PROCESS OPERATIONS WITH UNCERTAINTY AND INTEGRATION CONSIDERATIONS

by

ZUKUI LI

A Dissertation submitted to the

Graduate School-New Brunswick

Rutgers, The State University of New Jersey

in partial fulfillment of the requirements

for the degree of

Doctor of Philosophy

Graduate Program in Chemical and Biochemical Engineering

written under the direction of

Prof. Marianthi G. Ierapetritou

and approved by

_____

_____

_____

_____

New Brunswick, New Jersey

[May, 2010]

ABSTRACT OF THE DISSERTATION

Process Operations with Uncertainty and Integration Considerations

By ZUKUI LI

Dissertation Director:
Prof. Marianthi Ierapetritou

There has been a lot of attention in recent years towards the application of mathematical modeling and optimization approaches for the solution of production planning and scheduling problems. This is mainly due to the changing economic environment which pushes for more efficient process operations. However, there are still a number of challenges that restrict the effective application of optimization for planning and scheduling problem especially in the process industry. First, decision making in process operations is frequently based on parameters of which the values are uncertain. A systematic treatment of those uncertainties (e.g., processing time variations, rush orders, failed batches, machine breakdowns, etc) is necessary to satisfy the customer demands, increase the efficiency of operations and improve the plant profitability. Moreover, the interactions between the different decision-making levels were often ignored in existing solution approaches, which leads to sub-optimal and even infeasible solutions. Thus the integration of different decision making levels has been recognized by the research community as an imperative problem.

In this work, systematic methods have been developed to address the above challenges. First, different methodologies are proposed to address the uncertainties in process scheduling problem: robust optimization based preventive scheduling strategy which aims at generating a robust preventive schedule to handle the possible parameter perturbations; parametric programming based reactive scheduling strategy which aims at responsive schedule regeneration or updating upon the happening of disruptive events. To address the interaction between planning and scheduling decision levels, a dual decomposition based approach that targets the solution of large scale planning and scheduling integration problem was proposed, which aims at decreasing the computational complexity through decomposition and parallel computation. Finally, the rolling horizon method which provides a promising modeling framework for integrated planning and

scheduling and incorporation of uncertainty is studied. A novel method of generating production capacity model through parametric programming technique is proposed, and it is verified that the incorporation of the capacity model into the rolling horizon framework can improve the solution quality.

Acknowledgements

I would like to express my deepest and most sincere gratitude to my supervisor, Prof. Marianthi Ierapetritou. Her wide knowledge and her logical way of thinking have been of great value for me. Her understanding, encouraging, personal guidance and important support throughout my Ph.D. study have provided a good basis for the present dissertation.

For this dissertation I would like to thank my thesis committee members: Dr. Ioannis Androulakis, Dr. Yee C. Chiew and Dr. David W. Coit for their time, interest, and helpful comments.

I gratefully acknowledge the funding support from National Science Foundation under grant CBET-0625515, 0966861 that made my Ph.D. work possible.

My time at Rutgers was made enjoyable in large part due to the many friends that became a part of my life. I am grateful for time spent with group members who have been great support to my study and also shared with the fantastic PhD life: Zhenya Jia, Steve Guzikowski, Eddie Davis, Patricia Portillo, Hong Yang, Kaiyuan He, Yijie Gao, Mehmet Orman, Nikisha Shah, Fani Boukouvala, Vidya Iyer, Beverly Smith, Shuliang Zhang.

Lastly, I would like to thank my family for all their love and encouragement: for my parents who raised me with a love of science and supported me in all my pursuits, for the constant support from my sister Zuqing, and most of all for my loving, supportive, encouraging, and patient wife Qian whose faithful support during all stages of this Ph.D. is so appreciated. Thank you.

To my parents: *Dingxiang Wang* and *Gaotong Li*

Table of Contents

List of tables

List of illustrations

Chapter 1

Introduction

1.1 Planning and scheduling in the process industry

Modern process industry (e.g., chemical, food, pharmaceutical, refineries, etc.) faces major new challenges through increased global competition, greater regulatory pressures and uncertain prices of energy, raw materials and products. These competitive concerns increase the focus on integrated processes, information technology, and consideration of multiple decision criteria including profitability, flexibility, service quality and the production efficiency. The success of process industry largely depends on how efficiently it generates value by dynamically optimizing deployment of its supply chain resources. Among the challenges for the dynamic optimization of the entire supply chain resources are the rigorous but tractable optimization of process operations and the efficient integration of different decision making stages as well as the consideration of uncertainty and risk factors. Planning and scheduling deal with the allocation of available resources over time to perform a set of tasks required to manufacture one or more products, and they are the most important topics in process operations (Grossmann & Westerberg, 2000).

Planning problem corresponds to a higher level of process operation decision making since it considers longer time horizon and multiple orders that involve different operating conditions as well as unit changes, price and cost variability. Planning in process industry is used to create production, distribution, sales and inventory plans based on customer and market information while observing all relevant constraints. In particular, operational plans have to be determined which are aimed to structure future production, distribution and other related activities according to business objectives. Based on these operational plans, detailed schedules are worked out which define the precise timing and sequencing of individual operations as well as the assignment of the required resources over time. Production planning provides the decision support systems for the logistics in the long range operation of networks of plants, and their coordination with marketing and business considerations. A higher level of planning is supply chain planning/management

(Kallrath, 2002). In supply chain planning, we usually consider material flow and balance equations connecting sources and sinks of a supply network. Time-indexed models using a relative coarse discretization of time, e.g., a year, quarters, months or weeks are usually accurate enough. Process industry supply chains, involving manufacturers, suppliers, retailers and distributors, strives to improve efficiency and responsiveness. Supply chain planning considers a fixed infrastructure over a short- to medium-term, and seeks to identify how best to use the production, distribution and storage resources in the chain to respond to orders and demand forecasts in an economically efficient manner. Supply chain planning provides the decision support systems for the logistics in the long range operation of networks of plants, and their coordination with marketing and business considerations. These problems often give rise to large multi-period optimization problems where a major challenge lies in the effective aggregation of more detailed scheduling and operational models (Shah, 2005).

Process scheduling addresses the optimal assignment of tasks to units over the allotted time horizon in the operations of multiproduct and multipurpose plants that manufacture a variety of products through several sequences of operations operate in batch and/or continuous mode. Scheduling of batch and continuous processes can have a major impact on the overall profitability of a process, as well as on the timely delivery of products. Major problems include sequencing, scheduling of equipment utilization and maintenance over a planning horizon, and inventory considerations of a process. Such problems form perhaps difficult combinatorial optimization problems but also contribute to high payoffs. In an industrial process, each task requires certain amounts of specified resources for a specific time interval called the processing time. The resources include the use of equipment, the utilization of raw material or intermediates, the employment of operators etc., and tasks involve the chemical or physical transformation of materials, transportation of products or intermediates, cleaning, and maintenance operations etc. Scheduling objective can take many forms such as minimizing the time required to complete all the tasks (the makespan), minimizing the number of orders completed after their committed due dates, maximizing customer satisfaction by completing orders in a timely fashion, maximizing plant throughput, maximizing profit or minimizing production costs. Scheduling decisions to be determined include the optimal sequence of tasks taking place in each unit, the amount of material being processed at each time in each unit and the processing time of each task in each unit.

Planning and scheduling can be distinguished based on various characteristics. First in terms of the considered time horizon, long-term planning problems deal with longer time horizons (e.g., months or years) and are focused on higher level decisions such as timing and locations of additional facilities and levels of production. The area of medium-term scheduling involves medium time horizons (e.g., weeks or months) and aims to determine detailed production schedules, which can result in very large scale problems. Short-term scheduling models address short time horizons (e.g., days or weeks) and are focused on determining detailed sequencing of various operational tasks. Second in terms of the decisions involved, short-term scheduling provides feasible production schedule considering the detailed operation conditions; while planning involves consideration of financial and business decisions over extended periods of time. Lastly considering uncertainty, short-term scheduling need to consider the disturbing events such as rush orders, machine breakdown and attempt to absorb the impact; while the planning need to foresee the possible changes in the future and the effects of the current decisions thus achieving an optimal solution for the benefits of the entire planning time horizon. Planning and scheduling of process systems are also closely linked activities. Production planning determines the optimal allocation of resources within the production facility over a time horizon of a few weeks up to few months, scheduling provides the feasible production schedules to the plant for every day operations. Since the boundaries of planning and scheduling problems are not well established and there is an intrinsic integration between these decision making stages.

1.2 Modeling and optimization for planning and scheduling

In the past, planning especially scheduling operations in the industry are mostly based on heuristics (Kallrath, 2002), (Elliott, 2000), mathematical programming based modeling and optimization become more and more the state-of-the-art in the planning and scheduling operations for the process industry.

Most of the planning problems in the process industry lead to linear programming (LP) or mixed integer linear/nonlinear programming (MILP/MINLP) models and contain the following building blocks: tracing the states of plants, modeling production, balance equations for material flows, transportation terms, consumption of utilities, cost terms, and special model features. Mode-changes, start-up and cancellation features, and nonlinear cost structures require many binary variables. Minimum utilization rates and transportation often require semi-continuous variables. Special features such as batch and campaign

constraints across periods require special constraints to implement the concept of contiguity. The model, however, remains linear in all variables. Only if the pooling problem occurs, e.g., in the refinery industry or the food industry, we are really facing a MINLP problem. In production or supply chain planning, time-indexed models using a relative coarse discretization of time, e.g., a year, quarters, months or weeks are usually accurate enough. The MILP/MINLP approaches are also often appropriate and successful for problems with a clear quantitative objective function (net profit, contribution margin, cost, total sales neglecting cost, total production for a fixed system of production reactors, energy consumption or the usage of other utilities, deviation of the usage of resources from their average usage), or quantitative multi-criteria objectives usually a subset of those just listed.

There are lots of different approaches that appear in the literature to address the problem of scheduling formulation, a recent review about classification of scheduling problems is given by (Méndez et al., 2006). One major classification is based on the nature of the production facility to manufacture the required number of products utilizing a limited set of units. If every job consists of the same set of tasks to be performed in the same order and the units are accordingly arranged in production lines, the problem is classified as a multiproduct plant (also called flow-shop problem). If production orders have different routes (require different sequences of tasks) and some orders may even visit a given unit several times, the problem is known as multipurpose plant (also called job-shop problem). A number of alternative ways of formulating the scheduling problem exist in the open literature. One distinguishing characteristic is the time representation, according to which the approaches are classified into two broad categories. Early attempts of formulating the scheduling problem were mainly concentrated on the discrete-time formulation, where the time horizon is divided into a number of intervals of equal duration. The other type of method aims at developing efficient methods based on a continuous-time representation, a thorough review is given by (Floudas & Lin, 2004).

1.3 Problems and challenges

Although there has been a lot of attention in recent years towards the application of mathematical modeling and optimization approaches for the solution of production planning and scheduling problems, there are still a number of challenges that restrict the effective application of optimization for planning and scheduling problem in the process industry.

1.3.1 Uncertainty issue

First, most of the work in the area of planning and scheduling deals with the deterministic optimization model where all the parameters are considered known. Along with the studies in deterministic planning and scheduling, consideration of uncertainties in these problems has got more attention in recent years. In real plants uncertainty is a very important concern that is coupled with the planning and especially scheduling process since many of the parameters that are associated with them are not known exactly. Parameters like raw material availability, prices, machine reliability, and market requirements vary with respect to time and are often subject to unexpected deviations. Having ways to systematically consider uncertainty is as important as having the mathematical model itself.

