Corollary 3.2**

When $t_s^* \neq 0$, $t_2^* \neq 0$, $t_1^* \neq 0$ and $t_1^* + \frac{1}{b_1}\ln\left(\theta_1 + e^{-b_1 t_1^*}\right) \leq T_2$, the first partial derivatives of (3.22) with respect to $u_1$ and $k$ are:

$$\frac{\partial t_s^*}{\partial u_1} = \frac{\left[\frac{\beta}{k u_1^2 (c_r + \beta)} + b_1 e^{-b_1 t_1^*} \times \frac{\partial t_1^*}{\partial u_1}\right]}{b_1\left[\frac{\beta}{k u_1 (c_r + \beta)} + e^{-b_1 t_1^*}\right]} = \frac{\left[\frac{\beta}{k u_1^2 (c_r + \beta)} + b_1 e^{-b_1 t_1^*} \times \frac{1}{u_1 b_1}\right]}{b_1\left[\frac{\beta}{k u_1 (c_r + \beta)} + e^{-b_1 t_1^*}\right]} = \frac{1}{u_1 b_1} \tag{3.39}$$

$$\frac{\partial t_s^*}{\partial k} = \frac{\left[\frac{\beta}{k^2 u_1 (c_r + \beta)} + b_1 e^{-b_1 t_1^*} \times \frac{\partial t_1^*}{\partial k}\right]}{b_1\left[\frac{\beta}{k u_1 (c_r + \beta)} + e^{-b_1 t_1^*}\right]} = \frac{\left[\frac{\beta}{k^2 u_1 (c_r + \beta)} + b_1 e^{-b_1 t_1^*} \times \frac{1}{k b_1}\right]}{b_1\left[\frac{\beta}{k u_1 (c_r + \beta)} + e^{-b_1 t_1^*}\right]} = \frac{1}{k b_1} \tag{3.40}$$

Therefore, $t_s^*$ increases with $u_1$ and $k$.

Defining overlapping degree between upstream and downstream stages $\gamma = 1 - t_s / t_1$, and $\gamma^*$ be the optimal overlapping degree. Then, based on the above results of (3.34), (3.35), (3.39), and (3.40), our model suggests that $\gamma^*$ decreases with

$u_1$ and $k$. This result is consistent with Loch and Terwiesch (1998). The proof is given below:

The first partial derivatives of $\gamma^*$ with respect to $u_1$ and $k$ are:

$$\frac{\partial \gamma^*}{\partial u_1} = -\frac{t_1^* \times \frac{\partial t_s^*}{\partial u_1} - t_s^* \times \frac{\partial t_1^*}{\partial u_1}}{(t_1^*)^2} = \frac{(t_s^* - t_1^*) \times \frac{1}{u_1 b_1}}{(t_1^*)^2} \tag{3.41}$$

$$\frac{\partial \gamma^*}{\partial k} = -\frac{t_1^* \times \frac{\partial t_s^*}{\partial k} - t_s^* \times \frac{\partial t_1^*}{\partial k}}{(t_1^*)^2} = \frac{(t_s^* - t_1^*) \times \frac{1}{k b_1}}{(t_1^*)^2} \tag{3.42}$$

Clearly, $t_s^* - t_1^* = -\frac{1}{b_1}\ln\left(\theta_1 + e^{-b_1 t_1^*}\right) - t_1^* \le 0$. Thus, $\frac{\partial \gamma^*}{\partial u_1} \le 0$, and $\frac{\partial \gamma^*}{\partial k} \le 0$.

**Proof of Corollary 3.3**

When $t_s^* \ne 0$, $t_2^* \ne 0$, $t_1^* \ne 0$ and $t_1^* + \frac{1}{b_1}\ln\left(\theta_1 + e^{-b_1 t_1^*}\right) \le T_2$, the first partial derivatives of (3.25) with respect to $c_2$ and $a_2$ are:

$$\frac{\partial t_2^*}{\partial c_2} = -\frac{1}{b_2}\left[ b_1 \frac{\partial t_1^* / \partial c_2}{(\theta_3 e^{b_1 t_1^*} + 1)} + \frac{1}{(c_2 + \beta)} \right] \tag{3.43}$$

$$\frac{\partial t_2^*}{\partial a_2} = \frac{\theta_3\left(1 - k u_1 e^{-b_1 t_1^*}\right) - b_1 a_2 e^{-b_1 t_1^*} \times \frac{\partial t_1^*}{\partial a_2}}{a_2 b_2 (\theta_3 + e^{-b_1 t_1^*})} \tag{3.44}$$

In (3.43), $\frac{\partial t_1^*}{\partial c_2} = \frac{b_1}{b_2(\theta_3 e^{b_1 t_1^*} + 1)} \Big/ \left.\frac{\partial^2 C}{\partial t_1^2}\right|_{t_1^*}$, which is larger than zero, we get: $\frac{\partial t_2^*}{\partial c_2} < 0$.

Similarly, $\frac{\partial t_1^*}{\partial a_2} = \frac{-(c_2 + \beta) b_1 w_2 e^{-b_1 t_1^*}}{b_2 k u_1 (w_2 - a_2)^2 (\theta_3 + e^{-b_1 t_1^*})^2} \Big/ \left.\frac{\partial^2 C}{\partial t_1^2}\right|_{t_1^*}$, which is less than zero, we get: $\frac{\partial t_2^*}{\partial a_2} > 0$. Thus, the optimal downstream testing duration $t_2^*$ decreases with downstream testing cost $c_2$, and increases with $a_2$.

**Proof of Proposition 3.4**

**(a)** The Hessian matrix of (3.16) is:

$$H(\tau_1,\tau_2)=\begin{bmatrix} \dfrac{\partial^2 C}{\partial \tau_1^2} & \dfrac{\partial^2 C}{\partial \tau_1 \partial \tau_2} \\ \dfrac{\partial^2 C}{\partial \tau_2 \partial \tau_1} & \dfrac{\partial^2 C}{\partial \tau_2^2} \end{bmatrix} = \begin{bmatrix} \dfrac{a_2 b_1^2 c_q}{\theta_3} e^{-b_1\tau_1 - b_2\tau_2} & \dfrac{a_2 b_1 b_2 c_q}{\theta_3} e^{-b_1\tau_1 - b_2\tau_2} \\ \dfrac{a_2 b_1 b_2 c_q}{\theta_3} e^{-b_1\tau_1 - b_2\tau_2} & a_2 b_2^2 c_q e^{-b_2\tau_2}\left(1+e^{-b_1\tau_1}/\theta_3\right) \end{bmatrix} \tag{3.45}$$

The above $H(\tau_1,\tau_2)$ is symmetric. Moreover, the first leading principle minor of $H(\tau_1,\tau_2)$, $\dfrac{a_2 b_1^2 c_q}{\theta_3} e^{-b_1\tau_1 - b_2\tau_2}$, is positive for all feasible $\tau_1$ and $\tau_2$. The second leading principle minor of $H(\tau_1,\tau_2)$, $\dfrac{a_2^2 b_1^2 b_2^2 c_q^2}{\theta_3} e^{-b_1\tau_1 - 2b_2\tau_2}$, is also positive for all feasible $\tau_1$ and $\tau_2$. Thus, $H(\tau_1,\tau_2)$ is positive definite, and the objective function (3.16) is strictly convex.

The first partial derivatives of (3.16) with respect to $\tau_1$ and $\tau_2$ are:

$$\frac{\partial C}{\partial \tau_1} = c_1 + \beta - k u_1 b_1 c_q (w_2 - a_2) e^{-b_1\tau_1 - b_2\tau_2} \tag{3.46}$$

$$\frac{\partial C}{\partial \tau_2} = c_2 + \beta - b_2 c_q e^{-b_2\tau_2}\left[a_2 + k u_1 (w_2 - a_2) e^{-b_1\tau_1}\right] \tag{3.47}$$

Since the objective (3.16) is convex and the constraint (3.17) is linear, we get:

$$\tau_2^* = \max\left[0, \frac{1}{b_2}\ln\left(\frac{\theta_3 + e^{-b_1\tau_1^*}}{\theta_2}\right)\right] \tag{3.48}$$

If $\dfrac{\theta_4}{b_1} \geq \min\left[1, \dfrac{\theta_2}{\theta_3 + 1}\right]$, then $\tau_1^* = 0$. Otherwise,

$$\tau_1^* = \frac{1}{b_1}\ln\left[\frac{b_1(c_2+\beta)}{\theta_3 b_2 (c_1+\beta)} - \frac{1}{\theta_3}\right] \tag{3.49}$$

Obviously, the conditions for $\tau_1^* = 0$ and $\tau_2^* = 0$ are same as those for $t_1^* = 0$ and $t_2^* = 0$. Therefore, we only need to prove that $\tau_1^* \le t_1^*$ and $\tau_2^* \ge t_2^*$, when $\tau_1^* \ne 0$ and $\tau_2^* \ne 0$. Suppose that $\tau_1^* > t_1^*$, then $\dfrac{b_1 e^{-b_1\tau_1^*}(c_2+\beta)}{b_2(\theta_3 + e^{-b_1\tau_1^*})} < \dfrac{b_1 e^{-b_1 t_1^*}(c_2+\beta)}{b_2(\theta_3 + e^{-b_1 t_1^*})}$. It follows that:

$$ku_1 b_1 c_q (w_2 - a_2) e^{-b_1\tau_1^* - b_2\tau_2^*} < ku_1 b_1 c_q (w_2 - a_2) e^{-b_1 t_1^* - b_2 t_2^*} \tag{3.50}$$

Let $\eta = t_1^* + \dfrac{1}{b_1}\ln\left(\theta_1 + e^{-b_1 t_1^*}\right)$. If (3.50) holds, then $c_1 + \beta < c_1 + ku_1 b_1 (c_r + \beta)\eta e^{-b_1 t_1^*}$, resulting in:

$$e^{b_1\eta} < b_1\eta + 1 \tag{3.51}$$

Since $\eta > 0$, it is clear that (3.51) does not hold. It follows that $\tau_1^* \le t_1^*$. Inspection of (3.13) and (3.48) shows that if $\tau_1^* \le t_1^*$, then $\tau_2^* \ge t_2^*$.

**(b)** Defining $y = t_1^* - \tau_1^*$, and $F(y) = t_1^* - \tau_1^* + t_2^* - \tau_2^*$. In Proposition 3.4(a), we have proved that $\tau_1^* \le t_1^*$, and thus we get $y \ge 0$. We have also shown that the conditions for $\tau_1^* = 0$ and $\tau_2^* = 0$ are same as those for $t_1^* = 0$ and $t_2^* = 0$. Thus, in what follows, we only consider the case that $\tau_1^* \ne 0$ and $\tau_2^* \ne 0$.

Substitute $t_2^* = \dfrac{1}{b_2}\ln\left(\dfrac{\theta_3 + e^{-b_1 t_1^*}}{\theta_2}\right)$, and $\tau_2^* = \dfrac{1}{b_2}\ln\left(\dfrac{\theta_3 + e^{-b_1\tau_1^*}}{\theta_2}\right)$ into $F(y)$, we get:

$$F(y) = y + \frac{1}{b_2}\ln\left(\frac{\theta_3 + e^{-b_1 t_1^*}}{\theta_3 + e^{-b_1\tau_1^*}}\right) = y + \frac{1}{b_2}\ln\left[\frac{\theta_3 + e^{-b_1(\tau_1^* + y)}}{\theta_3 + e^{-b_1\tau_1^*}}\right] \tag{3.52}$$

From (3.49), we get:

$$e^{-b_1\tau_1^*} = \frac{\theta_3 b_2 (c_1 + \beta)}{b_1(c_2 + \beta) - b_2(c_1 + \beta)} \tag{3.53}$$

Substitute (3.53) into (3.52), we get:

$$F(y) = y + \frac{1}{b_2}\ln\left[\frac{b_1(c_2+\beta)+b_2(c_1+\beta)(e^{-b_1 y}-1)}{b_1(c_2+\beta)}\right] \tag{3.54}$$

The first and second derivatives of (3.54) with respect to $y$ are:

$$\frac{dF(y)}{dy} = 1 - \frac{b_1(c_1+\beta)e^{-b_1 y}}{b_1(c_2+\beta)+b_2(c_1+\beta)(e^{-b_1 y}-1)} \tag{3.55}$$

$$\frac{d^2F(y)}{dy^2} = \frac{b_1^2(c_1+\beta)[b_1(c_2+\beta)-b_2(c_1+\beta)]e^{-b_1 y}}{[b_1(c_2+\beta)+b_2(c_1+\beta)(e^{-b_1 y}-1)]^2} \tag{3.56}$$

Inspection of (3.53) shows that $b_1(c_2+\beta)-b_2(c_1+\beta)\ge 0$, leading to $\frac{d^2F(y)}{dy^2}\ge 0$, and $\frac{dF(y)}{dy}$ increases monotonically in $y$. If $\left.\frac{dF(y)}{dy}\right|_{y=0}\ge 0$, i.e. if $c_2 \ge c_1$, then $\frac{dF(y)}{dy}\ge 0$, and $F(y)$ increases monotonically in $y$. When $y=0$, $F(y)=0$. Thus, if $c_2 \ge c_1$, then $F(y)\ge 0$ for all $y$, leading to $t_1^* + t_2^* \ge \tau_1^* + \tau_2^*$.

**Proof of Proposition 3.5**

To solve the problem (3.18), we associate a nonnegative multiplier $\lambda_1$ with the constraint $N(t_1,t_2)=[a_2+k(w_2-a_2)u_1e^{-b_1t_1}]e^{-b_2t_2}\le \overline{N}$, and obtain the following Lagrangian function:

$$L_1 = \beta[t_s + t_2 + R(t_s,t_1)] + c_1t_1 + c_2t_2 + c_rR(t_s,t_1) + \lambda_1[N(t_1,t_2)-\overline{N}] \tag{3.57}$$

Where $R(t_s,t_1) = ku_1\left[\frac{e^{-b_1t_s}}{b_1} - \left(t_1 - t_s + \frac{1}{b_1}\right)e^{-b_1t_1}\right]$.