Uncertainty appears in all the different levels of the industry from the detailed process description to multi-site manufacturing, such as demand or changes in product orders or order priority, batch or equipment failures, processing time variability, resource changes and/or recipe variations, etc. Based on the nature of the source of uncertainty in a process, a suitable classification has been proposed by (Pistikopoulos, 1995) as follows: (i) Model-inherent uncertainty, such as kinetic constants, physical properties, mass/heat transfer coefficients; (ii) Process-inherent uncertainty, such as flow rate and temperature variations, stream quality fluctuations, processing time and equipment availability; (iii) External uncertainty, including feed stream availability, product demands, prices and environmental conditions; (iv) Discrete uncertainty, such as equipment availability and other random discrete events, operational personnel absence. To include the description of uncertain parameters within the optimization model of the planning and scheduling problem, several methods have been used: bounded form; probability distribution function and fuzzy description. Following the alternative description methods for uncertainty, different scheduling models and optimization approaches have been developed.

Methodologies for process scheduling under uncertainty aim at producing feasible, robust and optimal schedules. According to the different treatment of uncertainty, process scheduling methods can be classified into two groups: reactive scheduling and preventive scheduling. Reactive scheduling is a process of modifying the existing schedule during the process operation to adapt to changes (uncertainty) in production environment, such as disruptive events, rush order arrivals, order cancellations or machine breakdowns. Preventive scheduling on the other hand generates scheduling policies before uncertainty occurs. Detail

classification of preventive scheduling includes: stochastic scheduling, robust optimization method, fuzzy programming method and sensitivity analysis and parametric programming method.

The inability of many scheduling systems to address the general issue of uncertainty is cited as a major reason for the lack of influence of scheduling research in industrial practice. In the commercial planning and scheduling systems, there exist few generic packages for risk analysis to support the resolution of optimization problems under uncertainty. Most tools claim to provide real-time scheduling capabilities and what if scenario analysis. In general, they are able to generate updated schedules as disruptions occur, and use interactive Gantt charts which allow to drag and drop operations for manual rescheduling; however, the incorporation of robustness issues within a systematic procedure is not considered at all.

1.3.2 Integration issue

Production planning and scheduling belong to different decision making levels in process operations, they are also closely related since the result of planning problem is the production target of scheduling problem. The planning problem deals with longer term issues compared to scheduling with the emphasis being on the optimization of production capacity minimizing cost. In the process industry, the most commonly used planning and scheduling decision making strategy follows a hierarchical approach, in which the planning problem is solved first to define the production targets and the scheduling problem is solved next to meet these targets. However, the main issue with this traditional strategy is the lack of communication between the two decision levels, i.e., the planning decisions generated might cause infeasible schedule subproblems. Thus at the planning level, the effects of changeovers and daily inventories are neglected, which tends to produce optimistic estimates that cannot be realized at the scheduling level, i.e., a solution determined at the planning level does not necessarily lead to feasible schedules. Moreover, the optimality of the planning solution cannot be ensured because the planning level problem might not provide an accurate estimation of the production cost, which should be calculated based on the details of the scheduling problem.

Therefore, it is important and necessary to develop methodologies that can effectively integrate production planning and scheduling. However, since production planning and scheduling are dealing with different time scales, the major challenge towards the integration is dealing with the big computational complexity associated with the resulted optimization problem. To overcome the above difficulty, most of the

work appeared in the literature aim at decreasing the problem scale through different types of problem reduction methodologies and developing efficient solution strategies as summarized by (Grossmann et al., 2002). In this chapter, we are aiming of reviewing the existing work in the area of planning and scheduling integration as classified in the following subsections.

1.4 Project objectives

To address the challenges for planning and scheduling operations as stated in previous subsections, the first general research objective is to develop systematic method in addressing the uncertainty in process operations and to help the decision maker better *understand*, *manage* and *tackle* uncertainty. The second part of the research objective is related to the integrated planning and scheduling decision making. Specifically, the following research objectives will be studied:

(1) Developing systematic uncertainty analysis framework with parametric programming technique,

(2) Developing effective preventive scheduling which uses robust optimization formulation,

(3) Developing new reactive scheduling technique,

(4) Developing efficient decomposition based methodology to address large scale integrated planning and scheduling optimization problems.

(5) Consideration of production capacity model in the rolling horizon solution framework.

The above objectives are studied through different strategies in this thesis as follows:

Specific objective 1: Parametric programming based uncertainty analysis method

In real plants, uncertainty is a very important concern that is coupled with the scheduling process since many of the parameters that are associated with scheduling are not known exactly. Parameters like raw material availability, prices, machine reliability, and market requirements vary with respect to time and are often subject to unexpected deviations. To better understand uncertainty, the effect of different uncertainties on planning/scheduling performance should be provided. To address this issue, a parametric programming based uncertainty analysis method for MILP problem is proposed and applied to the scheduling problem. This part of work will be elaborated in Chapter 2.

Specific Objective 2: Preventive scheduling – efficient robust schedule generation

Recognizing the fact that real plants exist in a dynamic environment where the scheduling parameters change as the schedule is being executed, a schedule developed beforehand may become inefficient or even infeasible. The uncertainty considered in this work can be viewed as fluctuations in product demands, prices and processing times, etc. The expected range of parameter uncertainty is incorporated within the short-term scheduling model so as to determine a robust schedule capable of meeting the expected range of uncertain parameter values. In this specific research, a new robust scheduling formulation is proposed which will avoid these drawbacks of existing methods. Chapter 3 covers this part of research.

Specific Objective 3: Reactive scheduling

The ability to react to unexpected events during the execution of a schedule is an important part of plant operating strategy. The uncertainty considered in this direction includes rush order, order cancellation and unexpected deviations in unit availability (machine break-down). Current capabilities of optimization methods to reactive scheduling problems are still very restricted and mostly focused on sequential batch processes. More general, efficient and systematic rescheduling tools are required for recovering feasibility and efficiency with short reaction time and minimum additional cost. The objective of reactive scheduling is to determine the optimal rescheduling policy that minimizes the deviations from the old schedule while taking into account the satisfaction of other production constraints. The main effort should be oriented towards avoiding a time-expensive full scale rescheduling, allowing during the rescheduling process only limited changes to the scheduling decisions already made at the beginning of the time horizon. In this direction, we will mainly aim at developing effective reactive scheduling method to avoid the high computational effort needed by the traditional re-scheduling framework. Chapter 4 discussed more detail of this work.

Specific Objective 4: Solution strategy for integrated planning and scheduling

In the past, the problem integrated planning and scheduling is not well addressed. The main reason is that many realistic industrial integrated planning and scheduling problems are large scale discrete optimization problems. The objective of this research is to develop efficient decomposition based methodology to solve the large scale full space integrated planning and scheduling problem. This part of work is presented in chapter 5.

Specific Objective 5: Rolling horizon framework for integrated planning and scheduling

The final objective is to develop an integrated methodology to handle uncertainty in integrated planning and scheduling problem. Rolling horizon method provides a promising framework for this objective. However, the most existing rolling horizon method does not consider the quality of the solution because of lacking the capability in modeling the production capacity information. Basically, we proposed a novel method to generating the accurate production capacity information for short term scheduling problem, which can be further incorporated into the planning level and also to improve the quality of the final solution. This has been verified through its application in rolling horizon based solution method. This work is explained in Chapter 6.

Chapter 2

Uncertainty Analysis with Parametric Programming

Abstract: In this chapter, a novel parametric programming algorithm is proposed to address the uncertainty in the right hand side (RHS), left hand side (LHS) and objective function of a mixed integer linear programming problem. The problem of process scheduling under uncertainty was further studied using the proposed parametric programming method.

2.1 Introduction

In real plants, uncertainty is a very important concern that is coupled with the scheduling process since many of the parameters that are associated with scheduling are not known exactly. Parameters like raw material availability, prices, machine reliability, and market requirements vary with respect to time and are often subject to unexpected deviations. Having ways to systematically consider uncertainty is as important as having the scheduling model itself. In essence, uncertainty consideration plays the role of validating the use of mathematical models and preserving plant feasibility and viability during operations.

Stochastic optimization is the most commonly used approach in the literature for scheduling under uncertainty (Balasubramanian & Grossmann, 2002; Bonfill et al., 2004; Bonfill et al., 2005; Ierapetritou & Pistikopoulos, 1996; Orçun et al., 1996; Petkov & Maranas, 1997), in which the original deterministic scheduling model is transformed into stochastic model treating the uncertainties as stochastic variables. Within the stochastic programming models we can distinguish the following categories: two-stage/multi-stage stochastic programming and chance constraint programming based approach. Fuzzy programming also addresses optimization problems under uncertainty and is applied in uncertain scheduling (Balasubramanian & Grossmann, 2003; Petrovic & Duenas, 2006; Wang, 2004). It can be used in the situation when probabilistic information is not available. Fuzzy set theory and interval arithmetic are used to describe the imprecision and uncertainties in process parameters. Robust optimization methods aims at

building the robust preventive schedule to minimize the effects of disruptions on the performance measure (Janak et al., 2007; Jia & Ierapetritou, 2007; Lin et al., 2004; Vin & Ierapetritou, 2001). It tries to ensure that the predictive and realized schedules do not differ drastically while maintaining a high level of schedule performance. Except the above methods, an alternative way in preventive scheduling is using MILP sensitivity analysis and parametric programming. These methods are important as they can offer significant analytical results to problems related to uncertainty. Sensitivity analysis is used to determine how a given model output depends upon the input parameters (Jia & Ierapetritou, 2004). Parametric programming serves as an analytic tool by mapping the uncertainties in the optimization problem to optimal solution alternatives. From this point of view, parametric programming provides the exact mathematical solution of the optimization problem under uncertainty.

In the literature, multiparametric linear programming (mpLP) and multiparametric quadratic programming (mpQP) problem are well studied due to the relatively smaller problem complexity (Bemporad et al., 2002; Borrelli et al., 2003; Johansen, 2002; Seron et al., 2000). General multiparametric nonlinear programming (mpNLP) problem is not well addressed because the exact solution of mpNLP is very complex (Acevedo & Salgueiro, 2003). On the other hand, existing multiparametric mixed integer programming methods are based on the solution of mpLP or mpQP subproblems (Acevedo & Pistikopoulos, 1997; Dua & Pistikopoulos, 1999). For the multiparametric mixed integer quadratic programming (mpMIQP), there is still not an efficient method for solving the general problem. Dua, Bozinis et al.(Dua et al., 2002) proposed a methodology to address this problem for the special case derived from optimal control problem.

In the past, the multiparametric programming method has been mainly applied in online optimization, process control, and process synthesis (Dua et al., 2002). All these problems are of relatively small scale. There are not many records on the application of parametric programming in process scheduling problem. To our knowledge, only Ryu and Pistikopoulos (Ryu & Pistikopoulos, 2007) has reported the application of parametric programming to a zero-waiting scheduling problem and Pistikopoulos et al.(Hugo & Pistikopoulos, 2005; Pistikopoulos & Dua, 1998) have applied parametric programming for the solution of process planning problem.

Formulating the scheduling problem under uncertainty as a multiparametric programming problem gives rise to mpMILP, mpMIQP or mpMINLP problem depending on the type of uncertain parameters. The

other important characteristic for multiparametric programming in scheduling formulation is that the problem is generally large scale, because the deterministic formulation of process scheduling problem involves a large number of constraints and integer variables.

In this chapter, we proposed a framework to solve the mpMILP and mpMIQP problems generated from uncertain process scheduling problem. The framework is based on the idea of decomposing the original problem into a series of smaller problems. The parametric solution of each subproblem provides the solution around a given set of parameter values. The chapter's structure is as following: in section 2.2, the parametric solution algorithm for general MILP problem is presented and also illustrated through a numerical example; in section 2.3, we illustrated its application in analyzing uncertainty for process scheduling problem; finally, the work is concluded in section 2.4.

2.2 Parametric programming algorithm

2.2.1 Problem definition

In this work, we are studying the following general mixed integer linear programming problem with possible uncertain parameter on the right hand side (RHS), or as coefficient of integer variables on the left hand side (LHS) or the objective function (Obj):

$$\min_{x,y} \quad (c + E\theta)x \tag{2.1a}$$

$$\text{s.t.} \quad Ax + (B + D\theta)y = b + E\theta \tag{2.1b}$$

$$x \geq 0,\ y \in \{0,1\} \tag{2.1c}$$

$$\theta \in [\theta^L, \theta^U] \tag{2.1d}$$

where y represents the binary decision variables; x represents the continuous variables, θ represent the uncertain parameters; $[\theta^L, \theta^U]$ represents an given range for those parameters θ. The objective is to identify the complete map of relationship between the optimal solution and the value of the uncertain parameter in the given parameter space. To describe the so-called relationship, the concept of "critical region" is used, which is defined as a region in the parameter space, in which a unique set of optimal integer solution and optimal parametric solution (represented by function of the uncertain parameters) exists.

2.2.2 Local parametric solution algorithm

Based on the above formulation, we present the parametric programming algorithm as follows (flowchart is shown in Figure 2.1):

Algorithm 1. Compute parametric solution around a given point θ^0

Step 1. Fix $\theta = \theta^0$, solve problem (2.1) and get the optimal integer solution y^* .

Step 2. With $y = y^*$, formulate relaxed linear programming problem based on (2.1) and compute

a) the optimal solution $x^*(\theta)=A_B^{-1}(b - By^* + E\theta - D\theta y^*)$ (2.2)

b) the optimal value function $f^*(\theta) = cA_B^{-1}(b - By^* + E\theta - D\theta y^*)$, (2.3)

c) the initial region $CR^* = \{\theta \mid A_B^{-1}(b - By^* + E\theta - D\theta y^*) \geq 0, \theta^L \leq \theta \leq \theta^U\}$, (2.4)

where the sub-index B represent the basis index for the linear programming problem.

Step 3. Solve the following problem (θ is treated as decision variable here):

$$\max_{x,y,\theta} \quad err = f^*(\theta) - (c + E\theta)x \tag{2.5a}$$

s.t. (2.1b)-(2.1d)

$$err \leq \varepsilon \tag{2.5b}$$

$$\theta \in CR^* \tag{2.5c}$$

If the optimal objective $err^* \leq 0$, return ($x^*(\theta)$, $f^*(\theta)$, CR^*) as the parametric solution around θ^0 and stop. Otherwise, store the solution (y' , θ') of problem (2.5) and go to step 4.

Step 4. Fix $\theta = \theta'$, $y = y'$, solve problem (2.5), identify a new set of basis index B' and get the following optimal value function and critical region

$$f'(\theta) = c_{B'}A_{B'}^{-1}(b - By' + E\theta - D\theta y') \tag{2.6}$$

$$CR' = \{\theta \mid A_{B'}^{-1}(b - By' + E\theta - D\theta y') \geq 0, \theta^L \leq \theta \leq \theta^U\} \tag{2.7}$$

Step 5. Define $CR^{EX} = CR^* \cap CR' \cap \{\theta \mid f'(\theta) \leq f^*(\theta)\}$, update CR^* by excluding CR^{EX} from it, then go to step 3.

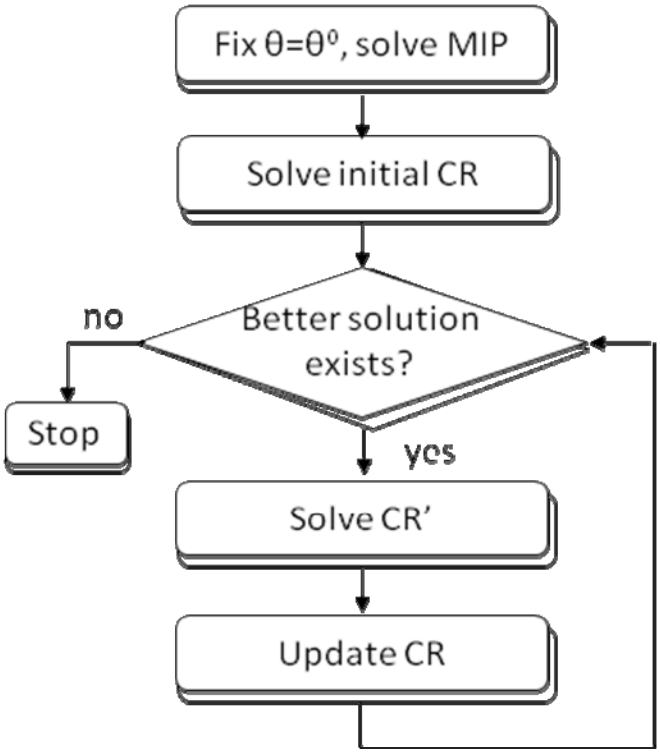


Figure 2.1 Flowchart of the parametric MILP algorithm

The correctness of the algorithm 1 is provided by the following theorem.

Theorem. The output of algorithm 1 ($x^*(\theta)$, $f^*(\theta)$) is the optimal solution and optimal value function of problem (2.1) in CR^* that covers θ^0 .

Proof: First, the solution $x^*(\theta)$ derived from (2.2) and the value function $f^*(\theta)$ derived from (2.3) must be the optimal solution at point θ^0 because they satisfy the optimality conditions of relaxed LP problem, which is derived from (2.1) at the optimal integer solution at point θ^0 .

Second, the solution ($x^*(\theta)$, $f^*(\theta)$) keeps its optimality in the final CR^* with the following reasons. Except all the original constraints of (2.1), problem (2.5) includes a new objective (2.5a) to seek the parameter value that can provide a better (smaller) value function, (2.5b) is used to stop the solution once a better value function is found, so ε is set as a small positive number; (2.5c) is a restriction of the solution space to current critical region CR^* . So problem (2.5) can be solved to check whether the value function $f^*(\theta)$ keeps its optimality for all the θ values in the region CR^* . If the optimal objective $err^* \le 0$, it means

that the optimality of $f^*(\theta)$ in the whole region is ensured because no better value function can be found. Otherwise, it means that at the solution point θ', the optimal value function should be a different one other than $f^*(\theta)$, then the current critical region CR^* needs to be updated by excluding the region around θ' that takes this better value function, this region is CR^{EX} as defined in step 5. With the update on the critical region, the final region will take a unique value function which is ensured to be optimal. Thus the correctness of the algorithm is proved. □

Remarks

(1) The exclusion operation in step 5 is performed through selecting one constraint in CR^{EX} that θ^0 violates, and add this constraint with reversed inequality sign to CR^*.

(2) Depending on the source of the uncertainty: objective function (Obj), Right hand side (RHS) of the constraints, constraint matrix on the Left hand side (LHS), the optimal parametric objective value function will be quadratic, the parametric solution will be either linear, quadratic or nonlinear fractional function, they are summarized as Table 2.1.

Table 2.1 Summary of the characteristic of parametric solution

	$f^*(\theta)$	$x^*(\theta)$	CR
RHS	Linear $cA_B^{-1}(b-By^*+E\theta)$	Linear $A_B^{-1}(b-By^*+E\theta)$	Linear
Obj	Linear $(c+F\theta)A_B^{-1}(b-By^*)$	Constant $A_B^{-1}(b-By^*)$	Linear
LHS*	Linear $cA_B^{-1}(b-By^*-D\theta y^*)$	Linear $A_B^{-1}(b-By^*-D\theta y^*)$	Linear
Obj+RHS	Quadratic $(c+F\theta)A_B^{-1}(b-By^*+E\theta)$	Linear $A_B^{-1}(b-By^*+E\theta)$	Quadratic Linear
Obj+LHS*	Quadratic $(c+F\theta)A_B^{-1}(b-By^*-D\theta y^*)$	Linear $A_B^{-1}(b-By^*-D\theta y^*)$	Quadratic Linear
RHS+LHS*	Quadratic $cA_B^{-1}(b-By^*+E\theta-D\theta y^*)$	Linear $A_B^{-1}(b-By^*+E\theta-D\theta y^*)$	Quadratic Linear
Obj+RHS+LHS*	Quadratic $(c+F\theta)A_B^{-1}(b-By^*+E\theta-D\theta y^*)$	Linear $A_B^{-1}(b-By^*+E\theta-D\theta y^*)$	Quadratic Linear

$f^*(\theta)$: optimal parametric objective value function; $x^*(\theta)$: Optimal parametric solution
CR: critical region; LHS*: uncertainties only appear as coefficients of integer variables

(3) Notice that here we are only considering the LHS uncertainty in the coefficients of integer variables, for the general LHS uncertainty appear also as coefficient of continuous variable, the optimal parametric

objective function, optimal parametric solution and critical region will be represented by nonlinear fractional function.

(4) The proposed parametric programming algorithm has been extended to address the general parametric Linear Complementarity Problem (Appendix A).

2.2.3 Exploring the parameter space

Based on the above algorithm for computing local parametric solution around a given point, we can develop the algorithm to compute the complete parametric solution map. As pointed out in Table 2.1, the critical region can be described by quadratic and/or linear constraints depending on the position of the uncertain parameters. For the case of quadratic constraints, the region might be nonconvex and it is hard to find an exact method to partition the original parameter space, so currently only random testing method is proposed to explore the parameter space, i.e., the initial given point is randomly generated inside the parameter space, then a testing step is applied to check whether it is already covered by identified critical regions, then the local parametric solution algorithm will be applied.

In the following, we are presenting an exact method for the case that the critical region is described by linear constraints. The whole parameter space is initially described as an unexplored region. Every time a new critical region is identified inside an unexplored region in the parameter space, this region is further partitioned to identify the unexplored areas. The unexplored area of the parameter space is represented by the union of a set of "unexplored regions". The process is repeated until all of the unexplored areas in the parameter space have been studied. The detail algorithm is as follows:

Algorithm 2. Compute complete parametric solution map

Step 1. Set initial *unexplored region set* $R = \{[\theta^L, \theta^U]\}$, and *identified critical region set* S=$\varnothing$.

Step 2. If *R* is empty, stop. Otherwise, arbitrarily select one region *r* from set R, call the local parametric solution algorithm to compute a critical region *CR* around an point inside this region. Store the critical region CR into S : $S = S \cup \{CR\}$.

Step 3. Partition the region r and identify the unexplored region in r. Store them into *R* and delete *r* from *R*. Go to step 2.

In the step 2 of algorithm 2, it is necessary to identify the remaining part of a given region based on the identified critical regions inside it. This can be achieved following the method described by (Dua & Pistikopoulos, 2000). For example, for a two-dimension parameter space, assume that the initial unexplored region is r_0, a critical region CR is identified inside it, and CR is described as follows:

$$\text{CR}= \{\, \theta \,|\, c_1(\theta) \le 0, c_2(\theta) \le 0, c_3(\theta) \le 0 \,\},$$

where c_1, c_2, c_3 represent linear constraint function of θ. The partition procedure considers one by one the inequalities that define CR. For example, considering the constraint $c_1(\theta) \le 0$, the first new unexplored region is given by: $r_1 = \{\theta \,|\, \theta \in r_0, c_1(\theta) \ge 0\}$, which is obtained by reversing the sign of the inequality $c_1(\theta) \le 0$, adding it to the constraints of r_0 and removing redundant constraints. Thus, by considering the rest of the inequalities one by one, the complete unexplored region in region r is given by $r_1 \cup r_2 \cup r_3$

$$r_2 = \{\theta \,|\, \theta \in r_0, c_1(\theta) \le 0, c_2(\theta) \ge 0\},\ r_3 = \{\theta \,|\, \theta \in r_0, c_1(\theta) \le 0, c_2(\theta) \le 0, c_3(\theta) \ge 0\}$$

These region are depicted in Figure 2.2.