The first derivatives of (3.57) with respect to $t_s$, $t_1$, $t_2$ and $\lambda_1$ are:

$$\frac{\partial L_1}{\partial t_s} = \beta + ku_1(c_r+\beta)\left(e^{-b_1t_1} - e^{-b_1t_s}\right) \tag{3.58}$$

$$\frac{\partial L_1}{\partial t_1} = c_1 - \lambda_1 k u_1 b_1 (w_2 - a_2) e^{-b_1 t_1 - b_2 t_2} + (c_r + \beta) k u_1 b_1 e^{-b_1 t_1} (t_1 - t_s) \tag{3.59}$$

$$\frac{\partial L_1}{\partial t_2} = \beta + c_2 - \lambda_1 b_2 e^{-b_2 t_2} [a_2 + k u_1 (w_2 - a_2) e^{-b_1 t_1}] \tag{3.60}$$

$$\frac{\partial L_1}{\partial \lambda_1} = [a_2 + k u_1 (w_2 - a_2) e^{-b_1 t_1}] e^{-b_2 t_2} - \overline{N} \tag{3.61}$$

By the first order conditions, we have:

$$t_s = \max\left[0,\ -\frac{1}{b_1} \ln\left(\frac{\beta}{k u_1 (c_r + \beta)} + e^{-b_1 t_1}\right)\right] \tag{3.62}$$

$$t_2 = \max\left[0,\ \frac{1}{b_2} \ln\left(\frac{a_2 + k u_1 (w_2 - a_2) e^{-b_1 t_1}}{\overline{N}}\right)\right] \tag{3.63}$$

$$\lambda_1 = \frac{\beta + c_2}{b_2 \overline{N}} \tag{3.64}$$

$$c_1 - \frac{\beta + c_2}{b_2 \overline{N}} k u_1 b_1 (w_2 - a_2) e^{-b_1 t_1} \times \min\left[1, \frac{\overline{N}}{a_2 + k u_1 (w_2 - a_2) e^{-b_1 t_1}}\right]$$

$$+ (c_r + \beta) k u_1 b_1 e^{-b_1 t_1} \left\{ t_1 - \max\left[0, -\frac{1}{b_1} \ln\left(\frac{\beta}{k u_1 (c_r + \beta)} + e^{-b_1 t_1}\right)\right]\right\} = 0 \tag{3.65}$$

The solutions to the *cost minimization problem* are:

$$t_s^* = \max\left[0, -\frac{1}{b_1} \ln\left(\theta_1 + e^{-b_1 t_1^*}\right)\right] = \max\left[0, -\frac{1}{b_1} \ln\left(\frac{\beta}{k u_1 (c_r + \beta)} + e^{-b_1 t_1^*}\right)\right] \tag{3.12}$$

$$t_2^* = \max\left[0, \frac{1}{b_2} \ln\left(\frac{\theta_3 + e^{-b_1 t_1^*}}{\theta_2}\right)\right] = \max\left[0, \frac{1}{b_2} \ln\left(\frac{b_2 c_q [a_2 + k u_1 (w_2 - a_2) e^{-b_1 t_1^*}]}{\beta + c_2}\right)\right] \tag{3.13}$$

$$c_1 - c_q k u_1 b_1 (w_2 - a_2) e^{-b_1 t_1} \times \min\left[1, \frac{\beta + c_2}{b_2 c_q [a_2 + k u_1 (w_2 - a_2) e^{-b_1 t_1}]}\right]$$

$$+ (c_r + \beta) k u_1 b_1 e^{-b_1 t_1} \left\{ t_1 - \max\left[0, -\frac{1}{b_1} \ln\left(\frac{\beta}{k u_1 (c_r + \beta)} + e^{-b_1 t_1}\right)\right]\right\} = 0 \tag{3.66}$$

Where Equation (3.66) is the summary of Equations (3.26)-(3.29). By comparing

(3.62) with (3.12), (3.63) with (3.13), (3.65) and (3.66), we can conclude that the *target quality problem* in (3.18) have same solution structure as the *cost minimization problem* in (3.9)-(3.11). If $c_q = \lambda_1 = \frac{\beta + c_2}{b_2 \overline{N}}$, then the *target quality problem* and the *cost minimization problem* have exact same solutions.

To solve the *budget constraint problem* in (3.19), we associate a nonnegative multiplier $\lambda_2$ with the constraint $C_D = \beta[t_s + t_2 + R(t_s, t_1)] + c_1 t_1 + c_2 t_2 + c_r R(t_s, t_1) \le \overline{C}$, and obtain the following Lagrangian function:

$$L_2 = [a_2 + k(w_2 - a_2)u_1 e^{-b_1 t_1}]e^{-b_2 t_2} + \lambda_2 \left(C_D - \overline{C}\right) \tag{3.67}$$

By the first order conditions, we get:

$$t_s = \max\left[0,\ -\frac{1}{b_1}\ln\left(\frac{\beta}{ku_1(c_r + \beta)} + e^{-b_1 t_1}\right)\right] \tag{3.68}$$

$$t_2 = \frac{1}{c_2 + \beta}\left\{\overline{C} - \beta t_s - c_1 t_1 - (c_r + \beta)ku_1\left[\frac{e^{-b_1 t_s}}{b_1} - \left(t_1 - t_s + \frac{1}{b_1}\right)e^{-b_1 t_1}\right]\right\} \tag{3.69}$$

$$\lambda_2 = \frac{[a_2 + k(w_2 - a_2)u_1 e^{-b_1 t_1}]b_2 e^{-b_2 t_2}}{(c_2 + \beta)} \tag{3.70}$$

$$\begin{aligned}&(c_r + \beta)ku_1 b_1 e^{-b_1 t_1}\left\{t_1 - \max\left[0, -\frac{1}{b_1}\ln\left(\frac{\beta}{ku_1(c_r + \beta)} + e^{-b_1 t_1}\right)\right]\right\} \\ &+ c_1 - \frac{(\beta + c_2)ku_1 b_1 (w_2 - a_2)e^{-b_1 t_1}}{b_2[a_2 + ku_1(w_2 - a_2)e^{-b_1 t_1}]} = 0\end{aligned} \tag{3.71}$$

Clearly, if $t_2^* \neq 0$, then the optimal $t_1^*$ and $t_s^*$ to the *budget constraint problem* are exactly same with those to the *cost minimization problem*, and so can be derived by the same approach. If $c_q = \frac{1}{\lambda_2}$, then the *budget constraint problem* and the *cost minimization problem* have exact same solutions.

# APPENDIX B

# PROOFS OF CHAPTER 4

**Proof of Proposition 4.1**

As illustrated in Figure B.1, defining $y_{i+1}^*$ be the optimal overlap between stage $i$ and $i+1$, $z_i^*$ be the optimal number of tests completed in stage $i$ before starting stage $i+1$, and $x_i^*$ be the optimal number of tests needed in stage $i$, that solve the problem (4.7). Given $x_i^*$ and $z_i^*$, suppose that $y_{i+1}^* \neq T_i(x_i^*) - T_i(z_i^*)$, and there exist a positive $u$ such that:

$$y_{i+1}^* = T_i(x_i^*) - T_i(z_i^*) - u \tag{4.16}$$

Where $u < T_i(z_i^* + 1) - T_i(z_i^*)$. In other words, in (4.16), we assume that it's optimal to start initial design of stage $i+1$ $u$ time units after the completion of $z_i^{th}$ test in stage $i$. In what follows, we will prove $u$ must equal to zero.

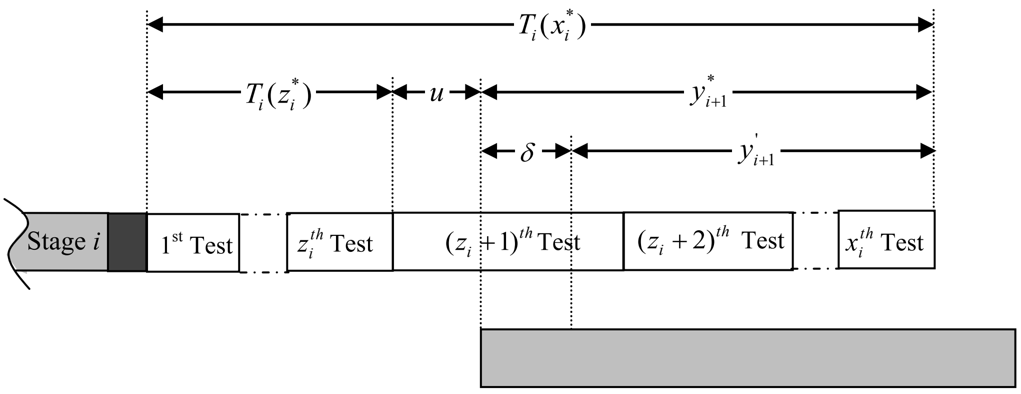


**Figure B.1 The scenario used in the proof of proposition 4.1**

Assume that there exist a positive $\delta$ such that $u+\delta < T_i(z_i^*+1)-T_i(z_i^*)$. Let:

$$y'_{i+1} = y^*_{i+1} - \delta \tag{4.17}$$

Substitute (4.16) into (4.5) and (4.6), we get:

$$G(x_i^*, y_{i+1}^*) = \sum_{i=1}^{n}\left[c_i^p a_i - c_i^p a_i (1-p_i)^{x_i^*} - c_i^t x_i^* - \beta T_i(x_i^*)\right] + \sum_{i=1}^{n-1}\left[\beta y_{i+1}^* - (c_{i+1}^h + \beta) H_{i+1}(x_i^*, y_{i+1}^*)\right] \tag{4.18}$$

$$H_{i+1}(x_i^*, y_{i+1}^*) = -a_i b_i u[(1-p_i)^{z_i^*} - (1-p_i)^{x_i^*}] + t_i^r a_i^2 b_i \left[\frac{(1-p_i)^{2z_i^*} + (1-p_i)^{2x_i^*+1}}{2-p_i} - (1-p_i)^{x_i^*+z_i^*}\right]$$
$$+ a_i b_i t_i^s \left[(z_i^* - x_i^*)(1-p_i)^{x_i^*} + \frac{(1-p_i)^{z_i^*} - (1-p_i)^{x_i^*}}{p_i}\right] \tag{4.19}$$

Similarly, substitute (4.17) into (4.5) and (4.6), we get:

$$G(x_i^*, y'_{i+1}) = \sum_{i=1}^{n}\left[c_i^p a_i - c_i^p a_i (1-p_i)^{x_i^*} - c_i^t x_i^* - \beta T_i(x_i^*)\right] + \sum_{i=1}^{n-1}\left[\beta y'_{i+1} - (c_{i+1}^h + \beta) H_{i+1}(x_i^*, y'_{i+1})\right] \tag{4.20}$$

$$H_{i+1}(x_i^*, y'_{i+1}) = -a_i b_i (u+\delta)[(1-p_i)^{z_i^*} - (1-p_i)^{x_i^*}] + t_i^r a_i^2 b_i \left[\frac{(1-p_i)^{2z_i^*} + (1-p_i)^{2x_i^*+1}}{2-p_i} - (1-p_i)^{x_i^*+z_i^*}\right]$$
$$+ a_i b_i t_i^s \left[(z_i^* - x_i^*)(1-p_i)^{x_i^*} + \frac{(1-p_i)^{z_i^*} - (1-p_i)^{x_i^*}}{p_i}\right] \tag{4.21}$$

Where $G(x_i^*, y_{i+1}^*)$ is the corresponding optimal project profit for $x_i = x_i^*$, $y_{i+1} = y_{i+1}^*$; $G(x_i^*, y'_{i+1})$ is the corresponding project profit for $x_i = x_i^*$, $y_{i+1} = y'_{i+1}$.

From (4.18)-(4.21), we get:

$$G(x_i^*, y_{i+1}^*) - G(x_i^*, y'_{i+1}) = \delta\beta - \delta a_i b_i (c_{i+1}^h + \beta)[(1-p_i)^{z_i^*} - (1-p_i)^{x_i^*}] \tag{4.22}$$

Since $G(x_i^*, y_{i+1}^*) > G(x_i^*, y'_{i+1})$, we get:

$$\beta > a_i b_i (c_{i+1}^h + \beta)[(1-p_i)^{z_i^*} - (1-p_i)^{x_i^*}] \tag{4.23}$$

The first derivative of $G(x_i^*, y_{i+1}^*)$ in (4.18) with respect to $u$ is:

$$\frac{\partial G(x_i^*, y_{i+1}^*)}{\partial u} = -\beta - (c_{i+1}^h + \beta)\frac{\partial H_{i+1}(x_i^*, y_{i+1}^*)}{\partial u}$$

$$= -\beta + a_i b_i (c_{i+1}^h + \beta)[(1-p_i)^{z_i^*} - (1-p_i)^{x_i^*}] \tag{4.24}$$

Clearly, if (4.23) holds, then $\frac{\partial G(x_i^*, y_{i+1}^*)}{\partial u} < 0$ for all possible values of $u$. In other words, project profit in (4.18) decreases monotonically in $u$, and so $u = 0$. A conflict arises. Thus, $y_{i+1}^* = T_i(x_i^*) - T_i(z_i^*)$, i.e., it's optimal to start initial design of stage $i+1$ directly after the completion of $z_i^*$ rounds of tests in stage $i$.

**Proof of Proposition 4.2**

**(a)** Given $x_i$, the item in constraint (4.12), $t_i^s(x_i - z_i) + t_i^r a_i[(1-p_i)^{z_i} - (1-p_i)^{x_i}]$, decreases monotonically in $z_i$. Thus, there exists a unique value of $z_i'$ such that: $t_i^s(x_i - z_i') + t_i^r a_i[(1-p_i)^{z_i'} - (1-p_i)^{x_i}] = d_{i+1}$. Let $\lceil z_i' \rceil$ be the minimum integer greater than or equal to $z_i'$. The lower bound of $z_i$ that satisfies constraints (4.11)-(4.13), $z_i^l$, can then be calculated as follows:

$$z_i^l = \max\{0, \lceil z_i' \rceil\} \tag{4.25}$$

The first and second partial derivatives of (4.10) with respect to $z_i$ are:

$$\frac{\partial G}{\partial z_i} = t_i^r a_i^2 b_i (c_{i+1}^h + \beta)\left[(1-p_i)^{x_i} - \frac{2(1-p_i)^{z_i}}{2-p_i} + \omega_i\right](1-p_i)^{z_i}\ln(1-p_i) - t_i^s\beta - t_i^s a_i b_i (c_{i+1}^h + \beta)(1-p_i)^{x_i}$$

(4.26)

$$\frac{\partial^2 G}{\partial z_i^2} = t_i^r a_i^2 b_i (c_{i+1}^h + \beta)\left[(1-p_i)^{x_i} - \frac{4(1-p_i)^{z_i}}{2-p_i} + \omega_i\right](1-p_i)^{z_i}\ln^2(1-p_i) \tag{4.27}$$

Where $\omega_i = \frac{\beta}{a_i b_i (c_{i+1}^h + \beta)} - \frac{t_i^s}{t_i^r a_i p_i}$.

For easier explanation, define: $F_1 = \left.\frac{\partial G}{\partial z_i}\right|_{z_i = z_i^l}$, and $F_2 = \left.\frac{\partial G}{\partial z_i}\right|_{z_i = x_i}$.

It can be verified that given $x_i$, the item in (4.27), $(1-p_i)^{x_i} - \frac{4(1-p_i)^{z_i}}{2-p_i} + \omega_i$, increases monotonically in $z_i$. It follows that:

- If $(1-p_i)^{x_i} - \frac{4(1-p_i)^{z_i^l}}{2-p_i} + \omega_i \geq 0$, then $\frac{\partial^2 G}{\partial z_i^2} \geq 0$ for all possible $z_i$, and $(1-p_i)^{x_i} - \frac{2(1-p_i)^{z_i^l}}{2-p_i} + \omega_i > 0$, leading to $\frac{\partial G}{\partial z_i} < 0$ for all possible $z_i$. In other words, the project profit in (4.10) decreases monotonically in $z_i$, and so $z_i^* = z_i^l$.

- If $\left.\frac{\partial^2 G}{\partial z_i^2}\right|_{z_i = x_i} \leq 0$, i.e., $\omega_i - \frac{2+p_i}{2-p_i}(1-p_i)^{x_i} \leq 0$, then $\frac{\partial^2 G}{\partial z_i^2} < 0 \quad \forall z_i$. It follows that, if $F_1 \leq 0$, then $\frac{\partial G}{\partial z_i} \leq 0 \quad \forall z_i$, and $z_i^* = z_i^l$; if $F_2 \geq 0$, then $\frac{\partial G}{\partial z_i} \geq 0 \quad \forall z_i$, and $z_i^* = x_i^*$; otherwise, setting (4.26) to zero yields a unique $z_i^c$, and $z_i^* = \lceil z_i^c \rceil$ or $z_i^* = \lceil z_i^c \rceil - 1$.