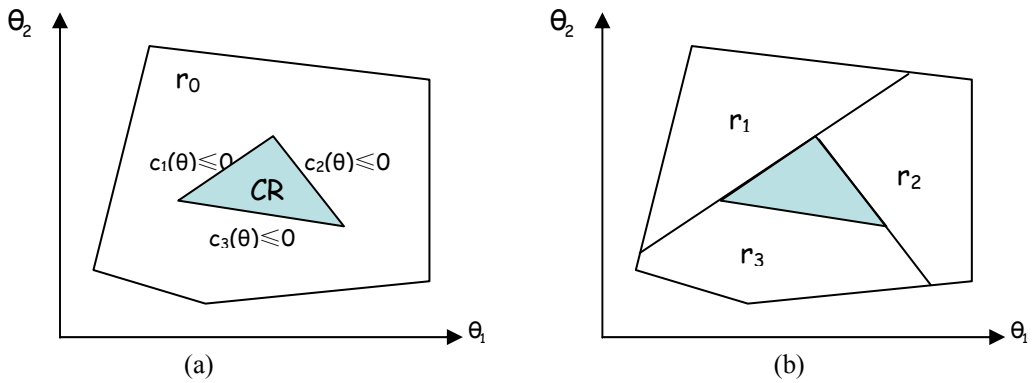


Figure 2.2 Illustration of identifying the remaining part of a given region.
a) A given region and an identified critical region inside this
b) Partition of the given region into new unexplored region

At the end of algorithm 2, a complete map of all critical regions is obtained. Each critical region is associated with a corresponding parametric solution as expressed in (2.2)-(2.3).

Finally, it is worth to point out that the number of critical regions is mostly related to the size of the parameter space which is determined by its dimension (the number of uncertain parameters) and the length in every dimension (the range for every uncertain parameter). Thus when the parameter space is large, the

number of the critical regions can be also large. However, in this work, we are focusing on the algorithm for generating this parametric solution information and do not study the parametric solution information storage problem.

The proposed method has been implemented in GAMS and MATLAB, where MATLAB is used to formulate the standard form of problem based on GAMS file, to control the flow and to calculate the optimality conditions. CPLEX 10.1 is used to solve MILP. In the following, problems are all solved in a Pentium CPU (2.8GHz, 1Gb RAM) running in Windows XP operation system.

Numerical Example

Considering the following MILP problem

$$
\begin{aligned}
\min_{x,y} \quad & z = -3x_1 - 2x_2 + 10y_1 + 5y_2 \\
s.t. \quad & x_1 \le 10 + \theta_1 + 2\theta_2 \\
& x_2 \le 10 - \theta_1 + \theta_2 \\
& x_1 + x_2 \le 20 - \theta_2 \\
& x_1 + 2x_2 \le 12 + \theta_1 \\
& x_1 - 20y_1 \le 0 \\
& x_2 - 20y_2 \le 0 \\
& -x_1 + x_2 \ge 4 \\
& x \ge 0, y \in \{0,1\}, 0 \le \theta_{1,2} \le 10
\end{aligned}
$$

To illustrate the proposed parametric solution algorithm, let's consider the case of finding the local parametric solution around the point θ^0=(5,1), then the following steps will be applied:

Step 1, solve MILP with fixed θ^0, get: y^*=(0,1), z*= -7

Step 2, solve mpLP with fixed y^* to cover θ^0

$f^*(\theta)= -15+2\theta_1-2\theta_2$, $x_1^*(\theta)=0$, $x_2^*(\theta)=10-\theta_1+\theta_2$

$$
CR^0 = \begin{cases} \theta_1 - \theta_2 \le 6 \\ -1.5\theta_1 + \theta_2 \le -4 \\ -\theta_1 + 2\theta_2 \le 10 \\ 0 \le \theta_1, \theta_2 \le 10 \end{cases} \quad \text{(Figure 2.3a)}
$$

1 st iteration: step 3, check whether better solution exist in CR^0

results: err*>0, y'=(1,1), θ'=(8,6)

1st iteration: step 4, solve mpLP with fixed y' to cover θ'

$f'(\theta)$= 0.3333-1.6667θ_1 $CR' = \begin{cases} 1.333\theta_1 - \theta_2 \le 4.667 \\ \theta_1 + 1.5\theta_2 \le 20 \\ 0 \le \theta_1, \theta_2 \le 10 \end{cases}$ (Figure 2.3b)

1 st iteration: step 5, update CR^0 by excluding CR^{ex} (Figure 2.3c)

$$CR^0 = \begin{cases} \theta_1 - \theta_2 \le 6 \\ -1.5\theta_1 + \theta_2 \le -4 \\ -\theta_1 + 2\theta_2 \le 10 \\ 1.333\theta_1 - \theta_2 \le 4.667 \\ 0 \le \theta_1, \theta_2 \le 10 \end{cases}$$ (Figure 2.3d)

2nd iteration: step 3, check whether better solution exist in CR^0

Results: err*>0, y'=(1,1), θ'=(8.3,6.3)

2 nd iteration: step 4, solve mpLP with fixed y' to cover θ'

$f'(\theta)$= -23+5θ_1-5θ_2 $CR' = \begin{cases} \theta_1 - \theta_2 \le 6 \\ -1.333\theta_1 + \theta_2 \le -4.667 \\ -\theta_1 + 1.5\theta_2 \le 2 \\ 0 \le \theta_1, \theta_2 \le 10 \end{cases}$

2 nd iteration: step 5, update CR^0 by excluding CR^{ex} (Figure 2.3e)

3 nd iteration: step 3, err*=0, stop and return the parametric solution around θ^0

$$CR^* = \begin{cases} \theta_1 - \theta_2 \le 6 \\ 1.333\theta_1 - \theta_2 \ge 4.667 \\ \theta_1 - \theta_2 \ge 2.667 \\ 0 \le \theta_1, \theta_2 \le 10 \end{cases}$$ (Figure 2.3f)

(y_1^*,y_2^*)=(0,1) , $x_1^*(\theta)$=0, $x_2^*(\theta)$=10-θ_1+θ_2, $f^*(\theta)$= -15+2θ_1-2θ_2

With this critical region, the original parameter space can be partitioned and same local parametric solution algorithm will be applied to every unexplored region. Finally, the critical region map of the parametric solution will be derived as shown in Figure 2.4.

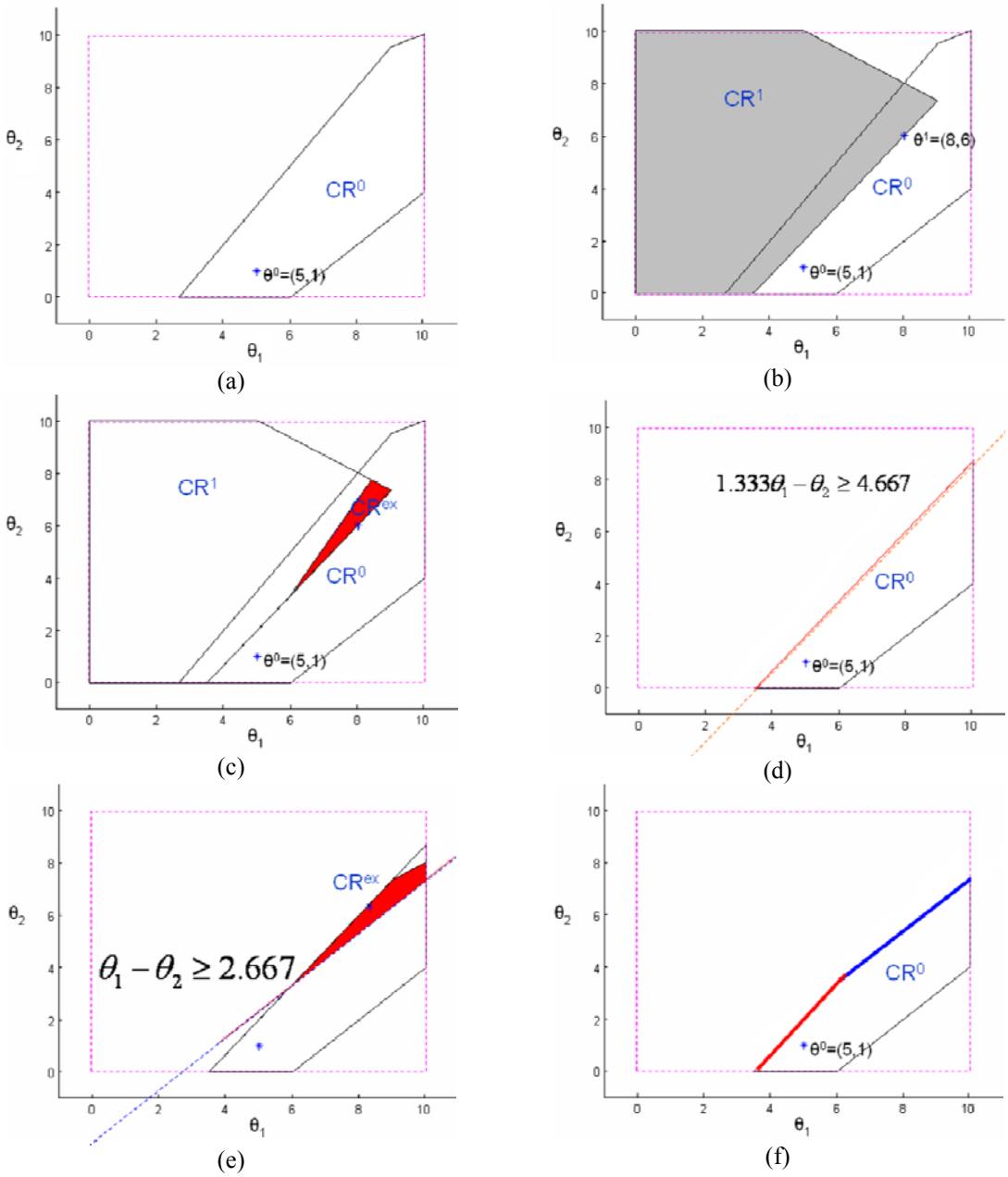


Figure 2.3 Illustration of the solution procedure for the numerical example

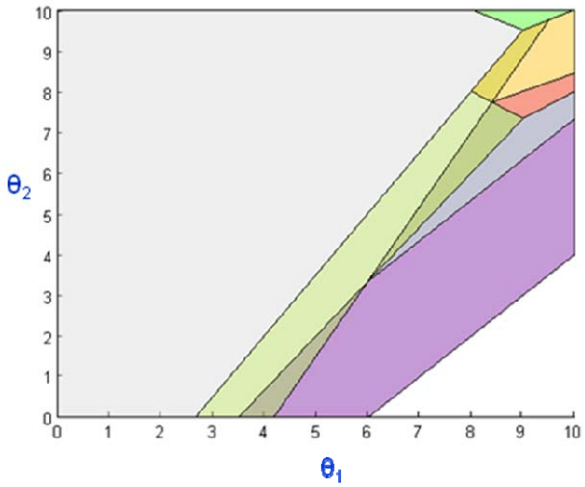


Figure 2.4 Final parametric solution map(partition of the parameter space into critical regions)

2.3 Uncertainty analysis for scheduling problem

Problem Formulation

The mathematical model used for batch process scheduling in this chapter follows the main idea of the continuous time formulation proposed by (Ierapetritou & Floudas, 1998). The general model involves the following objective and constraints:

Problem (2.8):

$$\max \sum_{s,n} price_s d_{s,n} \tag{2.8a}$$

s.t.

$$\sum_{i \in I_j} wv_{i,j,n} \le 1 \qquad \forall j \in J, \forall n \in N \tag{2.8b}$$

$$st_{s,n} = st_{s,n-1} - d_{s,n} - \sum_{i \in I_s} \rho^P_{s,i} \sum_{j \in J_i} b_{i,j,n} + \sum_{i \in I_s} \rho^c_{s,i} \sum_{j \in J_i} b_{i,j,n-1} \qquad \forall s \in S, \forall n \in N \tag{2.8c}$$

$$st_{s,n} \le st_s^{\max} \qquad \forall s \in S, \forall n \in N \tag{2.8d}$$

$$v_{i,j}^{\min} wv_{i,j,n} \le b_{i,j,n} \le v_{i,j}^{\max} wv_{i,j,n} \qquad \forall i \in I, \forall j \in J_i, \forall n \in N \tag{2.8e}$$

$$\sum_n d_{s,n} \ge r_s \qquad \forall s \in S \tag{2.8f}$$

$$Tf_{i,j,n} = Ts_{i,j,n} + \alpha_{i,j} wv_{i,j,n} + \beta_{i,j} b_{i,j,n} \qquad \forall i \in I, \forall j \in J_i, \forall n \in N \tag{2.8g}$$

$$Ts_{i,j,n+1} \ge Tf_{i,j,n} - H(1 - wv_{i,j,n}) \qquad \forall i \in I, \forall j \in J_i, \forall n \in N \tag{2.8h}$$

$$Ts_{i,j,n+1} \geq Tf_{i',j,n} - H(1 - wv_{i,'j,n}) \qquad \forall i, i' \in I_j, \forall j \in J, \forall n \in N \qquad (2.8i)$$

$$Ts_{i,j,n+1} \geq Tf_{i',j',n} - H(1 - wv_{i,'j',n}) \qquad \forall i, i' \in I_j, i \neq i', \forall j, j' \in J, \forall n \in N \qquad (2.8j)$$

$$Ts_{i,j,n+1} \geq Ts_{i,j,n} \qquad \forall i \in I, \forall j \in J_i, \forall n \in N \qquad (2.8k)$$

$$Tf_{i,j,n+1} \geq Tf_{i,j,n} \qquad \forall i \in I, \forall j \in J_i, \forall n \in N \qquad (2.8l)$$

$$Ts_{i,j,n} \leq H \qquad \forall i \in I, \forall j \in J_i, \forall n \in N \qquad (2.8m)$$

$$Tf_{i,j,n} \leq H \qquad \forall i \in I, \forall j \in J_i, \forall n \in N \qquad (2.8n)$$

In the above formulation, the objective function is the profit (different performance measures can be used like makespan); allocation constraints (2.8b) state that only one of the tasks can be performed in each unit at an event point (n); constraints (2.8c) represent the material balances for each state (s) expressing that at each event point (n) the amount $st_{s,n}$ is equal to that at event point (n-1), adjusted by any amounts produced and consumed between event points (n-1) and (n), and delivered to the market at event point (n); the storage and capacity limitations of production units are expressed by constraints (2.8d) and (2.8e); constraints (2.8f) are written to satisfy the demands of final products; and constraints (2.8g) to (2.8n) represent time limitations due to task duration and sequence requirements in the same or different production units. Detailed description of the symbols in the above formulation is provided in the notation section of this chapter. It should be noticed that minimum product demand and minimum processing time in the uncertain range can be used to identify an appropriate event point number before the multiparametric solution process to avoid the loss of solution optimality.

Example

This example process involves three processing stages, namely mixing, reaction, and separation, which are processed in 3 units respectively. The state-task-network (STN) representation of this example is shown in Figure 2.5 and the data is shown in Table 2.2. Products include S3 and S4 (the purified product). For the deterministic formulation with 5 event points, there are 236 constraints, 45 integer variables and 86 continuous variables. Solving the deterministic MILP problem normally requires around 0.25 CPU second.

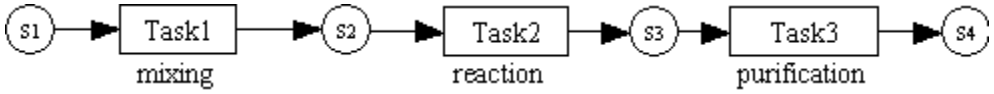


Figure 2.5 State Task Network (STN) of Example 1

(1) Demand Uncertainty

Assuming only demand uncertainty for the two products gives rise to RHS uncertainty in problem (P2) and an mpMILP problem is solved. The demands are defined in Table 2.3, where the variation ranges of the uncertain parameters are given in a boundary form. The corresponding multiparametric programming problem (P2) is solved using the proposed method and the solution is shown in Figure 2.6 and Table 2.4 (note that in all the examples of this chapter the objective is set as minimum negative profit). The total time consumed is 140.6 CPU sec.

As stated in section 2, all the parametric objectives are linear and the critical regions are formed by linear constraints in this case. Among the critical regions (Figure 2.6), CR3 and CR5 have an overlapping area, because they have same objective function and integer solution and are actually belong to the same larger nonconvex critical region. This is also a characteristic of the solution for mpMILP, which is different from mpLP which always has a convex critical region.

Table 2.2 Data for example 1

Unit	Capacity	Suitability	Processing time
Unit 1	100	Task 1	4.5
Unit 2	75	Task 2	3.0
Unit 3	50	Task 3	1.5
state	Storage capacity	Initial amount	Price
State 1	Unlimited	unlimited	0
State 2	100	0.0	0
State 3	100	0.0	0.7
State 4	unlimited	0.0	1.0

Table 2.3 Demand uncertainty for example 1

Parameter	Value	Variation Range
Demand of S3	θ_1	$0 \le \theta_1 \le 50$
Demand of S4	$50+\theta_2$	$-50 \le \theta_2 \le 50$

Table 2.4 Solution of example 1 with demand uncertainty

	Parametric Objective	Critical Region
1	-90.46	CR1
2	$-72.097+0.029\theta_2$	CR2
3	$-96.14+0.158\theta_1$	CR3, CR5
4	-88.55	CR4

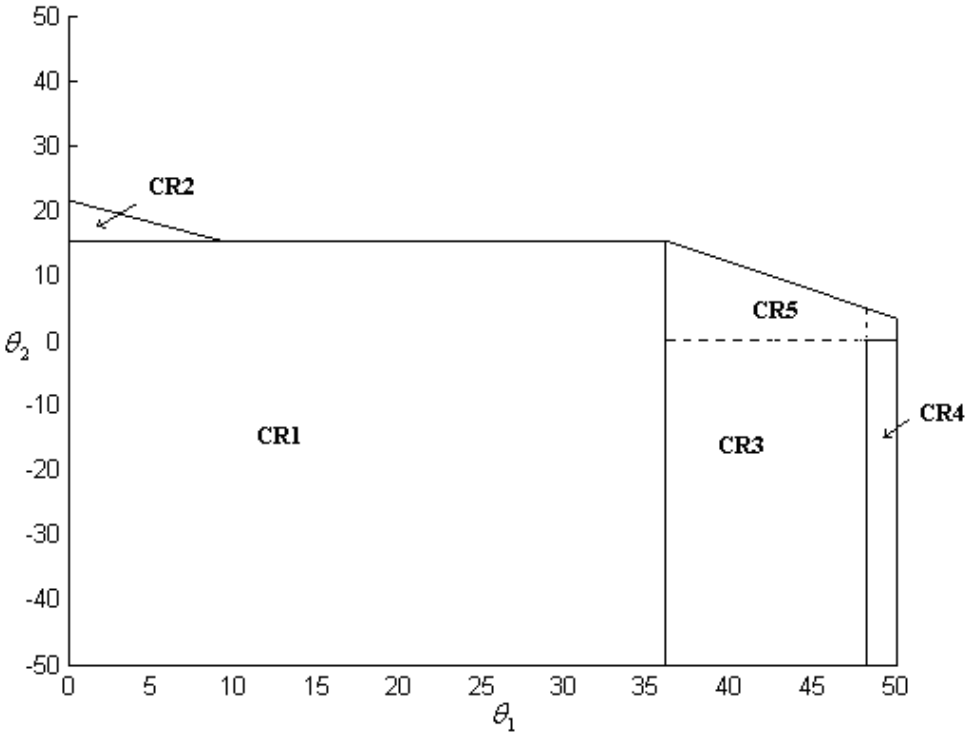


Figure 2.6 Critical region of Example 1 with demand uncertainty

(2) Price Uncertainty

In this case, we consider the price uncertainty as shown in Table 2.5. Thus the uncertainty is included in the objective function and a specific mpMIQP problem is solved. The solution of the multiparametric programming problem is shown in Table 2.6 and Figure 2.7, where the solution still has linear objectives and critical regions formed by linear inequalities. This verified the conclusion in section 2. Total time consumed is 114 CPU sec.

Table 2.5 Price uncertainty for example 1

Parameter	Value	Variation Range
Price of S3	$0.7+\theta_1$	$-0.5 \le \theta_1 \le 0.5$
Price of S4	$1+\theta_2$	$-0.5 \le \theta_2 \le 0.5$

Table 2.6 Solution of example 1 with price uncertainty

	Parametric Objective	Critical Region
1	$-90.46-36.06\theta_1-65.22\theta_2$	CR1
2	$-71.47-71.47\theta_2$	CR2
3	$-88.55-55.07\theta_1-50\theta_2$	CR3, CR4

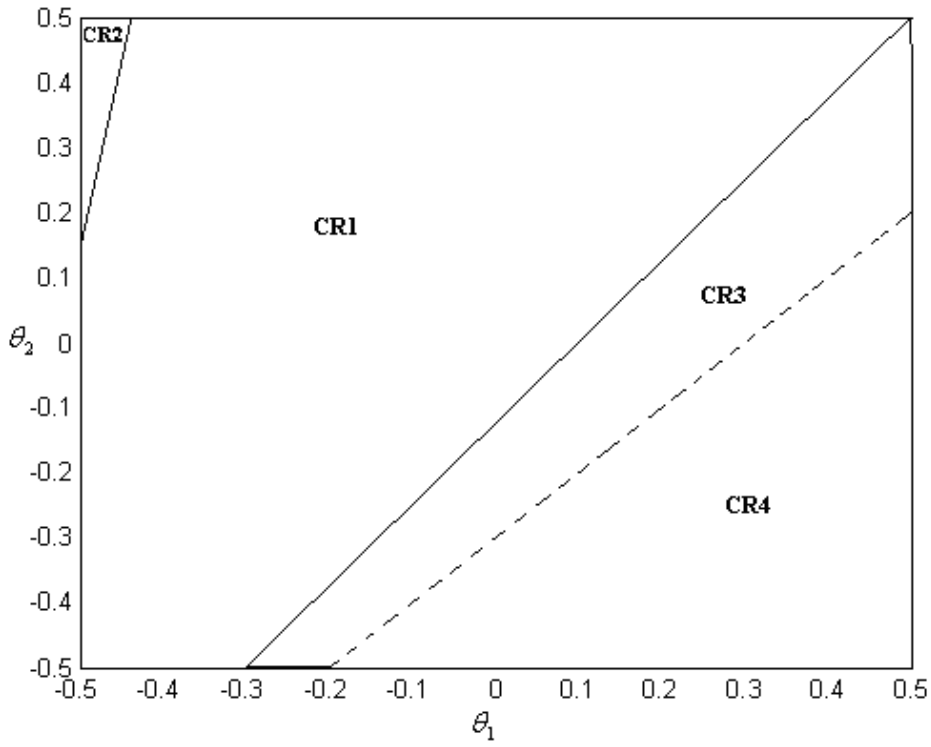


Figure 2.7 Critical region of Example 1 with price uncertainty

(3) Demand and Price uncertainty

In this case both demand and price uncertainties are considered as shown in Table 2.7. Thus the corresponding multiparametric programming problem contains both RHS and objective uncertainties and is also mpMIQP problem. This problem is solved within 250.86 CPU sec. As analyzed in section 2, in this case, the optimal objective functions as shown in Table 2.8 contain quadratic function of uncertain parameters. Note that the critical regions (Figure 2.8) don't involve quadratic constraints here because all of the objective comparison constraints are proved to be redundant in the redundancy test in step 5, so they are not involved in the constraints. Still, we can see that overlapped critical regions exist and they form larger nonconvex regions (e.g., CR1 and CR9, CR2 and CR3). The number of testing points used in this case is 600 and time consumed is 250.9 CPU sec.