- Otherwise, $\frac{\partial^2 G}{\partial z_i^2}$ increases monotonically from a negative value to a positive value. Note that here if $\left.\frac{\partial^2 G}{\partial z_i^2}\right|_{z_i = x_i} > 0$, then $F_2 < 0$. Thus, $\frac{\partial G}{\partial z_i}$ first decreases to a global minimum value, and then increases monotonically, and finally reaches a negative value. It follows that: if $F_1 \leq 0$, then $\frac{\partial G}{\partial z_i} < 0 \quad \forall z_i$, and $z_i^* = z_i^l$; otherwise if $F_1 > 0$, then setting (4.26) to zero yields a unique $z_i^c$ that maximizes the project profit, and $z_i^* = \lceil z_i^c \rceil$ or $z_i^* = \lceil z_i^c \rceil - 1$.

The above results are summarized in Proposition 4.2(a).

**(b)** Given $x_i$, the mixed partials:

$$\frac{\partial^2 G}{\partial z_i \partial a_i} = t_i^r a_i b_i (c_{i+1}^h + \beta)\left\{\left[2(1-p_i)^{x_i} - \frac{4(1-p_i)^{z_i}}{2-p_i} + \omega_i\right](1-p_i)^{z_i}\ln(1-p_i) - \frac{t_i^s}{t_i^r a_i}(1-p_i)^{x_i}\right\} \quad (4.28)$$

$$\frac{\partial^2 G}{\partial z_i \partial b_i} = t_i^r a_i^2 (c_{i+1}^h + \beta)\left\{\left[(1-p_i)^{x_i} - \frac{2(1-p_i)^{z_i}}{2-p_i} - \frac{t_i^s}{t_i^r a_i p_i}\right](1-p_i)^{z_i}\ln(1-p_i) - \frac{t_i^s}{t_i^r a_i}(1-p_i)^{x_i}\right\} \quad (4.29)$$

When $z_i = z_i^c$, $\dfrac{\partial G}{\partial z_i} = 0$. Equations (4.28) and (4.29) can then be rewritten as:

$$\left.\frac{\partial^2 G}{\partial z_i \partial a_i}\right|_{z_i = z_i^c} = \frac{\beta t_i^s}{a_i} + t_i^r a_i b_i (c_{i+1}^h + \beta)\left[(1-p_i)^{x_i} - \frac{2(1-p_i)^{z_i^c}}{2-p_i}\right](1-p_i)^{z_i^c}\ln(1-p_i) \quad (4.30)$$

$$\left.\frac{\partial^2 G}{\partial z_i \partial b_i}\right|_{z_i = z_i^c} = \frac{\beta}{b_i}\left[t_i^s - t_i^r a_i (1-p_i)^{z_i^c}\ln(1-p_i)\right] \quad (4.31)$$

Since the maximum value of $z_i^c$ is $x_i$, $(1-p_i)^{x_i} - \dfrac{2(1-p_i)^{z_i^c}}{2-p_i} < 0 \;\; \forall z_i^c$. It follows that $\left.\dfrac{\partial^2 G}{\partial z_i \partial a_i}\right|_{z_i = z_i^c} \geq 0$. Clearly, $\left.\dfrac{\partial^2 G}{\partial z_i \partial b_i}\right|_{z_i = z_i^c} \geq 0$, and $\left.\dfrac{\partial^2 G}{\partial z_i^2}\right|_{z_i = z_i^c} \leq 0$. By the implicit function theorem $\dfrac{\partial z_i^c}{\partial a_i} = -\left.\dfrac{\partial^2 G}{\partial z_i \partial a_i}\right|_{z_i = z_i^c} \Big/ \left.\dfrac{\partial^2 G}{\partial z_i^2}\right|_{z_i = z_i^c}$, $\dfrac{\partial z_i^c}{\partial b_i} = -\left.\dfrac{\partial^2 G}{\partial z_i \partial b_i}\right|_{z_i = z_i^c} \Big/ \left.\dfrac{\partial^2 G}{\partial z_i^2}\right|_{z_i = z_i^c}$, we get $\dfrac{\partial z_i^c}{\partial a_i} \geq 0$, and $\dfrac{\partial z_i^c}{\partial b_i} \geq 0$.

Similarly, the mixed partials:

$$\left.\frac{\partial^2 G}{\partial z_i \partial t_i^s}\right|_{z_i = z_i^c} = -a_i b_i (c_{i+1}^h + \beta)\left[\frac{\beta}{a_i b_i (c_{i+1}^h + \beta)} + (1-p_i)^{x_i} + \frac{(1-p_i)^{z_i^c}\ln(1-p_i)}{p_i}\right] \quad (4.32)$$

$$\left.\frac{\partial^2 G}{\partial z_i \partial t_i^r}\right|_{z_i = z_i^c} = \frac{t_i^s a_i b_i (c_{i+1}^h + \beta)}{t_i^r}\left[\frac{\beta}{a_i b_i (c_{i+1}^h + \beta)} + (1-p_i)^{x_i} + \frac{(1-p_i)^{z_i^c}\ln(1-p_i)}{p_i}\right] \quad (4.33)$$

Since $\left.\frac{\partial G}{\partial z_i}\right|_{z_i=z_i^c}=0$, we get:

$$\frac{p_i}{(1-p_i)^{z_i^c}\ln(1-p_i)}\left[\frac{\beta}{a_ib_i(c_{i+1}^h+\beta)}+(1-p_i)^{x_i}\right]=-\frac{t_i^s+t_i^ra_i\frac{2p_i}{2-p_i}(1-p_i)^{z_i^c}}{t_i^s-t_i^ra_i(1-p_i)^{z_i^c}\ln(1-p_i)} \tag{4.34}$$

It can be verified that, $\frac{2p_i}{2-p_i}+\ln(1-p_i)\le 0\ \ \forall p_i$. It follows that:

$$\frac{p_i}{(1-p_i)^{z_i^c}\ln(1-p_i)}\left[\frac{\beta}{a_ib_i(c_{i+1}^h+\beta)}+(1-p_i)^{x_i}\right]\ge -1 \tag{4.35}$$

It follows that $\left.\frac{\partial^2 G}{\partial z_i\partial t_i^s}\right|_{z_i=z_i^c}\ge 0$, $\left.\frac{\partial^2 G}{\partial z_i\partial t_i^r}\right|_{z_i=z_i^c}\le 0$. Since $\left.\frac{\partial^2 G}{\partial z_i^2}\right|_{z_i=z_i^c}\le 0$, by the implicit function theorem $\frac{\partial z_i^c}{\partial t_i^s}=-\left.\frac{\partial^2 G}{\partial z_i\partial t_i^s}\right|_{z_i=z_i^c}\Big/\left.\frac{\partial^2 G}{\partial z_i^2}\right|_{z_i=z_i^c}$, $\frac{\partial z_i^c}{\partial t_i^r}=-\left.\frac{\partial^2 G}{\partial z_i\partial t_i^r}\right|_{z_i=z_i^c}\Big/\left.\frac{\partial^2 G}{\partial z_i^2}\right|_{z_i=z_i^c}$, we get $\frac{\partial z_i^c}{\partial t_i^s}\ge 0$, and $\frac{\partial z_i^c}{\partial t_i^r}\le 0$.

**Proof of Proposition 4.3**

**(a)** Given $z_i$, the left side of (4.12), $t_i^s(x_i-z_i)+t_i^ra_i[(1-p_i)^{z_i}-(1-p_i)^{x_i}]$, increases monotonically in $x_i$. Thus, there exists a unique value of $x_i'$ such that: $t_i^s(x_i'-z_i)+t_i^ra_i[(1-p_i)^{z_i}-(1-p_i)^{x_i'}]=d_{i+1}$. Defining $x_i^u$ be the upper bound of $x_i$ that satisfies constraint (4.12), mathematically, $x_i^u=\lceil x_i'\rceil-1$.

The first and second partial derivatives of (4.10) with respect to $x_i$ are:

$$\frac{\partial G}{\partial x_i}=t_i^ra_i^2b_i(c_{i+1}^h+\beta)\left[(1-p_i)^{z_i}+\frac{t_i^s}{t_i^ra_i}(x_i-z_i)-\frac{2(1-p_i)^{x_i+1}}{2-p_i}+\theta_i\right](1-p_i)^{x_i}\ln(1-p_i)-c_i^t \tag{4.36}$$

$$\frac{\partial^2 G}{\partial x_i^2}=t_i^ra_i^2b_i(c_{i+1}^h+\beta)\left\{(1-p_i)^{z_i}+\frac{t_i^s}{t_i^ra_i}\left[x_i-z_i+\frac{1}{\ln(1-p_i)}\right]-\frac{4(1-p_i)^{x_i+1}}{2-p_i}+\theta_i\right\}(1-p_i)^{x_i}\ln^2(1-p_i) \tag{4.37}$$

Where $\theta_i = \frac{t_i^s}{t_i^r a_i}\left[\frac{1}{p_i} + \frac{1}{\ln(1-p_i)}\right] - \frac{c_i^p}{t_i^r a_i b_i (c_{i+1}^h + \beta)}$.

Let $\Phi(x_i) = (1-p_i)^{z_i} + \frac{t_i^s}{t_i^r a_i}\left[x_i - z_i + \frac{1}{\ln(1-p_i)}\right] - \frac{4(1-p_i)^{x_i+1}}{2-p_i} + \theta_i$, $F_3 = \left.\frac{\partial G}{\partial x_i}\right|_{x_i = z_i}$,

and $F_4 = \left.\frac{\partial G}{\partial x_i}\right|_{x_i = x_i^u}$.

It can be clear that given $z_i$, $\Phi(x_i)$ increases monotonically in $x_i$. It follows that:

- If $\Phi(z_i) \geq 0$, i.e. $\frac{3p_i - 2}{2-p_i}(1-p_i)^{z_i} + \frac{t_i^s}{t_i^r a_i \ln(1-p_i)} + \theta_i \geq 0$, then $\frac{\partial^2 G}{\partial x_i^2} \geq 0 \;\; \forall x_i$,

and $\frac{\partial G}{\partial x_i} < 0 \;\; \forall x_i$. Thus, $x_i^* = z_i^*$.

- If $\Phi(x_i^u) \leq 0$, i.e., $(1-p_i)^{z_i} + \frac{t_i^s}{t_i^r a_i}\left[x_i^u - z_i + \frac{1}{\ln(1-p_i)}\right] - \frac{4(1-p_i)^{x_i^u+1}}{2-p_i} + \theta_i \leq 0$,

then $\frac{\partial^2 G}{\partial x_i^2} \leq 0 \;\; \forall z_i$. It follows that: if $F_3 \leq 0$, then $\frac{\partial G}{\partial x_i} \leq 0 \;\; \forall x_i$, and so $x_i^* = z_i^*$; if $F_4 \geq 0$, then $\frac{\partial G}{\partial x_i} \geq 0 \;\; \forall x_i$, resulting in $x_i^* = x_i^u$; otherwise, setting (4.36) to zero yields a unique $x_i^c$, and $x_i^* = \lceil x_i^c \rceil$ or $x_i^* = \lceil x_i^c \rceil - 1$.

- Otherwise, $\frac{\partial^2 G}{\partial x_i^2}$ increases monotonically from a negative value to a positive value. Note that here if $\Phi(x_i^u) \geq 0$, then $F_4 < 0$. Thus, $\frac{\partial G}{\partial x_i}$ first decreases to a global minimum value, and then increases monotonically, and finally reaches a negative value. It follows that: if $F_3 \leq 0$, then $\frac{\partial G}{\partial x_i} \leq 0 \;\; \forall x_i$, and so $x_i^* = z_i^*$; otherwise if $F_3 > 0$, then setting (4.36) to zero yields a unique $x_i^c$, and $x_i^* = \lceil x_i^c \rceil$ or $x_i^* = \lceil x_i^c \rceil - 1$.

The above results are summarized in Proposition 4.3(a).

**(b)** Given $z_i$, the mixed partial:

$$\left.\frac{\partial^2 G}{\partial x_i \partial t_i^r}\right|_{x_i=x_i^c} = a_i^2 b_i (c_{i+1}^h + \beta)\left[(1-p_i)^{z_i} - \frac{2(1-p_i)^{x_i^c+1}}{2-p_i}\right](1-p_i)^{x_i^c}\ln(1-p_i) \qquad (4.38)$$

Since the minimum value of $x_i^c$ is $z_i$, we get $(1-p_i)^{z_i} - \frac{2(1-p_i)^{x_i^c+1}}{2-p_i} \ge 0 \ \forall x_i^c$. It follows that $\left.\frac{\partial^2 G}{\partial x_i \partial t_i^r}\right|_{x_i=x_i^c} \le 0$. Since $\left.\frac{\partial^2 G}{\partial x_i^2}\right|_{x_i=x_i^c} \le 0$, by the implicit function theorem, $\frac{\partial x_i^c}{\partial t_i^r} = -\left.\frac{\partial^2 G}{\partial x_i \partial t_i^r}\right|_{x_i=x_i^c} \Big/ \left.\frac{\partial^2 G}{\partial x_i^2}\right|_{x_i=x_i^c}$, we get $\frac{\partial x_i^c}{\partial t_i^r} \le 0$.

Similarly, given $z_i$, the mixed partials:

$$\left.\frac{\partial^2 G}{\partial x_i \partial t_i^s}\right|_{x_i=x_i^c} = a_i b_i (c_{i+1}^h + \beta)\left[x_i^c - z_i + \frac{1}{p_i} + \frac{1}{\ln(1-p_i)}\right](1-p_i)^{x_i^c}\ln(1-p_i) \qquad (4.39)$$

$$\left.\frac{\partial^2 G}{\partial x_i \partial b_i}\right|_{x_i=x_i^c} = t_i^r a_i^2 (c_{i+1}^h + \beta)\left\{(1-p_i)^{z_i} + \frac{t_i^s}{t_i^r a_i}\left[x_i^c - z_i + \frac{1}{p_i} + \frac{1}{\ln(1-p_i)}\right] - \frac{2(1-p_i)^{x_i^c+1}}{2-p_i}\right\}$$
$$\times (1-p_i)^{x_i^c}\ln(1-p_i) \qquad (4.40)$$

Clearly, $\frac{1}{p_i} + \frac{1}{\ln(1-p_i)} \ge 0$ for all positive $p_i$. It follows that $\left.\frac{\partial^2 G}{\partial x_i \partial t_i^s}\right|_{x_i=x_i^c} \le 0$, and $\left.\frac{\partial^2 G}{\partial x_i \partial b_i}\right|_{x_i=x_i^c} \le 0$. Since $\left.\frac{\partial^2 G}{\partial x_i^2}\right|_{x_i=x_i^c} \le 0$, by the implicit function theorem, $\frac{\partial x_i^c}{\partial t_i^s} = -\left.\frac{\partial^2 G}{\partial x_i \partial t_i^s}\right|_{x_i=x_i^c} \Big/ \left.\frac{\partial^2 G}{\partial x_i^2}\right|_{x_i=x_i^c}$, $\frac{\partial x_i^c}{\partial b_i} = -\left.\frac{\partial^2 G}{\partial x_i \partial b_i}\right|_{x_i=x_i^c} \Big/ \left.\frac{\partial^2 G}{\partial x_i^2}\right|_{x_i=x_i^c}$, we can conclude $\frac{\partial x_i^c}{\partial t_i^s} \le 0$, $\frac{\partial x_i^c}{\partial b_i} \le 0$.