Table 2.7 Price and demand Uncertainty for example 1

Parameter	Value	Variation Range
Price of S3	$0.7+\theta_1-0.01\theta_2$	$-0.5 \le \theta_1 \le 0.5$
Demand of S4	$50+\theta_2-20\theta_1$	$-50 \le \theta_2 \le 50$

Table 2.8 Solution of example 1 with demand and price uncertainty

	Parametric Objective	Critical Region
1	$-90.46-36.06\theta_1+0.36\theta_2$	CR1, CR9
2	-71.47	CR2, CR3
3	$-88.55-55.07\theta_1+0.55\theta_2$	CR4
4	$-88.55-49.07\theta_1+0.25\theta_2-20\theta_1^2+1.2\theta_1\theta_2-0.01\theta_2^2$	CR5
5	$-73.55-105.07\theta_1+1.55\theta_2$	CR6
6	$-72.1-32.15\theta_1+0.34\theta_2-29.4\theta_1^2+1.76\theta_1\theta_2-0.015\theta_2^2$	CR7
7	$-87.66-50.13\theta_1+0.35\theta_2-23.32\theta_1^2+1.4\theta_1\theta_2-0.012\theta_2^2$	CR8

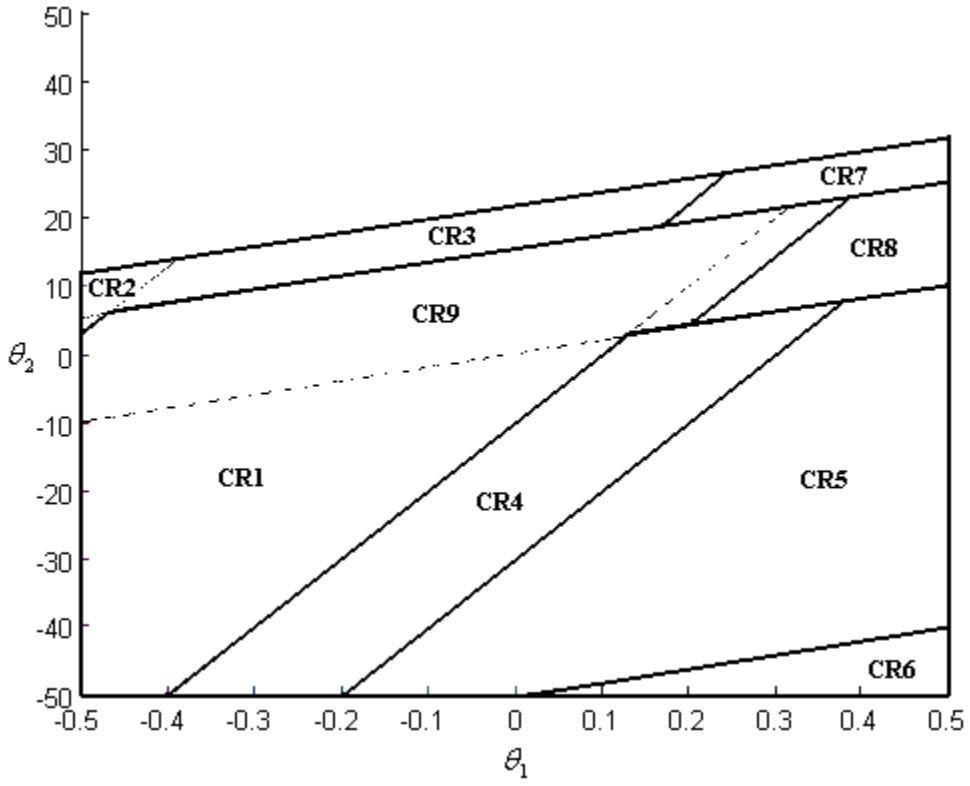


Figure 2.8 Critical region of Example 1 with price and demand uncertainty

(4) Demand, Price and Processing time Uncertainty

In this case, three different uncertain parameters are involved as shown in Table 2.9. To address the processing time uncertainty using the proposed method, we reformulate the duration constraints so that the LHS uncertain parameters only appear as coefficients of integer variables and can be transformed into RHS uncertainty. The original duration constraint (2.8g) is reformulated as follows:

$$Tf_{i,j,n} = Ts_{i,j,n} + (\alpha_{i,j} + \theta)wv_{i,j,n} + \beta_{i,j}b_{i,j,n}$$

In the new formulation, $\alpha_{i,j}$ and $\beta_{i,j}$ is calculated based on nominal processing time, and uncertainty is modeled through the term θ. The uncertain mixing time parameter are formulated using the above new duration constraints so that it is transformed into RHS uncertainty, thus an mpMIQP problem is formulated and solved here. Total 400 testing points is used and the elapsed time is 323.7 CPU sec.

Table 2.9 Demand, price and processing time uncertainty for example 1

Parameter	Value	Variation Range
Price of S4	$1+\theta_1$	$-0.5 \le \theta_1 \le 0.5$
Demand of S4	$50-20\theta_1$	
Mixing time	$4.5+\theta_2$	$-0.5 \le \theta_2 \le 0.5$

Table 2.10 Solution of example 1with demand, price and processing time uncertainty

	Parametric Objective	Critical Region ($-0.5 \le \theta_i \le 0.5, i=1,2$)
1	$-88.55-44\theta_1+25.16\theta_2+20\theta_1^2$	$CR_1=\begin{cases}\theta_1 \le -0.3\\ -2.63\theta_1+6.06\theta_2 \le -1\end{cases}$
2	$-87.66-46.33\theta_1+30.52\theta_2+20\theta_1^2$	$CR_2=\begin{cases}-2.63\theta_1+6.06\theta_2 \le -1\\ -1.533\theta_1+\theta_2 \le 1.17\\ \theta_1 \le -0.184\end{cases}$
3	$-86.27-42.4\theta_1+38.99\theta_2+46.08\theta_1\theta_2$	$CR_3=\begin{cases}-2.63\theta_1+6.06\theta_2 \le -1\\ -0.3 \le \theta_1 \le -0.184\end{cases}$
4	$-90.46-65.22\theta_1+32.92\theta_2+13.04\theta_1\theta_2$	$CR_4=\begin{cases}\theta_1 \ge -0.184\\ -15.22\theta_1+16.46\theta_2+13.04\theta_1\theta_2 \le 6.2\\ -15.22\theta_1+7.75\theta_2+13.04\theta_1\theta_2 \le 1.9\end{cases}$
		$CR_5=\{-0.184 \le \theta_1 \le -0.01$
5	$-88.55-50\theta_1+25.16\theta_2$	$CR_6=\begin{cases}\theta_1 \ge 0\\ 15.22\theta_1-7.75\theta_2-13.04\theta_1\theta_2 \le -1.9\end{cases}$
6	$-72.1-50.58\theta_1+25.16\theta_2+20\theta_1^2$	$CR_7=\begin{cases}-\theta_1+1.223\theta_2 \le 1.074\\ 1.533\theta_1-\theta_2 \le -1.17\end{cases}$

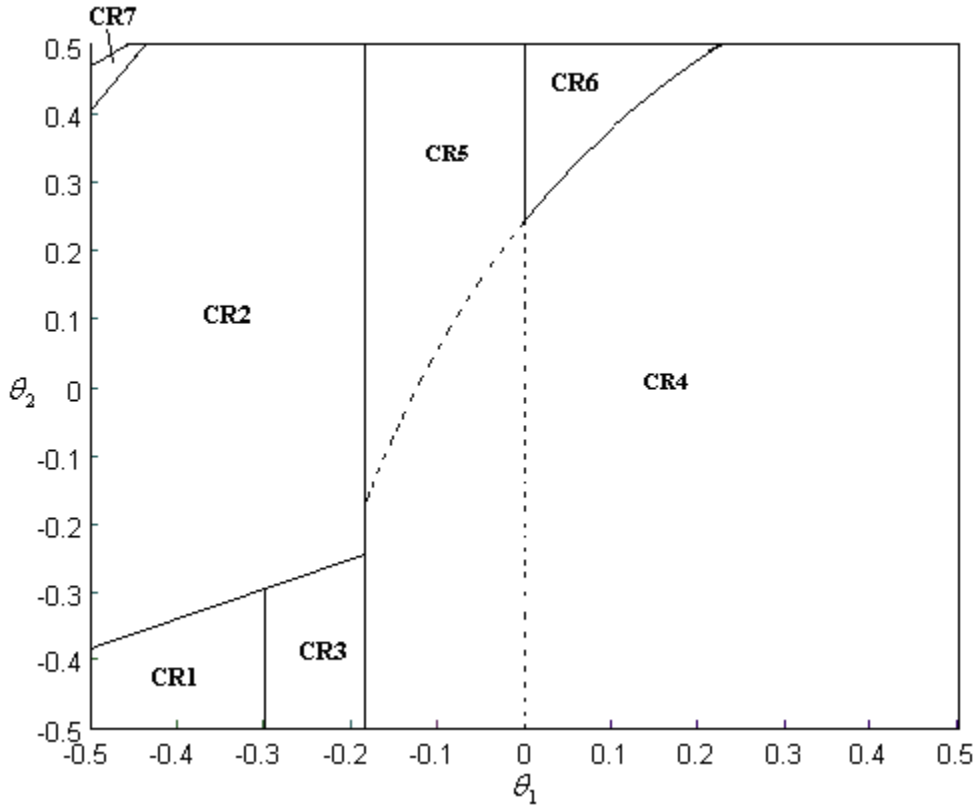


Figure 2.9 Critical region of Example 1 with demand, price and processing time uncertainty

As shown in the solution (Table 2.10, Figure 2.9), the critical region and parametric objective function both involve quadratic terms. This also verifies the conclusion in section 2 and proves the effectiveness of the proposed method in solving mpMILP and mpMIQP problems. In all the different uncertainty cases addressed for example 1, the computation time is no more than 300 CPU seconds, where 40 evenly distributed points are generated in every dimension of the parameter space, which is proved to be enough to cover all the critical regions. During the solution process, once a point is checked and is found to be covered by any critical region, it is fathomed. So although 1600 points are tested, only limited points are solved to find critical region, e.g., only 5,4,9,7 points are solved for the above four cases, respectively. The next example illustrates the computational complexity for relatively larger scale problem.

2.4 Summary

In this chapter, a multiparametric programming framework for MILP problem is proposed, which is further applied to solve the process scheduling problem under uncertainty. This method provides an exact and systematic way to analyze the uncertainty in process scheduling problem and the parametric information achieved can be used in several different ways: the result of multiparametric programming for the scheduling problem can be used to analyze the effect of parameter variation on the scheduling performance; and also the solution provides a basis for reactive scheduling in the sense that the decision maker can rapidly find a new schedule with the realization of uncertainty.

To increase the efficiency of attaining the exact solution map of the corresponding mpMILP and mpMIQP problem, the proposed framework uses a decomposition method which solves for the parametric information around a certain parameter value, and not seeking for a complete map of solution at one time. This method can give the decision maker useful information about uncertainty effects fast. Another advantage of the proposed methodology is that it can be easily parallelizable by decomposing the original parameter space into smaller regions that can be solved in parallel thus decreasing the computational complexity of the algorithm.

The proposed method is also efficient in the critical region updating process because the MILP/MINLP problem in step 3 of algorithm 1 is solved to seek just better but not best solution in a given critical region. In other words, the proposed formulation is used to seek a feasible solution but not a global optimal solution, thus the computation efficiency is increased. The only big computation effort is in the final iteration of the proposed framework which needs to prove that no better solution exists, so a global optimal solution of the MILP/MINLP is needed.

The consideration of a general form of LHS uncertainty as coefficient of continuous variables is still a challenge. Because for this case the exact parametric solution is complex, different efficient methods have to be developed to address the underlying complexity of this problem.

Nomenclature

$i \in I$	tasks
I_s	tasks which produce or consume state (s)
I_j	tasks which can be performed in unit (j)
$j \in J$	units
J_i	units which are suitable for performing task (i)
$n \in N$	event points representing the beginning of a task
$s \in S$	states
S_p	states belong to products
S_r	states belong to raw materials
$price_s$	price of state (s)
STI_s	initial amount of state (s)
STF_s	final amount of state (s)
$d_{s,n}$	amount of state (s) delivered to the market at event point (n)
$wv_{i,j,n}$	binary, whether or not task (i) in unit (j) start at event point (n)
$st_{s,n}$	continuous, amount of state (s) at event point (n)
$\rho^P_{s,i}$, $\rho^C_{s,i}$	proportion of state (s) produced, consumed by task(i), respectively
$b_{i,j,n}$	amount of material undertaking task (i) in unit (j) at event point (n)
$st^{\max}_s$	available maximum storage capacity for state (s)
$v^{\min}_{i,j}$, $v^{\max}_{i,j}$	minimum amount, maximum capacity of unit (j) when processing task (i)
r_s	market demand for state (s) at the end of the time horizon
$Tf_{i,j,n}$	time at which task (i) finishes in unit (j) while it starts at event point (n)
$Ts_{i,j,n}$	time at which task (i) starts in unit (j) at event point (n)
$\alpha_{i,j}$, $\beta_{i,j}$	constant, variable term of processing time of task (i) in unit (j)
H	time horizon

Chapter 3

Robust Preventive Scheduling

Abstract: This chapter addresses the preventive scheduling problem using robust optimization technique. Compared to traditional scenario based stochastic programming method; robust counterpart optimization method has a unique advantage that the scale of the corresponding optimization problem does not increase exponentially with the number of the uncertain parameters. Three robust counterpart optimization formulations are studied and applied to uncertain scheduling problems in this chapter. The results show that the "budget-parameter" based formulation is the most appropriate model for uncertain scheduling problems since it has the following advantages: the model has the same size as the other formulations; it preserves its linearity; it has the ability to control the degree of conservatism for every constraint and guarantees feasibility for the robust optimization problem.

3.1 Introduction

Uncertainty is a very important concern in real plants for process scheduling since many of the parameters associated with scheduling are not known exactly. Parameters like raw material availability, prices, machine reliability, processing or duration time and market requirements vary with respect to time and are often subject to unexpected deviations, which can cause infeasibilities and production disturbances. Thus scheduling under uncertainty has received a lot of attention in the open literature in recent years from chemical engineering and operations research communities.

According to the different treatment of uncertainty, scheduling methods can be divided into two groups: reactive scheduling and preventive scheduling. Reactive scheduling deals with the problem of modifying the original scheduling policy or generating scheduling policy on time when uncertainty occurs. On the other hand, preventive scheduling aims at generating robust scheduling policies before the uncertainty occurs. Almost all techniques that deal with uncertainty try to find solutions flexible to changes of input data. Although this solution might not be optimal the target is to be as close as possible to the optimal one. Robust

scheduling focuses on obtaining preventive schedules that minimize the effects of disruptions on the performance measure, and tries to ensure that the preventive schedules maintain a high level of performance.

For the problem of robust schedule generation, different methods have been proposed in the literature. Generally, the formulations can be classified into two groups: a) scenario based stochastic programming formulation; and b) robust counterpart optimization formulation.

In the literature, most existing work on robust scheduling has followed the scenario-based formulation. (Mulvey et al., 1995) developed the scenario-based robust optimization to handle the trade-off associated with solution and model robustness. A solution to an optimization is considered to be solution robust if it remains close to the optimal for all scenarios, and model robust if it remains feasible for most scenarios. (Kouvelis et al., 2000) made the first attempts to introduce the concept of robustness for scheduling problems. They suggest a robust schedule when processing times are uncertain and compute robust schedule based on maximum absolute deviation between the robust solution and all the possible scenarios, but this requires knowledge of all possible scenarios. Moreover, the optimal solution of each scenario is supposed to be known a priori. (Vin & Ierapetritou, 2001) addressed the problem of quantifying the schedule robustness under demand uncertainty, introduced several metrics to evaluate the robustness of a schedule and proposed a multiperiod programming model using extreme points of the demand range as scenarios to improve the schedule performance of batch plants under demand uncertainty. Using flexibility analysis, they observed that the schedules from the multiperiod programming approach were more robust than the deterministic schedules. (Balasubramanian & Grossmann, 2002) proposed a multiperiod MILP model for scheduling multistage flowshop plants with uncertain processing times. They minimized expected makespan and developed a special branch and bound algorithm with an aggregated probability model. The scenario-based approaches provide a straightforward way to implicitly incorporate uncertainty. However, they inevitably enlarge the size of the problem significantly as the number of scenarios increases exponentially with the number of uncertain parameters. This main drawback limits the application of these approaches to solve practical problems with a large number of uncertain parameters. (Jia & Ierapetritou, 2007) proposed a multi-objective robust optimization model to deal with the problem of uncertainty in scheduling considering the expected performance (makespan), model robustness and solution robustness. Normal Boundary Intersection (NBI) technique is utilized to solve the multi-objective model and successfully produce Pareto

optimal surface that captures the trade-off among different objectives in the face of uncertainty. The schedules obtained by solving this multiobjective optimization problem include robust assignments that can accommodate demand uncertainty.

Although those uncertainty-handling frameworks that follow the scenario-based formulation target the generation of robust solutions with respect to optimality and feasibility, the model still has the common drawback of scenario based formulation: it requires some statistic knowledge of the input data, which in many cases may be difficult to acquire. Moreover, optimization of expectations is a practice of questionable validity in processes involving only a small number of "trials", because the benefits of an optimum expected value can only be visible in the long term of a large number of trials. Finally, for the scenario based robust optimization method, the uncertainty is modeled through the use of a number of scenarios. This type of method provides a direct way to incorporate uncertainty. However, the problem size will increase exponentially with the number of uncertain parameters, which restricts its application in solving problems with a lot of uncertain parameters.

As an alternative to the scenario-based formulation, the robust counterpart optimization has been proposed which avoids the shortcomings of the scenario-based formulation. The underlying framework of robust counterpart scheduling formulation is based on solving robust counterpart optimization problem for the uncertain scheduling problem.

One of the earliest papers on robust counterpart optimization, by (Soyster, 1973), considered simple perturbations in the data and aimed to find a reformulation of the original problem such that the resulting solution would be feasible under all possible perturbations. The pioneering work by (Ben-Tal & Nemirovski, 1999), (El-Ghaoui et al., 1998), and (Bertsimas & Sim, 2003) extended the framework of robust counterpart optimization, and included sophisticated solution techniques with non-trivial uncertainty sets describing the data. The major advantages of robust counterpart optimization compared to scenario-based stochastic programming are that no assumptions are needed regarding the underlying probability distribution of the uncertain data and that it provides a way of incorporating different attitudes toward risk.

For the problem of process scheduling under uncertainty, only very few works have been done in robust counterpart optimization for generating robust schedules. (Lin et al., 2004) proposed a robust optimization method to address the problem of scheduling with uncertain processing times, market demands, or prices.

The robust optimization model was derived from its deterministic model considering the worst-case values of the uncertain parameters, when uncertainty is in a bounded form. They also studied the case that uncertainty is described by known probability distribution, where the robust optimization formulation introduces a small number of auxiliary variables and additional constraints into the original MILP problem, generating a deterministic robust counterpart problem which provides the optimal/feasible solution given the (relative) magnitude of the uncertain data, a feasibility tolerance, and a reliability level (Janak et al., 2007).

In this chapter, we compare several robust counterpart optimization formulations and proposed the appropriate formulation for scheduling under uncertainty. The rest of the chapter is organized as follows. Section 3.2 introduces the robust counterpart optimization formulations and their extensions to mixed integer linear programming problems. In section 3.3, the problem of scheduling under uncertainty is studied based on robust optimization considering specific uncertainties. Several case studies are presented in section 3.4 to illustrate the application of different robust formulations and present some comparison results, whereas section 3.5 summarizes the main conclusions of the chapter.

3.2 Robust optimization

Compared to scenario-based stochastic programming, robust counterpart optimization represents a more systematic approach for optimization under uncertainty in order to determine flexible solutions. The aim of robust counterpart optimization is to choose a solution which is able to cope best with the various realizations of the uncertain data. The uncertain data is assumed to be unknown but bounded, and most current research assumes convexity of the uncertainty space.

The optimization problem with uncertain parameters is reformulated into a robust counterpart optimization problem. Unlike stochastic programming, robust optimization does not require information about the probability distribution of the uncertain data, and does not optimize an expected value objective function. Robust optimization promises to essentially ensure robustness and flexibility by enforcing feasibility of an optimization problem for the entire given uncertainty space.

In this section, three robust counterpart optimization formulations are presented assuming the following general mixed integer linear programming problem:

$$\max cx \tag{3.P1}$$

$$\text{s.t.} \quad \sum_m a_{lm} x_m \le p_l \quad \forall l$$

$$x_m \quad \text{binary or continuous, m} = 1,2...,n$$

For the rest of the chapter, we assume without loss of generality that data uncertainty affects only the elements of the left hand side matrix coefficients due to the following reasons:

1) The objective function can be transformed into constraint;

2) If the right hand side constant p_l is subject to uncertainty, we can introduce a new variable x_{n+1} which is a binary variable with fixed value 1, and the original constraint is transformed into the following:

$$\sum_m a_{lm} x_m - p_l x_{n+1} \le 0 \tag{3.1}$$

$$1 \le x_{n+1} \le 1 \tag{3.2}$$

Assuming that a_{lm} are uncertain parameters, the constraints in (3.P1) are expanded as follows:

$$\sum_{m \notin M_l} a_{lm} x_m + \sum_{m \in M_l} \tilde{a}_{lm} x_m \le p_l \tag{3.3}$$

where, M_l denotes the index set for the uncertain coefficients in l-th constraint; $\tilde{a}_{lm}$ represent true values for the coefficient parameters that take value within the uncertain range $[a_{lm} - \hat{a}_{lm}, a_{lm} + \hat{a}_{lm}]$, where a_{lm} represent nominal values and $\hat{a}_{lm}$ represent the variation amplitude.