**(c)** Given $z_i$, the mixed partial:

$$\left.\frac{\partial^2 G}{\partial x_i \partial a_i}\right|_{x_i=x_i^c} = t_i^r a_i b_i (c_{i+1}^h + \beta)\left\{2(1-p_i)^{z_i} + \frac{t_i^s}{t_i^r a_i}(x_i - z_i) - \frac{4(1-p_i)^{x_i^c+1}}{2-p_i} + \theta_i\right\}(1-p_i)^{x_i^c}\ln(1-p_i) \tag{4.41}$$

Let $\bar{a}_i = \frac{2-p_i}{2p_i(1-p_i)^{z_i}}\left\{\frac{c_i^p}{t_i^r b_i(c_{i+1}^h+\beta)} - \frac{t_i^s}{t_i^r}\left[\frac{1}{p_i} + \frac{1}{\ln(1-p_i)}\right]\right\}$. It can be verified that, $\left\{2(1-p_i)^{z_i} + \frac{t_i^s}{t_i^r a_i}(x_i - z_i) - \frac{4(1-p_i)^{x_i+1}}{2-p_i} + \theta_i\right\}\ln(1-p_i)$, decreases monotonically in $x_i$. It follows that: if $a_i \geq \bar{a}_i$, then $\left.\frac{\partial^2 G}{\partial x_i \partial a_i}\right|_{x_i=x_i^c} \leq 0 \quad \forall x_i^c$, and so by the implicit function theorem, $\frac{\partial x_i^c}{\partial a_i} = -\left.\frac{\partial^2 G}{\partial x_i \partial a_i}\right|_{x_i=x_i^c} \Big/ \left.\frac{\partial^2 G}{\partial x_i^2}\right|_{x_i=x_i^c}$, we get $\frac{\partial x_i^c}{\partial a_i} \leq 0$; if $a_i < \bar{a}_i$, then setting (4.41) to zero yields a unique $\bar{x}_i$ such that: if $x_i^c < \bar{x}_i$, $\frac{\partial x_i^c}{\partial a_i} \geq 0$; if $x_i^c > \bar{x}_i$, $\frac{\partial x_i^c}{\partial a_i} \leq 0$.

**Proof of Proposition 4.4**

**(a)** It is clear that the cost of conducting $x_i$ rounds of tests is larger than $c_i^t x_i$. Inspection of (4.2) shows that the maximum gain from conducting tests is no more than: $c_i^p a_i$. To maximize project profit, the testing costs should be less than the potential benefit. Therefore, $x_i^* < \frac{c_i^p a_i}{c_i^t}$.

**(b)** The first and second derivatives of (4.10) with respect to $x_n$ are:

$$\frac{dG}{dx_n} = a_n(t_n^r\beta - c_n^p)(1-p_n)^{x_n}\ln(1-p_n) - c_n^t - t_n^s\beta \tag{4.42}$$

$$\frac{d^2G}{dx_n^2} = a_n(t_n^r\beta - c_n^p)(1-p_n)^{x_n}\ln^2(1-p_n) \tag{4.43}$$

Inspection of (4.42) and (4.43) shows that:

- If $t_n^r\beta - c_n^p \geq 0$, then $\frac{dG}{dx_n} \leq 0 \ \forall x_n$, leading to $x_n^* = 0$.
- Otherwise if $t_n^r\beta - c_n^p < 0$, then $\frac{d^2G}{dx_n^2} \leq 0 \ \forall x_n$. It follows that: if $\left.\frac{dG}{dx_n}\right|_{x_n=0} \leq 0$,

i.e. $c_n^t + t_n^s\beta \geq a_n(t_n^r\beta - c_n^p)\ln(1-p_n)$, then $\frac{dG}{dx_n} \leq 0 \quad \forall x_n$, leading to $x_n^* = 0$; otherwise if $c_n^t + t_n^s\beta < a_n(t_n^r\beta - c_n^p)\ln(1-p_n)$, then setting (4.42) to zero yields a unique $x_n^c$ that maximizes the project profit, and so $x_n^* = \lceil x_n^c \rceil$ or $x_n^* = \lceil x_n^c \rceil - 1$.

The above results are summarized in Proposition 4.4(b).

# APPENDIX C

# PROOFS OF CHAPTER 5

## Proof of Theorem 5.1

| Activities | 1 | 2 | … | $i-1$ | $i$ | $i+1$ | $i+2$ | … | $n$ |
|---|---|---|---|---|---|---|---|---|---|
| 1 | ■ | $d_{1,2}$ | … | $d_{1,i-1}$ | $d_{1,i}$ | $d_{1,i+1}$ | $d_{1,i+2}$ | … | $d_{1,n}$ |
| 2 | $d_{2,1}$ | ■ | … | $d_{2,i-1}$ | $d_{2,i}$ | $d_{2,i+1}$ | $d_{2,i+2}$ | … | $d_{2,n}$ |
| … | … | … | ■ | … | … | … | … | … | … |
| $i-1$ | $d_{i-1,1}$ | $d_{i-1,2}$ | … | ■ | $d_{i-1,i}$ | $d_{i-1,i+1}$ | $d_{i-1,i+2}$ | … | $d_{i-1,n}$ |
| $i$ | $d_{i,1}$ | $d_{i,2}$ | … | $d_{i,i-1}$ | ■ | $d_{i,i+1}$ | $d_{i,i+2}$ | … | $d_{i,n}$ |
| $i+1$ | $d_{i+1,1}$ | $d_{i+1,2}$ | … | $d_{i+1,i-1}$ | $d_{i+1,i}$ | ■ | $d_{i+1,i+2}$ | … | $d_{i+1,n}$ |
| $i+2$ | $d_{i+2,1}$ | $d_{i+2,2}$ | … | $d_{i+2,i-1}$ | $d_{i+2,i}$ | $d_{i+2,i+1}$ | ■ | … | $d_{i+2,n}$ |
| … | … | … | … | … | … | … | … | ■ | … |
| $n$ | $d_{n,1}$ | $d_{n,2}$ | … | $d_{n,i-1}$ | $d_{n,i}$ | $d_{n,i+1}$ | $d_{n,i+2}$ | … | ■ |

**(a)** Original NDSM

| Activities | 1 | 2 | … | $i-1$ | $i+1$ | $i$ | $i+2$ | … | $n$ |
|---|---|---|---|---|---|---|---|---|---|
| 1 | ■ | $d_{1,2}$ | … | $d_{1,i-1}$ | $d_{1,i+1}$ | $d_{1,i}$ | $d_{1,i+2}$ | … | $d_{1,n}$ |
| 2 | $d_{2,1}$ | ■ | … | $d_{2,i-1}$ | $d_{2,i+1}$ | $d_{2,i}$ | $d_{2,i+2}$ | … | $d_{2,n}$ |
| … | … | … | ■ | … | … | … | … | … | … |
| $i-1$ | $d_{i-1,1}$ | $d_{i-1,2}$ | … | ■ | $d_{i-1,i+1}$ | $d_{i-1,i}$ | $d_{i-1,i+2}$ | … | $d_{i-1,n}$ |
| $i+1$ | $d_{i+1,1}$ | $d_{i+1,2}$ | … | $d_{i+1,i-1}$ | ■ | $d_{i+1,i}$ | $d_{i+1,i+2}$ | … | $d_{i+1,n}$ |
| $i$ | $d_{i,1}$ | $d_{i,2}$ | … | $d_{i,i-1}$ | $d_{i,i+1}$ | ■ | $d_{i,i+2}$ | … | $d_{i,n}$ |
| $i+2$ | $d_{i+2,1}$ | $d_{i+2,2}$ | … | $d_{i+2,i-1}$ | $d_{i+2,i+1}$ | $d_{i+2,i}$ | ■ | … | $d_{i+2,n}$ |
| … | … | … | … | … | … | … | … | ■ | … |
| $n$ | $d_{n,1}$ | $d_{n,2}$ | … | $d_{n,i-1}$ | $d_{n,i+1}$ | $d_{n,i}$ | $d_{n,i+2}$ | … | ■ |

**(b)** The resulting NDSM after exchanging activity *i* and activity *i*+1

**Figure C.1 The scenario used in the proof of Theorem 5.1**

Figure C.1(a) shows the original NDSM where activities are executed in the order of $\{1,2,...,i-1,i,i+1,...,n\}$. Here the total feedbacks is: $\sum_{k=1}^{n}\sum_{j=k+1}^{n} d_{k,j}$, which can be written as: $\sum_{k=1}^{i-1}\sum_{j=k+1}^{n} d_{k,j} + \sum_{k=i+2}^{n}\sum_{j=k+1}^{n} d_{k,j} + \sum_{j=i+2}^{n}\left(d_{i,j}+d_{i+1,j}\right) + d_{i,i+1}$.

Figure C.1(b) shows the resulting NDSM after exchanging activity $i$ and activity $i+1$, where the total feedbacks can be expressed as: $\sum_{k=1}^{i-1}\sum_{j=k+1}^{n} d_{k,j} + \sum_{k=i+2}^{n}\sum_{j=k+1}^{n} d_{k,j} + \sum_{j=i+2}^{n}\left(d_{i,j}+d_{i+1,j}\right) + d_{i+1,i}$. After cancellation, we are left with the terms of $d_{i,i+1}$ and $d_{i+1,i}$. Hence, for an initial activity sequence of $\{1,2,...,i-1,i,i+1,...,n\}$, if $d_{i,i+1} > d_{i+1,i}$, then through exchanging activity $i$ and activity $i+1$, the total feedbacks can be reduced by $\left(d_{i,i+1}-d_{i+1,i}\right)$.

**Proof of Theorem 5.2**

Figure C.2(a) shows the original NDSM where the activities are executed in the order of $\{1,2,...,j-1,j,j+1,...,i-1,i,i+1,...,n\}$. Here the total feedbacks can be written as:

$$\sum_{k=1}^{j-1}\sum_{h=k+1}^{n} d_{k,h} + \sum_{k=i+1}^{n}\sum_{h=k+1}^{n} d_{k,h} + \sum_{k=j+1}^{i-2}\sum_{h=k+1}^{i-1} d_{k,h} + \sum_{k=j}^{i}\sum_{h=i+1}^{n} d_{k,h} + \sum_{k=j}^{i-1} d_{k,i} + \sum_{k=j+1}^{i-1} d_{j,k} \tag{5.14}$$

Figure C.2(b) shows the resulting NDSM after exchanging the positions of activity $j$ and activity $i$, where the total feedbacks can be written as:

$$\sum_{k=1}^{j-1}\sum_{h=k+1}^{n} d_{k,h} + \sum_{k=i+1}^{n}\sum_{h=k+1}^{n} d_{k,h} + \sum_{k=j+1}^{i-2}\sum_{h=k+1}^{i-1} d_{k,h} + \sum_{h=i+1}^{n}\left(\sum_{k=j+1}^{i-1} d_{k,h} + d_{i,h} + d_{j,h}\right) + \sum_{k=j+1}^{i} d_{k,j} + \sum_{k=j+1}^{i-1} d_{i,k} \tag{5.15}$$

Subtracting (5.15) from (5.14), we get $\theta_{j,i} = \sum_{k=j}^{i-1} d_{k,i} + \sum_{k=j+1}^{i-1} d_{j,k} - \sum_{k=j+1}^{i} d_{k,j} - \sum_{k=j+1}^{i-1} d_{i,k}$.

Thus, for an activity sequence of $\{1, 2, \dots, j-1, j, j+1, \dots, i-1, i, i+1, \dots, n\}$, if $\theta_{j,i} > 0$, then through exchanging activity $j$ and activity $i$, the resulting feedbacks can be reduced by $\theta_{j,i}$.

| Activities | 1 | 2 | … | $j-1$ | $j$ | $j+1$ | … | $i-1$ | $i$ | $i+1$ | … | $n$ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | ■ | $d_{1,2}$ | … | $d_{1,j-1}$ | $d_{1,j}$ | $d_{1,j+1}$ | … | $d_{1,i-1}$ | $d_{1,i}$ | $d_{1,i+1}$ | … | $d_{1,n}$ |
| 2 | $d_{2,1}$ | ■ | … | $d_{2,j-1}$ | $d_{2,j}$ | $d_{2,j+1}$ | … | $d_{2,i-1}$ | $d_{2,i}$ | $d_{2,i+1}$ | … | $d_{2,n}$ |
| … | … | … | ■ | … | … | … | … | … | … | … | … | … |
| $j-1$ | $d_{j-1,1}$ | $d_{j-1,2}$ | … | ■ | $d_{j-1,j}$ | $d_{j-1,j+1}$ | … | $d_{j-1,i-1}$ | $d_{j-1,i}$ | $d_{j-1,i+1}$ | … | $d_{j-1,n}$ |
| $j$ | $d_{j,1}$ | $d_{j,2}$ | … | $d_{j,j-1}$ | ■ | $d_{j,j+1}$ | … | $d_{j,i-1}$ | $d_{j,i}$ | $d_{j,i+1}$ | … | $d_{j,n}$ |
| $j+1$ | $d_{j+1,1}$ | $d_{j+1,2}$ | … | $d_{j+1,j-1}$ | $d_{j+1,j}$ | ■ | … | $d_{j+1,i-1}$ | $d_{j+1,i}$ | $d_{j+1,i+1}$ | … | $d_{j+1,n}$ |
| … | … | … | … | … | … | … | ■ | … | … | … | … | … |
| $i-1$ | $d_{i-1,1}$ | $d_{i-1,2}$ | … | $d_{i-1,j-1}$ | $d_{i-1,j}$ | $d_{i-1,j+1}$ | … | ■ | $d_{i-1,i}$ | $d_{i-1,i+1}$ | … | $d_{i-1,n}$ |
| $i$ | $d_{i,1}$ | $d_{i,2}$ | … | $d_{i,j-1}$ | $d_{i,j}$ | $d_{i,j+1}$ | … | $d_{i,i-1}$ | ■ | $d_{i,i+1}$ | … | $d_{i,n}$ |
| $i+1$ | $d_{i+1,1}$ | $d_{i+1,2}$ | … | $d_{i+1,j-1}$ | $d_{i+1,j}$ | $d_{i+1,j+1}$ | … | $d_{i+1,i-1}$ | $d_{i+1,i}$ | ■ | … | $d_{i+1,n}$ |
| … | … | … | … | … | … | … | … | … | … | … | ■ | … |
| $n$ | $d_{n,1}$ | $d_{n,2}$ | … | $d_{n,j-1}$ | $d_{n,j}$ | $d_{n,j+1}$ | … | $d_{n,i-1}$ | $d_{n,i}$ | $d_{n,i+1}$ | … | ■ |

**(a)** Original NDSM

| Activities | 1 | 2 | … | $j-1$ | $i$ | $j+1$ | … | $i-1$ | $j$ | $i+1$ | … | $n$ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | ■ | $d_{1,2}$ | … | $d_{1,j-1}$ | $d_{1,i}$ | $d_{1,j+1}$ | … | $d_{1,i-1}$ | $d_{1,j}$ | $d_{1,i+1}$ | … | $d_{1,n}$ |
| 2 | $d_{2,1}$ | ■ | … | $d_{2,j-1}$ | $d_{2,i}$ | $d_{2,j+1}$ | … | $d_{2,i-1}$ | $d_{2,j}$ | $d_{2,i+1}$ | … | $d_{2,n}$ |
| … | … | … | ■ | … | … | … | … | … | … | … | … | … |
| $j-1$ | $d_{j-1,1}$ | $d_{j-1,2}$ | … | ■ | $d_{j-1,i}$ | $d_{j-1,j+1}$ | … | $d_{j-1,i-1}$ | $d_{j-1,j}$ | $d_{j-1,i+1}$ | … | $d_{j-1,n}$ |
| $i$ | $d_{i,1}$ | $d_{i,2}$ | … | $d_{i,j-1}$ | ■ | $d_{i,j+1}$ | … | $d_{i,i-1}$ | $d_{i,j}$ | $d_{i,i+1}$ | … | $d_{i,n}$ |
| $j+1$ | $d_{j+1,1}$ | $d_{j+1,2}$ | … | $d_{j+1,j-1}$ | $d_{j+1,i}$ | ■ | … | $d_{j+1,i-1}$ | $d_{j+1,j}$ | $d_{j+1,i+1}$ | … | $d_{j+1,n}$ |
| … | … | … | … | … | … | … | ■ | … | … | … | … | … |
| $i-1$ | $d_{i-1,1}$ | $d_{i-1,2}$ | … | $d_{i-1,j-1}$ | $d_{i-1,i}$ | $d_{i-1,j+1}$ | … | ■ | $d_{i-1,j}$ | $d_{i-1,i+1}$ | … | $d_{i-1,n}$ |
| $j$ | $d_{j,1}$ | $d_{j,2}$ | … | $d_{j,j-1}$ | $d_{j,i}$ | $d_{j,j+1}$ | … | $d_{j,i-1}$ | ■ | $d_{j,i+1}$ | … | $d_{j,n}$ |
| $i+1$ | $d_{i+1,1}$ | $d_{i+1,2}$ | … | $d_{i+1,j-1}$ | $d_{i+1,i}$ | $d_{i+1,j+1}$ | … | $d_{i+1,i-1}$ | $d_{i+1,j}$ | ■ | … | $d_{i+1,n}$ |
| … | … | … | … | … | … | … | … | … | … | … | ■ | … |
| $n$ | $d_{n,1}$ | $d_{n,2}$ | … | $d_{n,j-1}$ | $d_{n,i}$ | $d_{n,j+1}$ | … | $d_{n,i-1}$ | $d_{n,j}$ | $d_{n,i+1}$ | … | ■ |

**(b)** The resulting NDSM after exchanging activity *j* and activity *i*

**Figure C.2 The scenario used in the proof of Theorem 5.2**

To make our statement more legible, in Figure C.2, we highlight in blue for the different items between (5.14) and (5.15).

**Proof of Theorem 5.3**

Defining:

$$z_{i,j} = x_{im} \times \sum_{h=m+1}^{n} x_{jh}\ ,\ \ \forall m \tag{5.16}$$

Since $x_{im}$ and $\sum_{h=m+1}^{n} x_{jh}$ are all 0-1 binary variables, $z_{i,j}$ is also a binary variable. Inspection of (5.16) shows that if either $x_{im} = 0$ or $\sum_{h=m+1}^{n} x_{jh} = 0$, then $z_{i,j} = 0$. It follows that (5.16) can be rewritten as:

$$z_{i,j} = \max\left[0, x_{im} - 1 + \sum_{h=m+1}^{n} x_{jh}\right],\ \ \forall m \tag{5.17}$$

Where $z_{i,j} \in \{0,1\}$.

Clearly, (5.17) is equivalent to:

$$z_{i,j} \geq x_{im} - 1 + \sum_{h=m+1}^{n} x_{jh}\ ,\ \ \forall m \tag{5.18}$$

Since $\sum_{h=1}^{n} x_{jh} = 1$, (5.18) can be rewritten as:

$$z_{i,j} \geq x_{im} - \sum_{h=1}^{m-1} x_{jh}\ ,\ \ \forall m \tag{5.7}$$

Substituting (5.16) into (5.1), and adding constraints (5.7)-(5.8), we get the 0-1 LIP.

**Proof of Theorem 5.4**

By (5.9), we have:

$$d(\lambda) = \min \sum_{i=1}^{n} \sum_{\substack{j=1 \\ j \neq i}}^{n} \left[ d_{i,j} z_{i,j} + \sum_{m=1}^{n} \lambda_{i,j}^{m} \left( x_{im} - \sum_{h=1}^{m-1} x_{jh} - z_{i,j} \right) \right]$$

$$= \min \left[ \sum_{i=1}^{n} \sum_{\substack{j=1 \\ j \neq i}}^{n} \left( d_{i,j} - \sum_{m=1}^{n} \lambda_{i,j}^{m} \right) z_{i,j} + \sum_{i=1}^{n} \sum_{\substack{j=1 \\ j \neq i}}^{n} \sum_{m=1}^{n} \lambda_{i,j}^{m} \left( x_{im} - \sum_{h=1}^{m-1} x_{jh} \right) \right]$$

$$= \min \left[ \sum_{i=1}^{n} \sum_{\substack{j=1 \\ j \neq i}}^{n} \left( d_{i,j} - \sum_{m=1}^{n} \lambda_{i,j}^{m} \right) z_{i,j} + \sum_{i=1}^{n} \sum_{m=1}^{n} \left( \sum_{\substack{j=1 \\ j \neq i}}^{n} \lambda_{i,j}^{m} - \sum_{\substack{j=1 \\ j \neq i}}^{n} \sum_{h=m+1}^{n} \lambda_{j,i}^{h} \right) x_{im} \right] \tag{5.19}$$

For given non-negative Lagrangian multipliers that satisfy: $\sum_{m=1}^{n} \lambda_{i,j}^{m} \leq d_{i,j}$, equation (5.19) can be reduced to:

$$d(\lambda) = \min \sum_{i=1}^{n} \sum_{m=1}^{n} \left( \sum_{\substack{j=1 \\ j \neq i}}^{n} \lambda_{i,j}^{m} - \sum_{\substack{j=1 \\ j \neq i}}^{n} \sum_{h=m+1}^{n} \lambda_{j,i}^{h} \right) x_{im} \tag{5.10}$$

This proves Theorem 5.4.

# APPENDIX D

# PROOFS OF CHAPTER 6

## Proof of Theorem 6.1

Figure D.1(a) shows the original $n \times n$ NDSM where activities are executed in the order of $\{1, ..., j-1, j, ..., i-1, i, i+1, ..., n\}$. Separate the NDSM into four blocks and define:

$$B_1 = \begin{array}{|c|c|c|}\hline 0 & \cdots & d_{1,j-1} \\ \hline \cdots & 0 & \cdots \\ \hline d_{j-1,1} & \cdots & 0 \\ \hline \end{array} \quad B_J = \begin{array}{|c|c|c|}\hline 0 & \cdots & d_{j,i-1} \\ \hline \cdots & 0 & \cdots \\ \hline d_{i-1,j} & \cdots & 0 \\ \hline \end{array} \quad B_3 = \begin{array}{|c|}\hline 0 \\ \hline \end{array} \quad B_4 = \begin{array}{|c|c|c|}\hline 0 & \cdots & d_{i+1,n} \\ \hline \cdots & 0 & \cdots \\ \hline d_{n,i+1} & \cdots & 0 \\ \hline \end{array}$$

$$B_{1,J} = \begin{bmatrix} d_{1,j} & ... & d_{1,i-1} \\ ... & ... & ... \\ d_{j-1,j} & ... & d_{j-1,i-1} \end{bmatrix} \quad B_{1,3} = \begin{bmatrix} d_{1,i} \\ ... \\ d_{j-1,i} \end{bmatrix} \quad B_{1,4} = \begin{bmatrix} d_{1,i+1} & ... & d_{1,n} \\ ... & ... & ... \\ d_{j-1,i+1} & ... & d_{j-1,n} \end{bmatrix}$$

$$B_{J,1} = \begin{bmatrix} d_{j,1} & ... & d_{j,j-1} \\ ... & ... & ... \\ d_{i-1,1} & ... & d_{i-1,j-1} \end{bmatrix} \quad B_{J,3} = \begin{bmatrix} d_{j,i} \\ ... \\ d_{i-1,i} \end{bmatrix} \quad B_{J,4} = \begin{bmatrix} d_{j,i+1} & ... & d_{j,n} \\ ... & ... & ... \\ d_{i-1,i+1} & ... & d_{i-1,n} \end{bmatrix}$$

$$B_{3,1} = [d_{i,1} \quad ... \quad d_{i,j-1}] \qquad B_{3,J} = [d_{i,j} \quad ... \quad d_{i,i-1}] \qquad B_{3,4} = [d_{i,i+1} \quad ... \quad d_{i,n}]$$

$$B_{4,1} = \begin{bmatrix} d_{i+1,1} & ... & d_{i+1,j-1} \\ ... & ... & ... \\ d_{n,1} & ... & d_{n,j-1} \end{bmatrix} \quad B_{4,J} = \begin{bmatrix} d_{i+1,j} & ... & d_{i+1,i-1} \\ ... & ... & ... \\ d_{n,j} & ... & d_{n,i-1} \end{bmatrix} \quad B_{4,3} = \begin{bmatrix} d_{i+1,i} \\ ... \\ d_{n,i} \end{bmatrix}$$

Figure D.1(b) shows the resulting NDSM after exchanging block $B_J$ and activity $i$. Based on above definitions, the NDSMs in Figure D.1(a) and Figure D.1(b) can be expressed as following (6.4) and (6.5), respectively:

$$\begin{bmatrix} B_1 & B_{1,J} & B_{1,3} & B_{1,4} \\ B_{J,1} & B_J & B_{J,3} & B_{J,4} \\ B_{3,1} & B_{3,J} & B_3 & B_{3,4} \\ B_{4,1} & B_{4,J} & B_{4,3} & B_4 \end{bmatrix} \tag{6.4}$$

$$\begin{bmatrix} B_1 & B_{1,3} & B_{1,J} & B_{1,4} \\ B_{3,1} & B_3 & B_{3,J} & B_{3,4} \\ B_{J,1} & B_{J,3} & B_J & B_{J,4} \\ B_{4,1} & B_{4,3} & B_{4,J} & B_4 \end{bmatrix} \tag{6.5}$$

| Activities | $1$ | $\cdots$ | $j-1$ | $j$ | $\cdots$ | $i-1$ | $i$ | $i+1$ | $\cdots$ | $n$ |
|---|---|---|---|---|---|---|---|---|---|---|
| $1$ | 0 | $\cdots$ | $d_{1,j-1}$ | $d_{1,j}$ | $\cdots$ | $d_{1,i-1}$ | $d_{1,i}$ | $d_{1,i+1}$ | $\cdots$ | $d_{1,n}$ |
| $\cdots$ | $\cdots$ | 0 | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ |
| $j-1$ | $d_{j-1,1}$ | $\cdots$ | 0 | $d_{j-1,j}$ | $\cdots$ | $d_{j-1,i-1}$ | $d_{j-1,i}$ | $d_{j-1,i+1}$ | $\cdots$ | $d_{j-1,n}$ |
| $j$ | $d_{j,1}$ | $\cdots$ | $d_{j,j-1}$ | 0 | $\cdots$ | $d_{j,i-1}$ | $d_{j,i}$ | $d_{j,i+1}$ | $\cdots$ | $d_{j,n}$ |
| $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | 0 | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ |
| $i-1$ | $d_{i-1,1}$ | $\cdots$ | $d_{i-1,j-1}$ | $d_{i-1,j}$ | $\cdots$ | 0 | $d_{i-1,i}$ | $d_{i-1,i+1}$ | $\cdots$ | $d_{i-1,n}$ |
| $i$ | $d_{i,1}$ | $\cdots$ | $d_{i,j-1}$ | $d_{i,j}$ | $\cdots$ | $d_{i,i-1}$ | 0 | $d_{i,i+1}$ | $\cdots$ | $d_{i,n}$ |
| $i+1$ | $d_{i+1,1}$ | $\cdots$ | $d_{i+1,j-1}$ | $d_{i+1,j}$ | $\cdots$ | $d_{i+1,i-1}$ | $d_{i+1,i}$ | 0 | $\cdots$ | $d_{i+1,n}$ |
| $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | 0 | $\cdots$ |
| $n$ | $d_{n,1}$ | $\cdots$ | $d_{n,j-1}$ | $d_{n,j}$ | $\cdots$ | $d_{n,i-1}$ | $d_{n,i}$ | $d_{n,i+1}$ | $\cdots$ | 0 |

**(a)** Original NDSM

| Activities | $1$ | $\cdots$ | $j-1$ | $i$ | $j$ | $\cdots$ | $i-1$ | $i+1$ | $\cdots$ | $n$ |
|---|---|---|---|---|---|---|---|---|---|---|
| $1$ | 0 | $\cdots$ | $d_{1,j-1}$ | $d_{1,i}$ | $d_{1,j}$ | $\cdots$ | $d_{1,i-1}$ | $d_{1,i+1}$ | $\cdots$ | $d_{1,n}$ |
| $\cdots$ | $\cdots$ | 0 | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ |
| $j-1$ | $d_{j-1,1}$ | $\cdots$ | 0 | $d_{j-1,i}$ | $d_{j-1,j}$ | $\cdots$ | $d_{j-1,i-1}$ | $d_{j-1,i+1}$ | $\cdots$ | $d_{j-1,n}$ |
| $i$ | $d_{i,1}$ | $\cdots$ | $d_{i,j-1}$ | 0 | $d_{i,j}$ | $\cdots$ | $d_{i,i-1}$ | $d_{i,i+1}$ | $\cdots$ | $d_{i,n}$ |
| $j$ | $d_{j,1}$ | $\cdots$ | $d_{j,j-1}$ | $d_{j,i}$ | 0 | $\cdots$ | $d_{j,i-1}$ | $d_{j,i+1}$ | $\cdots$ | $d_{j,n}$ |
| $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | 0 | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ |
| $i-1$ | $d_{i-1,1}$ | $\cdots$ | $d_{i-1,j-1}$ | $d_{i-1,i}$ | $d_{i-1,j}$ | $\cdots$ | 0 | $d_{i-1,i+1}$ | $\cdots$ | $d_{i-1,n}$ |
| $i+1$ | $d_{i+1,1}$ | $\cdots$ | $d_{i+1,j-1}$ | $d_{i+1,i}$ | $d_{i+1,j}$ | $\cdots$ | $d_{i+1,i-1}$ | 0 | $\cdots$ | $d_{i+1,n}$ |
| $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | 0 | $\cdots$ |
| $n$ | $d_{n,1}$ | $\cdots$ | $d_{n,j-1}$ | $d_{n,i}$ | $d_{n,j}$ | $\cdots$ | $d_{n,i-1}$ | $d_{n,i+1}$ | $\cdots$ | 0 |

**(b)** The resulting NDSM after exchanging block $B_J$ and activity $i$

**Figure D.1 The scenario used in the proof of Theorem 6.1**

Clearly, the different super-diagonal items (i.e. feedbacks) between (6.4) and (6.5) are $B_{J,3}$ and $B_{3,J}$. Then, subtracting the feedbacks in the NDSM of Figure D.1(a) from those in the NDSM of Figure D.1(b), we can get: $\sum_{k \in J}(d_{k,i} - d_{i,k})$, where $J = (j,...,i-1)$ denotes the activities from position $j$ to position $i-1$. It follows that if $\sum_{k \in J} d_{k,i} > \sum_{k \in J} d_{i,k}$, then exchanging block $B_J$ and activity $i$ results in a feedback reduction of $\sum_{k \in J}(d_{k,i} - d_{i,k})$. To make our statement more legible, in Figure D.1(a) and Figure D.1(b), $B_{J,3}$ is highlighted in grey and $B_{3,J}$ is highlighted in blue.

**Proof of Theorem 6.2**

Figure D.2(a) shows the original NDSM where activities are executed in the order of $\{1,...,j-1,j,...,i-1,i,...,h-1,h,h+1,...,n\}$. Separate the NDSM into five blocks and define:

$$B_1 = \begin{array}{|c|c|c|}\hline 0 & \cdots & d_{1,j-1} \\ \hline \cdots & 0 & \cdots \\ \hline d_{j-1,1} & \cdots & 0 \\ \hline\end{array} \quad B_J = \begin{array}{|c|c|c|}\hline 0 & \cdots & d_{j,i-1} \\ \hline \cdots & 0 & \cdots \\ \hline d_{i-1,j} & \cdots & 0 \\ \hline\end{array} \quad B_3 = \begin{array}{|c|c|c|}\hline 0 & \cdots & d_{i,h-1} \\ \hline \cdots & 0 & \cdots \\ \hline d_{h-1,i} & \cdots & 0 \\ \hline\end{array} \quad B_4 = \begin{array}{|c|}\hline 0 \\ \hline\end{array}$$

$$B_5 = \begin{array}{|c|c|c|}\hline 0 & \cdots & d_{h+1,n} \\ \hline \cdots & 0 & \cdots \\ \hline d_{n,h+1} & \cdots & 0 \\ \hline\end{array} \quad B_{1,J} = \begin{bmatrix} d_{1,j} & ... & d_{1,i-1} \\ ... & ... & ... \\ d_{j-1,j} & ... & d_{j-1,i-1} \end{bmatrix} \quad B_{1,3} = \begin{bmatrix} d_{1,i} & ... & d_{1,h-1} \\ ... & ... & ... \\ d_{j-1,i} & ... & d_{j-1,h-1} \end{bmatrix}$$

$$B_{1,4} = \begin{bmatrix} d_{1,h} \\ ... \\ d_{j-1,h} \end{bmatrix} \quad B_{1,5} = \begin{bmatrix} d_{1,h+1} & ... & d_{1,n} \\ ... & ... & ... \\ d_{j-1,h+1} & ... & d_{j-1,n} \end{bmatrix} \quad B_{J,1} = \begin{bmatrix} d_{j,1} & ... & d_{j,j-1} \\ ... & ... & ... \\ d_{i-1,1} & ... & d_{i-1,j-1} \end{bmatrix}$$

$$B_{J,3} = \begin{bmatrix} d_{j,i} & ... & d_{j,h-1} \\ ... & ... & ... \\ d_{i-1,i} & ... & d_{i-1,h-1} \end{bmatrix} \quad B_{J,4} = \begin{bmatrix} d_{j,h} \\ ... \\ d_{i-1,h} \end{bmatrix} \quad B_{J,5} = \begin{bmatrix} d_{j,h+1} & ... & d_{j,n} \\ ... & ... & ... \\ d_{i-1,h+1} & ... & d_{i-1,n} \end{bmatrix}$$

$$B_{3,1} = \begin{bmatrix} d_{i,1} & \dots & d_{i,j-1} \\ \dots & \dots & \dots \\ d_{h-1,1} & \dots & d_{h-1,j-1} \end{bmatrix} \qquad B_{3,J} = \begin{bmatrix} d_{i,j} & \dots & d_{i,i-1} \\ \dots & \dots & \dots \\ d_{h-1,j} & \dots & d_{h-1,i-1} \end{bmatrix} \qquad B_{3,4} = \begin{bmatrix} d_{i,h} \\ \dots \\ d_{h-1,h} \end{bmatrix}$$

$$B_{3,5} = \begin{bmatrix} d_{i,h+1} & \dots & d_{i,n} \\ \dots & \dots & \dots \\ d_{h-1,h+1} & \dots & d_{h-1,n} \end{bmatrix} \qquad B_{4,1} = [d_{h,1} \ \dots \ d_{h,j-1}] \qquad B_{4,J} = [d_{h,j} \ \dots \ d_{h,i-1}]$$

$$B_{4,3} = [d_{h,i} \ \dots \ d_{h,h-1}] \qquad B_{4,5} = [d_{h,h+1} \ \dots \ d_{h,n}] \qquad B_{5,1} = \begin{bmatrix} d_{h+1,1} & \dots & d_{h+1,j-1} \\ \dots & \dots & \dots \\ d_{n,1} & \dots & d_{n,j-1} \end{bmatrix}$$

$$B_{5,J} = \begin{bmatrix} d_{h+1,j} & \dots & d_{h+1,i-1} \\ \dots & \dots & \dots \\ d_{n,j} & \dots & d_{n,i-1} \end{bmatrix} \qquad B_{5,3} = \begin{bmatrix} d_{h+1,i} & \dots & d_{h+1,h-1} \\ \dots & \dots & \dots \\ d_{n,i} & \dots & d_{n,h-1} \end{bmatrix} \qquad B_{5,4} = \begin{bmatrix} d_{h+1,h} \\ \dots \\ d_{n,h} \end{bmatrix}$$

Figure D.2(b) shows the resulting NDSM after exchanging block $B_J$ and activity $h$. Based on above definitions, the NDSMs in Figure D.2(a) and Figure D.2 (b) can be represented as following (6.6) and (6.7), respectively:

$$\begin{bmatrix} B_1 & B_{1,J} & B_{1,3} & B_{1,4} & B_{1,5} \\ B_{J,1} & B_J & B_{J,3} & B_{J,4} & B_{J,5} \\ B_{3,1} & B_{3,J} & B_3 & B_{3,4} & B_{3,5} \\ B_{4,1} & B_{4,J} & B_{4,3} & B_4 & B_{4,5} \\ B_{5,1} & B_{5,J} & B_{5,3} & B_{5,4} & B_5 \end{bmatrix} \tag{6.6}$$

$$\begin{bmatrix} B_1 & B_{1,4} & B_{1,3} & B_{1,J} & B_{1,5} \\ B_{4,1} & B_4 & B_{4,3} & B_{4,J} & B_{4,5} \\ B_{3,1} & B_{3,4} & B_3 & B_{3,J} & B_{3,5} \\ B_{J,1} & B_{J,4} & B_{J,3} & B_J & B_{J,5} \\ B_{5,1} & B_{5,4} & B_{5,3} & B_{5,J} & B_5 \end{bmatrix} \tag{6.7}$$

The different super-diagonal items between (6.6) and (6.7) are $B_{J,3}$, $B_{J,4}$, $B_{3,4}$ and $B_{3,J}$, $B_{4,J}$, $B_{4,3}$. In Figure D.2(a) and Figure D.2(b), $B_{J,3}$, $B_{J,4}$ and $B_{3,4}$ are highlighted in grey, while $B_{3,J}$, $B_{4,J}$ and $B_{4,3}$ are highlighted in blue.

| Activities | 1 | … | $j-1$ | $j$ | … | $i-1$ | $i$ | … | $h-1$ | $h$ | $h+1$ | … | $n$ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 0 | … | $d_{1,j-1}$ | $d_{1,j}$ | … | $d_{1,i-1}$ | $d_{1,i}$ | … | $d_{1,h-1}$ | $d_{1,h}$ | $d_{1,h+1}$ | … | $d_{1,n}$ |
| … | … | 0 | … | … | … | … | … | … | … | … | … | … | … |
| $j-1$ | $d_{j-1,1}$ | … | 0 | $d_{j-1,j}$ | … | $d_{j-1,i-1}$ | $d_{j-1,i}$ | … | $d_{j-1,h-1}$ | $d_{j-1,h}$ | $d_{j-1,h+1}$ | … | $d_{j-1,n}$ |
| $j$ | $d_{j,1}$ | … | $d_{j,j-1}$ | 0 | … | $d_{j,i-1}$ | $d_{j,i}$ | … | $d_{j,h-1}$ | $d_{j,h}$ | $d_{j,h+1}$ | … | $d_{j,n}$ |
| … | … | … | … | … | 0 | … | … | … | … | … | … | … | … |
| $i-1$ | $d_{i-1,1}$ | … | $d_{i-1,j-1}$ | $d_{i-1,j}$ | … | 0 | $d_{i-1,i}$ | … | $d_{i-1,h-1}$ | $d_{i-1,h}$ | $d_{i-1,h+1}$ | … | $d_{i-1,n}$ |
| $i$ | $d_{i,1}$ | … | $d_{i,j-1}$ | $d_{i,j}$ | … | $d_{i,i-1}$ | 0 | … | $d_{i,h-1}$ | $d_{i,h}$ | $d_{i,h+1}$ | … | $d_{i,n}$ |
| … | … | … | … | … | … | … | … | 0 | … | … | … | … | … |
| $h-1$ | $d_{h-1,1}$ | … | $d_{h-1,j-1}$ | $d_{h-1,j}$ | … | $d_{h-1,i-1}$ | $d_{h-1,i}$ | … | 0 | $d_{h-1,h}$ | $d_{h-1,h+1}$ | … | $d_{h-1,n}$ |
| $h$ | $d_{h,1}$ | … | $d_{h,j-1}$ | $d_{h,j}$ | … | $d_{h,i-1}$ | $d_{h,i}$ | … | $d_{h,h-1}$ | 0 | $d_{h,h+1}$ | … | $d_{h,n}$ |
| $h+1$ | $d_{h+1,1}$ | … | $d_{h+1,j-1}$ | $d_{h+1,j}$ | … | $d_{h+1,i-1}$ | $d_{h+1,i}$ | … | $d_{h+1,h-1}$ | $d_{h+1,h}$ | 0 | … | $d_{h+1,n}$ |
| … | … | … | … | … | … | … | … | … | … | … | … | 0 | … |
| $n$ | $d_{n,1}$ | … | $d_{n,j-1}$ | $d_{n,j}$ | … | $d_{n,i-1}$ | $d_{n,i}$ | … | $d_{n,h-1}$ | $d_{n,h}$ | $d_{n,h+1}$ | … | 0 |

**(a)** Original NDSM

| Activities | 1 | … | $j-1$ | $h$ | $i$ | … | $h-1$ | $j$ | … | $i-1$ | $h+1$ | … | $n$ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 0 | … | $d_{1,j-1}$ | $d_{1,h}$ | $d_{1,i}$ | … | $d_{1,h-1}$ | $d_{1,j}$ | … | $d_{1,i-1}$ | $d_{1,h+1}$ | … | $d_{1,n}$ |
| … | … | 0 | … | … | … | … | … | … | … | … | … | … | … |
| $j-1$ | $d_{j-1,1}$ | … | 0 | $d_{j-1,h}$ | $d_{j-1,i}$ | … | $d_{j-1,h-1}$ | $d_{j-1,j}$ | … | $d_{j-1,i-1}$ | $d_{j-1,h+1}$ | … | $d_{j-1,n}$ |
| $h$ | $d_{h,1}$ | … | $d_{h,j-1}$ | 0 | $d_{h,i}$ | … | $d_{h,h-1}$ | $d_{h,j}$ | … | $d_{h,i-1}$ | $d_{h,h+1}$ | … | $d_{h,n}$ |
| $i$ | $d_{i,1}$ | … | $d_{i,j-1}$ | $d_{i,h}$ | 0 | … | $d_{i,h-1}$ | $d_{i,j}$ | … | $d_{i,i-1}$ | $d_{i,h+1}$ | … | $d_{i,n}$ |
| … | … | … | … | … | … | 0 | … | … | … | … | … | … | … |
| $h-1$ | $d_{h-1,1}$ | … | $d_{h-1,j-1}$ | $d_{h-1,h}$ | $d_{h-1,i}$ | … | 0 | $d_{h-1,j}$ | … | $d_{h-1,i-1}$ | $d_{h-1,h+1}$ | … | $d_{h-1,n}$ |
| $j$ | $d_{j,1}$ | … | $d_{j,j-1}$ | $d_{j,h}$ | $d_{j,i}$ | … | $d_{j,h-1}$ | 0 | … | $d_{j,i-1}$ | $d_{j,h+1}$ | … | $d_{j,n}$ |
| … | … | … | … | … | … | … | … | … | 0 | … | … | … | … |
| $i-1$ | $d_{i-1,1}$ | … | $d_{i-1,j-1}$ | $d_{i-1,h}$ | $d_{i-1,i}$ | … | $d_{i-1,h-1}$ | $d_{i-1,j}$ | … | 0 | $d_{i-1,h+1}$ | … | $d_{i-1,n}$ |
| $h+1$ | $d_{h+1,1}$ | … | $d_{h+1,j-1}$ | $d_{h+1,h}$ | $d_{h+1,i}$ | … | $d_{h+1,h-1}$ | $d_{h+1,j}$ | … | $d_{h+1,i-1}$ | 0 | … | $d_{h+1,n}$ |
| … | … | … | … | … | … | … | … | … | … | … | … | 0 | … |
| $n$ | $d_{n,1}$ | … | $d_{n,j-1}$ | $d_{n,h}$ | $d_{n,i}$ | … | $d_{n,h-1}$ | $d_{n,j}$ | … | $d_{n,i-1}$ | $d_{n,h+1}$ | … | 0 |

**(b)** The resulting NDSM after exchanging block $B_J$ and activity $h$

**Figure D.2 The scenario used in the proof of Theorem 6.2**

Let $J=(j,...,i-1)$, $I=(i,...,h-1)$. Clearly, subtracting the feedbacks in the

NDSM of Figure D.2(a) from those in the NDSM of Figure D.2(b), we can get $\theta_J^h = \sum_{k \in J}\sum_{r \in I}\left(d_{k,r} - d_{r,k}\right) + \sum_{k \in J}\left(d_{k,h} - d_{h,k}\right) + \sum_{r \in I}\left(d_{r,h} - d_{h,r}\right)$. It follows that if $\theta_J^h > 0$, then through exchanging block $B_J$ and activity $h$, the feedbacks in the NDSM of Figure D.2(a) can be reduced by $\theta_J^h$.

**Proof of Theorem 6.3**

Figure D.3(a) shows the original NDSM for an activity sequence of $\{1,...,j-1, j,...,i-1, i,...,h-1, h,...,n\}$. Break the NDSM into four blocks and define:

$$B_1 = \begin{array}{|c|c|c|}\hline 0 & \cdots & d_{1,j-1} \\ \hline \cdots & 0 & \cdots \\ \hline d_{j-1,1} & \cdots & 0 \\ \hline \end{array} \quad B_J = \begin{array}{|c|c|c|}\hline 0 & \cdots & d_{j,i-1} \\ \hline \cdots & 0 & \cdots \\ \hline d_{i-1,j} & \cdots & 0 \\ \hline \end{array} \quad B_I = \begin{array}{|c|c|c|}\hline 0 & \cdots & d_{i,h-1} \\ \hline \cdots & 0 & \cdots \\ \hline d_{h-1,i} & \cdots & 0 \\ \hline \end{array} \quad B_4 = \begin{array}{|c|c|c|}\hline 0 & \cdots & d_{h,n} \\ \hline \cdots & 0 & \cdots \\ \hline d_{n,h} & \cdots & 0 \\ \hline \end{array}$$

$$B_{1,I} = \begin{bmatrix} d_{1,i} & \dots & d_{1,h-1} \\ \dots & \dots & \dots \\ d_{j-1,i} & \dots & d_{j-1,h-1} \end{bmatrix} \quad B_{1,4} = \begin{bmatrix} d_{1,h} & \dots & d_{1,n} \\ \dots & \dots & \dots \\ d_{j-1,h} & \dots & d_{j-1,n} \end{bmatrix} \quad B_{1,J} = \begin{bmatrix} d_{1,j} & \dots & d_{1,i-1} \\ \dots & \dots & \dots \\ d_{j-1,j} & \dots & d_{j-1,i-1} \end{bmatrix}$$

$$B_{J,1} = \begin{bmatrix} d_{j,1} & \dots & d_{j,j-1} \\ \dots & \dots & \dots \\ d_{i-1,1} & \dots & d_{i-1,j-1} \end{bmatrix} \quad B_{J,I} = \begin{bmatrix} d_{j,i} & \dots & d_{j,h-1} \\ \dots & \dots & \dots \\ d_{i-1,i} & \dots & d_{i-1,h-1} \end{bmatrix} \quad B_{J,4} = \begin{bmatrix} d_{j,h} & \dots & d_{j,n} \\ \dots & \dots & \dots \\ d_{i-1,h} & \dots & d_{i-1,n} \end{bmatrix}$$

$$B_{I,1} = \begin{bmatrix} d_{i,1} & \dots & d_{i,j-1} \\ \dots & \dots & \dots \\ d_{h-1,1} & \dots & d_{h-1,j-1} \end{bmatrix} \quad B_{I,J} = \begin{bmatrix} d_{i,j} & \dots & d_{i,i-1} \\ \dots & \dots & \dots \\ d_{h-1,j} & \dots & d_{h-1,i-1} \end{bmatrix} \quad B_{I,4} = \begin{bmatrix} d_{i,h} & \dots & d_{i,n} \\ \dots & \dots & \dots \\ d_{h-1,h} & \dots & d_{h-1,n} \end{bmatrix}$$

$$B_{4,1} = \begin{bmatrix} d_{h,1} & \dots & d_{h,j-1} \\ \dots & \dots & \dots \\ d_{n,1} & \dots & d_{n,j-1} \end{bmatrix} \quad B_{4,J} = \begin{bmatrix} d_{h,j} & \dots & d_{h,i-1} \\ \dots & \dots & \dots \\ d_{n,j} & \dots & d_{n,i-1} \end{bmatrix} \quad B_{4,I} = \begin{bmatrix} d_{h,i} & \dots & d_{h,h-1} \\ \dots & \dots & \dots \\ d_{n,i} & \dots & d_{n,h-1} \end{bmatrix}$$

Based on above definitions, the NDSM in Figure D.3(a) can be expressed as:

$$\begin{bmatrix} B_1 & B_{1,J} & B_{1,I} & B_{1,4} \\ B_{J,1} & B_J & B_{J,I} & B_{J,4} \\ B_{I,1} & B_{I,J} & B_I & B_{I,4} \\ B_{4,1} & B_{4,J} & B_{4,I} & B_4 \end{bmatrix} \tag{6.8}$$

| Activities | 1 | ⋯ | $j-1$ | $j$ | ⋯ | $i-1$ | $i$ | ⋯ | $h-1$ | $h$ | ⋯ | $n$ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 0 | ⋯ | $d_{1,j-1}$ | $d_{1,j}$ | ⋯ | $d_{1,i-1}$ | $d_{1,i}$ | ⋯ | $d_{1,h-1}$ | $d_{1,h}$ | ⋯ | $d_{1,n}$ |
| ⋯ | ⋯ | 0 | ⋯ | ⋯ | ⋯ | ⋯ | ⋯ | ⋯ | ⋯ | ⋯ | ⋯ | ⋯ |
| $j-1$ | $d_{j-1,1}$ | ⋯ | 0 | $d_{j-1,j}$ | ⋯ | $d_{j-1,i-1}$ | $d_{j-1,i}$ | ⋯ | $d_{j-1,h-1}$ | $d_{j-1,h}$ | ⋯ | $d_{j-1,n}$ |
| $j$ | $d_{j,1}$ | ⋯ | $d_{j,j-1}$ | 0 | ⋯ | $d_{j,i-1}$ | $d_{j,i}$ | ⋯ | $d_{j,h-1}$ | $d_{j,h}$ | ⋯ | $d_{j,n}$ |
| ⋯ | ⋯ | ⋯ | ⋯ | ⋯ | 0 | ⋯ | ⋯ | ⋯ | ⋯ | ⋯ | ⋯ | ⋯ |
| $i-1$ | $d_{i-1,1}$ | ⋯ | $d_{i-1,j-1}$ | $d_{i-1,j}$ | ⋯ | 0 | $d_{i-1,i}$ | ⋯ | $d_{i-1,h-1}$ | $d_{i-1,h}$ | ⋯ | $d_{i-1,n}$ |
| $i$ | $d_{i,1}$ | ⋯ | $d_{i,j-1}$ | $d_{i,j}$ | ⋯ | $d_{i,i-1}$ | 0 | ⋯ | $d_{i,h-1}$ | $d_{i,h}$ | ⋯ | $d_{i,n}$ |
| ⋯ | ⋯ | ⋯ | ⋯ | ⋯ | ⋯ | ⋯ | ⋯ | 0 | ⋯ | ⋯ | ⋯ | ⋯ |
| $h-1$ | $d_{h-1,1}$ | ⋯ | $d_{h-1,j-1}$ | $d_{h-1,j}$ | ⋯ | $d_{h-1,i-1}$ | $d_{h-1,i}$ | ⋯ | 0 | $d_{h-1,h}$ | ⋯ | $d_{h-1,n}$ |
| $h$ | $d_{h,1}$ | ⋯ | $d_{h,j-1}$ | $d_{h,j}$ | ⋯ | $d_{h,i-1}$ | $d_{h,i}$ | ⋯ | $d_{h,h-1}$ | 0 | ⋯ | $d_{h,n}$ |
| ⋯ | ⋯ | ⋯ | ⋯ | ⋯ | ⋯ | ⋯ | ⋯ | ⋯ | ⋯ | ⋯ | 0 | ⋯ |
| $n$ | $d_{n,1}$ | ⋯ | $d_{n,j-1}$ | $d_{n,j}$ | ⋯ | $d_{n,i-1}$ | $d_{n,i}$ | ⋯ | $d_{n,h-1}$ | $d_{n,h}$ | ⋯ | 0 |

**(a)** Original NDSM

| Activities | 1 | ⋯ | $j-1$ | $i$ | ⋯ | $h-1$ | $j$ | ⋯ | $i-1$ | $h$ | ⋯ | $n$ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 0 | ⋯ | $d_{1,j-1}$ | $d_{1,i}$ | ⋯ | $d_{1,h-1}$ | $d_{1,j}$ | ⋯ | $d_{1,i-1}$ | $d_{1,h}$ | ⋯ | $d_{1,n}$ |
| ⋯ | ⋯ | 0 | ⋯ | ⋯ | ⋯ | ⋯ | ⋯ | ⋯ | ⋯ | ⋯ | ⋯ | ⋯ |
| $j-1$ | $d_{j-1,1}$ | ⋯ | 0 | $d_{j-1,i}$ | ⋯ | $d_{j-1,h-1}$ | $d_{j-1,j}$ | ⋯ | $d_{j-1,i-1}$ | $d_{j-1,h}$ | ⋯ | $d_{j-1,n}$ |
| $i$ | $d_{i,1}$ | ⋯ | $d_{i,j-1}$ | 0 | ⋯ | $d_{i,h-1}$ | $d_{i,j}$ | ⋯ | $d_{i,i-1}$ | $d_{i,h}$ | ⋯ | $d_{i,n}$ |
| ⋯ | ⋯ | ⋯ | ⋯ | ⋯ | 0 | ⋯ | ⋯ | ⋯ | ⋯ | ⋯ | ⋯ | ⋯ |
| $h-1$ | $d_{h-1,1}$ | ⋯ | $d_{h-1,j-1}$ | $d_{h-1,i}$ | ⋯ | 0 | $d_{h-1,j}$ | ⋯ | $d_{h-1,i-1}$ | $d_{h-1,h}$ | ⋯ | $d_{h-1,n}$ |
| $j$ | $d_{j,1}$ | ⋯ | $d_{j,j-1}$ | $d_{j,i}$ | ⋯ | $d_{j,h-1}$ | 0 | ⋯ | $d_{j,i-1}$ | $d_{j,h}$ | ⋯ | $d_{j,n}$ |
| ⋯ | ⋯ | ⋯ | ⋯ | ⋯ | ⋯ | ⋯ | ⋯ | 0 | ⋯ | ⋯ | ⋯ | ⋯ |
| $i-1$ | $d_{i-1,1}$ | ⋯ | $d_{i-1,j-1}$ | $d_{i-1,i}$ | ⋯ | $d_{i-1,h-1}$ | $d_{i-1,j}$ | ⋯ | 0 | $d_{i-1,h}$ | ⋯ | $d_{i-1,n}$ |
| $h$ | $d_{h,1}$ | ⋯ | $d_{h,j-1}$ | $d_{h,i}$ | ⋯ | $d_{h,h-1}$ | $d_{h,j}$ | ⋯ | $d_{h,i-1}$ | 0 | ⋯ | $d_{h,n}$ |
| ⋯ | ⋯ | ⋯ | ⋯ | ⋯ | ⋯ | ⋯ | ⋯ | ⋯ | ⋯ | ⋯ | 0 | ⋯ |
| $n$ | $d_{n,1}$ | ⋯ | $d_{n,j-1}$ | $d_{n,i}$ | ⋯ | $d_{n,h-1}$ | $d_{n,j}$ | ⋯ | $d_{n,i-1}$ | $d_{n,h}$ | ⋯ | 0 |

(b) The resulting NDSM after exchanging block $B_J$ and block $B_I$

**Figure D.3 The scenario used in the proof of Theorem 6.3**

Figure D.3(b) shows the resulting NDSM after exchanging block $B_J$ and block $B_I$, which can be expressed as:

$$\begin{bmatrix} B_1 & B_{1,I} & B_{1,J} & B_{1,4} \\ B_{I,1} & B_I & B_{I,J} & B_{I,4} \\ B_{J,1} & B_{J,I} & B_J & B_{J,4} \\ B_{4,1} & B_{4,I} & B_{4,J} & B_4 \end{bmatrix} \tag{6.9}$$

The different super-diagonal items between (6.8) and (6.9) are $B_{J,I}$ and $B_{I,J}$, and so subtracting the feedbacks in the NDSM of Figure D.3(a) from those in the NDSM of Figure D.3(b), we are left with $\sum_{k\in J}\sum_{r\in I}(d_{k,r} - d_{r,k})$, where $J = (j,...,i-1)$ and $I = (i,...,h-1)$. Thus, if $\sum_{k\in J}\sum_{r\in I} d_{k,r} > \sum_{k\in J}\sum_{r\in I} d_{r,k}$, exchanging block $B_J$ and block $B_I$ leads to a feedback reduction of $\sum_{k\in J}\sum_{r\in I}(d_{k,r} - d_{r,k})$. In Figure D.3(a) and Figure D.3(b), $B_{J,I}$ is highlighted in grey, and $B_{I,J}$ is highlighted in blue.

**Proof of Theorem 6.4**

Figure D.4(a) shows the original DSM, where activities are in the order of $\{1,...,j-1, j,...,i-1, i,...,h-1, h, ...,k-1, k...,n\}$. Let:

$$B_1 = \begin{array}{|c|c|c|}\hline 0 & \cdots & d_{1,j-1} \\ \hline \cdots & 0 & \cdots \\ \hline d_{j-1,1} & \cdots & 0 \\ \hline \end{array} \qquad B_J = \begin{array}{|c|c|c|}\hline 0 & \cdots & d_{j,i-1} \\ \hline \cdots & 0 & \cdots \\ \hline d_{i-1,j} & \cdots & 0 \\ \hline \end{array} \qquad B_3 = \begin{array}{|c|c|c|}\hline 0 & \cdots & d_{i,h-1} \\ \hline \cdots & 0 & \cdots \\ \hline d_{h-1,i} & \cdots & 0 \\ \hline \end{array}$$

$$B_H = \begin{array}{|c|c|c|}\hline 0 & \cdots & d_{h,k-1} \\ \hline \cdots & 0 & \cdots \\ \hline d_{k-1,h} & \cdots & 0 \\ \hline \end{array} \qquad B_5 = \begin{array}{|c|c|c|}\hline 0 & \cdots & d_{k,n} \\ \hline \cdots & 0 & \cdots \\ \hline d_{n,k} & \cdots & 0 \\ \hline \end{array}$$

$$B_{1,J} = \begin{bmatrix} d_{1,j} & \cdots & d_{1,i-1} \\ \cdots & \cdots & \cdots \\ d_{j-1,j} & \cdots & d_{j-1,i-1} \end{bmatrix} \qquad B_{1,3} = \begin{bmatrix} d_{1,i} & \cdots & d_{1,h-1} \\ \cdots & \cdots & \cdots \\ d_{j-1,i} & \cdots & d_{j-1,h-1} \end{bmatrix} \qquad B_{1,H} = \begin{bmatrix} d_{1,h} & \cdots & d_{1,k-1} \\ \cdots & \cdots & \cdots \\ d_{j-1,h} & \cdots & d_{j-1,k-1} \end{bmatrix}$$

$$B_{1,5} = \begin{bmatrix} d_{1,k} & ... & d_{1,n} \\ ... & ... & ... \\ d_{j-1,k} & ... & d_{j-1,n} \end{bmatrix} \quad B_{J,1} = \begin{bmatrix} d_{j,1} & ... & d_{j,j-1} \\ ... & ... & ... \\ d_{i-1,1} & ... & d_{i-1,j-1} \end{bmatrix} \quad B_{J,3} = \begin{bmatrix} d_{j,i} & ... & d_{j,h-1} \\ ... & ... & ... \\ d_{i-1,i} & ... & d_{i-1,h-1} \end{bmatrix}$$

$$B_{J,H} = \begin{bmatrix} d_{j,h} & ... & d_{j,k-1} \\ ... & ... & ... \\ d_{i-1,h} & ... & d_{i-1,k-1} \end{bmatrix} \quad B_{J,5} = \begin{bmatrix} d_{j,k} & ... & d_{j,n} \\ ... & ... & ... \\ d_{i-1,k} & ... & d_{i-1,n} \end{bmatrix} \quad B_{3,1} = \begin{bmatrix} d_{i,1} & ... & d_{i,j-1} \\ ... & ... & ... \\ d_{h-1,1} & ... & d_{h-1,j-1} \end{bmatrix}$$

$$B_{3,J} = \begin{bmatrix} d_{i,j} & ... & d_{i,i-1} \\ ... & ... & ... \\ d_{h-1,j} & ... & d_{h-1,i-1} \end{bmatrix} \quad B_{3,H} = \begin{bmatrix} d_{i,h} & ... & d_{i,k-1} \\ ... & ... & ... \\ d_{h-1,h} & ... & d_{h-1,k-1} \end{bmatrix} \quad B_{3,5} = \begin{bmatrix} d_{i,k} & ... & d_{i,n} \\ ... & ... & ... \\ d_{h-1,k} & ... & d_{h-1,n} \end{bmatrix}$$

$$B_{H,1} = \begin{bmatrix} d_{h,1} & ... & d_{h,j-1} \\ ... & ... & ... \\ d_{k-1,1} & ... & d_{k-1,j-1} \end{bmatrix} \quad B_{H,J} = \begin{bmatrix} d_{h,j} & ... & d_{h,i-1} \\ ... & ... & ... \\ d_{k-1,j} & ... & d_{k-1,i-1} \end{bmatrix} \quad B_{H,3} = \begin{bmatrix} d_{h,i} & ... & d_{h,h-1} \\ ... & ... & ... \\ d_{k-1,i} & ... & d_{k-1,h-1} \end{bmatrix}$$

$$B_{H,5} = \begin{bmatrix} d_{h,k} & ... & d_{h,n} \\ ... & ... & ... \\ d_{k-1,k} & ... & d_{k-1,n} \end{bmatrix} \quad B_{5,1} = \begin{bmatrix} d_{k,1} & ... & d_{k,j-1} \\ ... & ... & ... \\ d_{n,1} & ... & d_{n,j-1} \end{bmatrix} \quad B_{5,J} = \begin{bmatrix} d_{k,j} & ... & d_{k,i-1} \\ ... & ... & ... \\ d_{n,j} & ... & d_{n,i-1} \end{bmatrix}$$

$$B_{5,3} = \begin{bmatrix} d_{k,i} & ... & d_{k,h-1} \\ ... & ... & ... \\ d_{n,i} & ... & d_{n,h-1} \end{bmatrix} \quad B_{5,H} = \begin{bmatrix} d_{k,h} & ... & d_{k,k-1} \\ ... & ... & ... \\ d_{n,h} & ... & d_{n,k-1} \end{bmatrix}$$

Based on the definitions, the NDSM in Figure D.4(a) can be expressed as:

$$\begin{bmatrix} B_1 & B_{1,J} & B_{1,3} & B_{1,H} & B_{1,5} \\ B_{J,1} & B_J & B_{J,3} & B_{J,H} & B_{J,5} \\ B_{3,1} & B_{3,J} & B_3 & B_{3,H} & B_{3,5} \\ B_{H,1} & B_{H,J} & B_{H,3} & B_H & B_{H,5} \\ B_{5,1} & B_{5,J} & B_{5,3} & B_{5,H} & B_5 \end{bmatrix} \tag{6.10}$$

Figure D.4(b) shows the resulting NDSM after exchanging block $B_J$ and block $B_H$, which can be written as:

$$\begin{bmatrix} B_1 & B_{1,H} & B_{1,3} & B_{1,J} & B_{1,5} \\ B_{H,1} & B_H & B_{H,3} & B_{H,J} & B_{H,5} \\ B_{3,1} & B_{3,H} & B_3 & B_{3,J} & B_{3,5} \\ B_{J,1} & B_{J,H} & B_{J,3} & B_J & B_{J,5} \\ B_{5,1} & B_{5,H} & B_{5,3} & B_{5,J} & B_5 \end{bmatrix} \tag{6.11}$$

| Activities | 1 | $\cdots$ | $j-1$ | $j$ | $\cdots$ | $i-1$ | $i$ | $\cdots$ | $h-1$ | $h$ | $\cdots$ | $k-1$ | $k$ | $\cdots$ | $n$ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 0 | $\cdots$ | $d_{1,j-1}$ | $d_{1,j}$ | $\cdots$ | $d_{1,i-1}$ | $d_{1,i}$ | $\cdots$ | $d_{1,h-1}$ | $d_{1,h}$ | $\cdots$ | $d_{1,k-1}$ | $d_{1,k}$ | $\cdots$ | $d_{1,n}$ |
| $\cdots$ | $\cdots$ | 0 | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ |
| $j-1$ | $d_{j-1,1}$ | $\cdots$ | 0 | $d_{j-1,j}$ | $\cdots$ | $d_{j-1,i-1}$ | $d_{j-1,i}$ | $\cdots$ | $d_{j-1,h-1}$ | $d_{j-1,h}$ | $\cdots$ | $d_{j-1,k-1}$ | $d_{j-1,k}$ | $\cdots$ | $d_{j-1,n}$ |
| $j$ | $d_{j,1}$ | $\cdots$ | $d_{j,j-1}$ | 0 | $\cdots$ | $d_{j,i-1}$ | $d_{j,i}$ | $\cdots$ | $d_{j,h-1}$ | $d_{j,h}$ | $\cdots$ | $d_{j,k-1}$ | $d_{j,k}$ | $\cdots$ | $d_{j,n}$ |
| $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | 0 | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ |
| $i-1$ | $d_{i-1,1}$ | $\cdots$ | $d_{i-1,j-1}$ | $d_{i-1,j}$ | $\cdots$ | 0 | $d_{i-1,i}$ | $\cdots$ | $d_{i-1,h-1}$ | $d_{i-1,h}$ | $\cdots$ | $d_{i-1,k-1}$ | $d_{i-1,k}$ | $\cdots$ | $d_{i-1,n}$ |
| $i$ | $d_{i,1}$ | $\cdots$ | $d_{i,j-1}$ | $d_{i,j}$ | $\cdots$ | $d_{i,i-1}$ | 0 | $\cdots$ | $d_{i,h-1}$ | $d_{i,h}$ | $\cdots$ | $d_{i,k-1}$ | $d_{i,k}$ | $\cdots$ | $d_{i,n}$ |
| $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | 0 | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ |
| $h-1$ | $d_{h-1,1}$ | $\cdots$ | $d_{h-1,j-1}$ | $d_{h-1,j}$ | $\cdots$ | $d_{h-1,i-1}$ | $d_{h-1,i}$ | $\cdots$ | 0 | $d_{h-1,h}$ | $\cdots$ | $d_{h-1,k-1}$ | $d_{h-1,k}$ | $\cdots$ | $d_{h-1,n}$ |
| $h$ | $d_{h,1}$ | $\cdots$ | $d_{h,j-1}$ | $d_{h,j}$ | $\cdots$ | $d_{h,i-1}$ | $d_{h,i}$ | $\cdots$ | $d_{h,h-1}$ | 0 | $\cdots$ | $d_{h,k-1}$ | $d_{h,k}$ | $\cdots$ | $d_{h,n}$ |
| $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | 0 | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ |
| $k-1$ | $d_{k-1,1}$ | $\cdots$ | $d_{k-1,j-1}$ | $d_{k-1,j}$ | $\cdots$ | $d_{k-1,i-1}$ | $d_{k-1,i}$ | $\cdots$ | $d_{k-1,h-1}$ | $d_{k-1,h}$ | $\cdots$ | 0 | $d_{k-1,k}$ | $\cdots$ | $d_{k-1,n}$ |
| $k$ | $d_{k,1}$ | $\cdots$ | $d_{k,j-1}$ | $d_{k,j}$ | $\cdots$ | $d_{k,i-1}$ | $d_{k,i}$ | $\cdots$ | $d_{k,h-1}$ | $d_{k,h}$ | $\cdots$ | $d_{k,k-1}$ | 0 | $\cdots$ | $d_{k,n}$ |
| $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | 0 | $\cdots$ |
| $n$ | $d_{n,1}$ | $\cdots$ | $d_{n,j-1}$ | $d_{n,j}$ | $\cdots$ | $d_{n,i-1}$ | $d_{n,i}$ | $\cdots$ | $d_{n,h-1}$ | $d_{n,h}$ | $\cdots$ | $d_{n,k-1}$ | $d_{n,k}$ | $\cdots$ | 0 |

**(a)** Original NDSM

| Activities | 1 | $\cdots$ | $j-1$ | $h$ | $\cdots$ | $k-1$ | $i$ | $\cdots$ | $h-1$ | $j$ | $\cdots$ | $i-1$ | $k$ | $\cdots$ | $n$ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 0 | $\cdots$ | $d_{1,j-1}$ | $d_{1,h}$ | $\cdots$ | $d_{1,k-1}$ | $d_{1,i}$ | $\cdots$ | $d_{1,h-1}$ | $d_{1,j}$ | $\cdots$ | $d_{1,i-1}$ | $d_{1,k}$ | $\cdots$ | $d_{1,n}$ |
| $\cdots$ | $\cdots$ | 0 | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ |
| $j-1$ | $d_{j-1,1}$ | $\cdots$ | 0 | $d_{j-1,h}$ | $\cdots$ | $d_{j-1,k-1}$ | $d_{j-1,i}$ | $\cdots$ | $d_{j-1,h-1}$ | $d_{j-1,j}$ | $\cdots$ | $d_{j-1,i-1}$ | $d_{j-1,k}$ | $\cdots$ | $d_{j-1,n}$ |
| $h$ | $d_{h,1}$ | $\cdots$ | $d_{h,j-1}$ | 0 | $\cdots$ | $d_{h,k-1}$ | $d_{h,i}$ | $\cdots$ | $d_{h,h-1}$ | $d_{h,j}$ | $\cdots$ | $d_{h,i-1}$ | $d_{h,k}$ | $\cdots$ | $d_{h,n}$ |
| $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | 0 | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ |
| $k-1$ | $d_{k-1,1}$ | $\cdots$ | $d_{k-1,j-1}$ | $d_{k-1,h}$ | $\cdots$ | 0 | $d_{k-1,i}$ | $\cdots$ | $d_{k-1,h-1}$ | $d_{k-1,j}$ | $\cdots$ | $d_{k-1,i-1}$ | $d_{k-1,k}$ | $\cdots$ | $d_{k-1,n}$ |
| $i$ | $d_{i,1}$ | $\cdots$ | $d_{i,j-1}$ | $d_{i,h}$ | $\cdots$ | $d_{i,k-1}$ | 0 | $\cdots$ | $d_{i,h-1}$ | $d_{i,j}$ | $\cdots$ | $d_{i,i-1}$ | $d_{i,k}$ | $\cdots$ | $d_{i,n}$ |
| $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | 0 | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ |
| $h-1$ | $d_{h-1,1}$ | $\cdots$ | $d_{h-1,j-1}$ | $d_{h-1,h}$ | $\cdots$ | $d_{h-1,k-1}$ | $d_{h-1,i}$ | $\cdots$ | 0 | $d_{h-1,j}$ | $\cdots$ | $d_{h-1,i-1}$ | $d_{h-1,k}$ | $\cdots$ | $d_{h-1,n}$ |
| $j$ | $d_{j,1}$ | $\cdots$ | $d_{j,j-1}$ | $d_{j,h}$ | $\cdots$ | $d_{j,k-1}$ | $d_{j,i}$ | $\cdots$ | $d_{j,h-1}$ | 0 | $\cdots$ | $d_{j,i-1}$ | $d_{j,k}$ | $\cdots$ | $d_{j,n}$ |
| $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | 0 | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ |
| $i-1$ | $d_{i-1,1}$ | $\cdots$ | $d_{i-1,j-1}$ | $d_{i-1,h}$ | $\cdots$ | $d_{i-1,k-1}$ | $d_{i-1,i}$ | $\cdots$ | $d_{i-1,h-1}$ | $d_{i-1,j}$ | $\cdots$ | 0 | $d_{i-1,k}$ | $\cdots$ | $d_{i-1,n}$ |
| $k$ | $d_{k,1}$ | $\cdots$ | $d_{k,j-1}$ | $d_{k,h}$ | $\cdots$ | $d_{k,k-1}$ | $d_{k,i}$ | $\cdots$ | $d_{k,h-1}$ | $d_{k,j}$ | $\cdots$ | $d_{k,i-1}$ | 0 | $\cdots$ | $d_{k,n}$ |
| $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | 0 | $\cdots$ |
| $n$ | $d_{n,1}$ | $\cdots$ | $d_{n,j-1}$ | $d_{n,h}$ | $\cdots$ | $d_{n,k-1}$ | $d_{n,i}$ | $\cdots$ | $d_{n,h-1}$ | $d_{n,j}$ | $\cdots$ | $d_{n,i-1}$ | $d_{n,k}$ | $\cdots$ | 0 |

**(b)** The resulting NDSM after exchanging block $B_J$ and block $B_H$

**Figure D.4 The scenario used in the proof of Theorem 6.4**

It is clear that the different super-diagonal items between (6.10) and (6.11) are $B_{J,3}$, $B_{J,H}$, $B_{3,H}$ and $B_{3,J}$, $B_{H,J}$, $B_{H,3}$. In Figure D.4(a) and Figure D.4(b), $B_{J,3}$, $B_{J,H}$, $B_{3,H}$ are highlighted in grey, and $B_{3,J}$, $B_{H,J}$, $B_{H,3}$ are highlighted in blue. Let $J=(j,...,i-1)$, $I=(i,...,h-1)$ and $H=(h,...,k-1)$. Subtracting the feedbacks in the NDSM of Figure D.4(a) from the feedbacks in the NDSM Figure D.4(b), we get $\omega_J^H = \sum_{r\in J}\sum_{p\in I}\left(d_{r,p}-d_{p,r}\right)+\sum_{r\in J}\sum_{p\in H}\left(d_{r,p}-d_{p,r}\right)+\sum_{r\in I}\sum_{p\in H}\left(d_{r,p}-d_{p,r}\right)$. Thus, if $\omega_J^H > 0$, then exchanging block $B_J$ and block $B_H$ results in a feedback reduction of $\omega_J^H$.

# APPENDIX E

# PROOFS OF CHAPTER 7

**Proof of P1**

For two fuzzy triangular numbers $\widetilde{A}$ and $\widetilde{B}$ that are described by $(a_1,a_2,a_3)$ and $(b_1,b_2,b_3)$, their $\alpha$-cuts are: $\left[(a_2-a_1)\alpha+a_1,a_3-(a_3-a_2)\alpha\right]$, and $\left[(b_2-b_1)\alpha+b_1+b_3-(b_3-b_2)\alpha\right]$, respectively. From (7.4), we get:

$$D(\widetilde{A})=\frac{1}{2}\int_0^1\left[(a_2-a_1)\alpha+a_1+a_3-(a_3-a_2)\alpha\right]d\alpha=\frac{a_1+2a_2+a_3}{4} \tag{7.13}$$

$$D(\widetilde{B})=\frac{1}{2}\int_0^1\left[(b_2-b_1)\alpha+b_1+b_3-(b_3-b_2)\alpha\right]d\alpha=\frac{b_1+2b_2+b_3}{4} \tag{7.14}$$

Let $\widetilde{C}=\widetilde{A}\oplus\widetilde{B}$. From (7.2) and (7.4), we get:

$$\widetilde{C}=\widetilde{A}\oplus\widetilde{B}=(a_1+b_1,a_2+b_2,a_3+b_3) \tag{7.15}$$

$$D(\widetilde{C})=\frac{1}{2}\int_0^1\left[(a_2+b_2-a_1-b_1)\alpha+a_1+b_1+a_3+b_3-(a_3+b_3-a_2-b_2)\alpha\right]d\alpha$$

$$=\frac{a_1+b_1+2(a_2+b_2)+a_3+b_3}{4} \tag{7.16}$$

Clearly, $D\left(\widetilde{C}\right)=D(\widetilde{A})+D(\widetilde{B})$.

**Proof of P2**

Let $r$ be a real number, which can be represented as $(r,r,r)$. From (7.3) and (7.4), we get:

$$r\widetilde{A}=(ra_1,ra_2,ra_3) \tag{7.17}$$

$$D(r\widetilde{A}) = \frac{1}{2}\int_0^1 \left[(ra_2 - ra_1)\alpha + ra_1 + ra_3 - (ra_3 - ra_2)\alpha\right] d\alpha = r\frac{a_1 + 2a_2 + a_3}{4} \tag{7.18}$$

From (7.13) and (7.18), we get $D(r\widetilde{A}) = rD(\widetilde{A})$.

**Proof of P3**

Let $r_1$ and $r_2$ be two real numbers, which can be expressed as $(r_1, r_1, r_1)$ and $(r_2, r_2, r_2)$, respectively. Let $\widetilde{F} = r_1\widetilde{A} \oplus r_2\widetilde{B}$, from (7.2)-(7.4), we get:

$$\widetilde{F} = r_1\widetilde{A} \oplus r_2\widetilde{B} = (r_1a_1 + r_2b_1, r_1a_2 + r_2b_2, r_1a_3 + r_2b_3) \tag{7.19}$$

$$D(\widetilde{F}) = \frac{1}{2}\int_0^1 \left[(r_1a_2 + r_2b_2 - r_1a_1 - r_2b_1)\alpha + r_1a_1 + r_2b_1 + r_1a_3 + r_2b_3 - (r_1a_3 + r_2b_3 - r_1a_2 - r_2b_2)\alpha\right] d\alpha$$

$$= \frac{r_1a_1 + r_2b_1 + 2(r_1a_2 + r_2b_2) + r_1a_3 + r_2b_3}{4} \tag{7.20}$$

From (7.13) and (7.14), we get:

$$r_1D(\widetilde{A}) + r_2D(\widetilde{B}) = r_1\frac{a_1 + 2a_2 + a_3}{4} + r_2\frac{b_1 + 2b_2 + b_3}{4} \tag{7.21}$$

Inspection of (7.20) and (7.21) shows that $D(\widetilde{F}) = r_1D(\widetilde{A}) + r_2D(\widetilde{B})$.