3.2.1 Soyster's formulation

Robust counterpart optimization can be traced back to the work of (Soyster, 1973), which is the first work that considered coefficient uncertainty in linear programming formulations, and showed that such uncertainty can be handled by an equivalent linear programming model. The approach, however, admits the highest protection and is the most conservative one since it ensures feasibility against all potential realizations. Thus it corresponds to the worst-case version of the scenario approach.

The worst case formulation of equation (3.3) then take the following form considering all uncertain parameters to take their boundary values which aims to ensure that for every possible value of the uncertain coefficient, the solution remain feasible:

$$\sum_{m \notin M_l} a_{lm} x_m + \sum_{m \in M_l} a_{lm} x_m + \sum_{m \in M_l} \hat{a}_{lm} \mid x_m \mid \leq p_l \tag{3.4}$$

To eliminate the absolute value, auxiliary variable u_m is introduced for each x_m, $m \in M_l$, which defines new bounds for x_m. Thus the following robust formulation (3.P2) is obtained.

$$\max cx \tag{3.P2}$$

$$\text{s.t.} \quad \sum_{m \notin M_l} a_{lm} x_m + \sum_{m \in M_l} a_{lm} x_m + \sum_{m \in M_l} \hat{a}_{lm} u_m \leq p_l$$

$$\begin{cases} u_m = x_m, & \text{if } x_m \text{ is positive or binary variable} \\ -u_m \leq x_m \leq u_m, & \text{otherwise} \end{cases}$$

In formulation (3.P2), if the uncertain parameter is coefficient of positive or binary variable, then no auxiliary variable and constraint is added because the absolute value is eliminated naturally. The robust formulation (ε, σ)-Interval Robust Counterpart (IRC$[\varepsilon, \sigma]$) proposed by (Lin et al., 2004) belongs to this type of formulation. The difference is that they add certain infeasibility tolerance to the constraints to increase the level of control towards conservative solutions. The motivation to use Soyster's formulation is to provide maximum protection against uncertainty. This type of formulation allows for mitigation of the worse-case scenario, however since all possible realizations of the data are considered, the solution can end up being overly pessimistic and the problem is more likely to be infeasible. For example, a worst case parameter combination where all the processing times and all the demands take the maximum value might lead to an infeasible schedule in a fixed time horizon due to inability of the plant to satisfy the demand in a fixed time horizon.

3.2.2 Ben-Tal and Nemirovski's formulation

Since Soyster's formulation is extremely conservative, it is highly desirable to provide a mechanism to allow tradeoff between robustness and performance. A significant step forward for developing a theory for robust

optimization was taken by (Ben-Tal & Nemirovski, 1999), who proposed the following robust counterpart formulation:

$$\max cx \tag{3.P3}$$

$$\text{s.t.} \quad \sum_m a_{lm} x_m + \sum_{m \in M_l} \hat{a}_{lm} u_{lm} + \Omega_l \sqrt{\sum_{m \in M_l} \hat{a}_{lm}^2 z_{lm}^2} \le p_l$$

$$-u_{lm} \le x_m - z_{lm} \le u_{lm}$$

Let's consider that the values of the uncertain coefficients are obtained through random perturbations:

$$\tilde{a}_{lm} = a_{lm} + \xi_{lm} \hat{a}_{lm} \tag{3.5}$$

where $\{\xi_{lm}\}$ ($m \in M_l$) are independent random variables symmetrically distributed in interval [-1, 1]. As shown by (Ben-Tal & Nemirovski, 1999), this robust formulation ensures that the probability that the l-th constraint is violated is at most $e^{-\Omega_l^2/2}$, i.e.,

$$\Pr\left\{ \sum_{m \notin M_l} a_{lm} x_m + \sum_{m \in M_l} \tilde{a}_{lm} x_m \ge p_l \right\} \le \kappa_l \tag{3.6}$$

$$\kappa_l = e^{-\Omega_l^2/2} \tag{3.7}$$

This robust optimization formulation was first introduced for linear programming problems with uncertain linear coefficients and is extended by (Lin et al., 2004) to MILP problems under uncertainty. The robust formulation $(\varepsilon, \sigma, \kappa)$ - Robust Counterpart (RC$[\varepsilon, \sigma, \kappa]$) proposed by the authors belongs to this type of formulation. This type of robust counterpart formulation has the flexibility of controlling the degree of solution conservatism through the constraint violation probability $e^{-\Omega_l^2/2}$. Its main drawback is that it corresponds to a nonlinear optimization formulation.

3.2.3 Bertsimas and Sim's formulation

Although Ben-Tal and Nemirovski's robust formulation provides a way to consider the tradeoff between performance and robustness, it results in a nonlinear formulation. To avoid the complication of a nonlinear optimization, (Bertsimas & Sim, 2003) considers robust linear programming with coefficient uncertainty using an uncertainty set with budgets. In this robust counterpart optimization formulation, a budget parameter Γ_l (which takes value between 0 and the number of uncertain coefficient parameters in the constraints and is

not necessarily integer) is introduced to control the degree of conservatism of the solution. In other words, it is unlikely that all of the uncertain coefficient parameters will get the worst-case value at the same time, so the goal of this formulation is to control that up to $\lfloor \Gamma_l \rfloor$ of those parameters are allowed to get their worst case value.

$$\sum_m a_{lm} x_m + \max_{\{S_l \cup \{t_l\} | S_l \subseteq M_l, |S_l| = \lfloor \Gamma_l \rfloor, t_l \in M_l \setminus S_l\}} \left\{ \sum_{m \in S_l} \hat{a}_{lm} \mid x_m \mid + (\Gamma_l - \lfloor \Gamma_l \rfloor) \hat{a}_{lt_l} \mid x_{t_l} \mid \right\} \le p_l \tag{3.8}$$

Where S_l represents the subset contains $\lfloor \Gamma_l \rfloor$ uncertain parameters in the constraint, t_l is an index to describe an additional uncertain parameter if Γ_l is not an integer. Thus constraint (3.8) expresses the requirement that up to $\lfloor \Gamma_l \rfloor$ uncertain parameters can get their worst-case values simultaneously, which can be clearly seen when Γ_l is chosen as an integer ($\Gamma_l = \lfloor \Gamma_l \rfloor$), then constraint (3.8) becomes:

$$\sum_m a_{lm} x_m + \max_{\{S_l | S_l \subseteq M_l, |S_l| = \Gamma_l\}} \left\{ \sum_{m \in S_l} \hat{a}_{lm} \mid x_m \mid \right\} \le p_l \tag{3.9}$$

When Γ_l is not integer, one more uncertain parameter a_{lt} can change by $(\Gamma_l - \lfloor \Gamma_l \rfloor)\hat{a}_{lt_l}$. Thus the robust formulation takes the following form:

$$\max cx \tag{3.P4}$$

$$\text{s.t.} \quad \sum_m a_{lm} x_m + \max_{\{S_l \cup \{t_l\} | S_l \subseteq M_l, |S_l| = \lfloor \Gamma_l \rfloor, t_l \in M_l \setminus S_l\}} \left\{ \sum_{m \in S_l} \hat{a}_{lm} u_m + (\Gamma_l - \lfloor \Gamma_l \rfloor) \hat{a}_{lt_l} u_{t_l} \right\} \le p_l$$

$$\begin{cases} u_m = x_m, & \text{if } x_m \text{ is positive or binary variable} \\ -u_m \le x_m \le u_m, & \text{otherwise} \end{cases}$$

To transform problem (3.P4) into a single optimization problem, let

$$\beta(x, \Gamma_l) = \max_{\{S_l \cup \{t_l\} | S_l \subseteq M_l, |S_l| = \lfloor \Gamma_l \rfloor, t_l \in M_l \setminus S_l\}} \left\{ \sum_{m \in S_l} \hat{a}_{lm} u_m + (\Gamma_l - \lfloor \Gamma_l \rfloor) \hat{a}_{lt_l} u_{t_l} \right\} \tag{3.10}$$

Then, $\beta(x, \Gamma_l)$ equals to the objective of optimization problem (3.P5) because the optimal solution z_{lm}^* of (3.P5) must consist of $\lfloor \Gamma_l \rfloor$ variables at 1 and one variable at $\Gamma_l - \lfloor \Gamma_l \rfloor$ with $u_m \ge 0$.

$$\max \sum_{m \in M_l} \hat{a}_{lm} u_m z_{lm} \tag{3.P5}$$

$$\text{s.t.} \quad \sum_m z_{lm} \le \Gamma_l$$

$$0 \le z_{lm} \le 1$$

The dual problem of problem (3.P5) is as follows:

$$\min \quad \Gamma_l z_l + \sum_{m \in M_l} q_{lm} \tag{3.P6}$$

$$\text{s.t.} \quad z_l + q_{lm} \ge \hat{a}_{lm} u_m$$

$$q_{lm} \ge 0 , \; z_l \ge 0$$

where q_{lm} corresponds to the dual variable of the equation $z_{lm} \le 1$, z_l corresponds to the dual variable of the equation $\sum_m z_{lm} \le \Gamma_l$.

The robust formulation is transformed into the following equivalent formulation after substituting the inner optimization problem in (3.P4) with the equivalent optimization problem (3.P6).

$$\max \quad cx \tag{3.P7}$$

$$\text{s.t.} \quad \sum_m a_{lm} x_m + \Gamma_l z_l + \sum_{m \in M_l} q_{lm} \le p_l$$

$$z_l + q_{lm} \ge \hat{a}_{lm} u_m$$

$$q_{lm} \ge 0 , z_l \ge 0$$

$$\begin{cases} u_m = x_m, & \text{if } x_m \text{ is positive or binary variable} \\ -u_m \le x_m \le u_m, & \text{otherwise} \end{cases}$$

In this model, a budget parameter for each constraint in (3.P1) Γ_l limits the number of coefficients that can simultaneously take their worst-case value; the resulting robust optimization remains a linear formulation. This is different from the worst case formulation where all the parameters are considered to get their worst case values at the same time without control of conservatism of the solution. Also note that formulation (3.P7) maintains its linearity. For this robust counterpart formulation, (Bertsimas & Sim, 2003) calculate probability bounds of constraint violation. Specifically, if the uncertain coefficient parameter $\tilde{a}_{ij}$ follows symmetric distribution and takes values in $[a_{ij} - \hat{a}_{ij}, a_{ij} + \hat{a}_{ij}]$, then the probability that the ith constraint is violated satisfies the following constraint:

$$P\left(\sum_m \tilde{a}_{lm} x_m > p_l \right) \le \frac{1}{2^n} \left\{ (1-\mu) \sum_{l=\lfloor v \rfloor}^{n} \binom{n}{l} + \mu \sum_{l=\lfloor v \rfloor+1}^{n} \binom{n}{l} \right\} \le (1-\mu) C(n, \lfloor v \rfloor) + \sum_{k=\lfloor v \rfloor+1}^{n} C(n,k) \tag{3.11}$$

where $n = |M_l|$, $v = \frac{\Gamma_l + n}{2}$, $\mu = v - \lfloor v \rfloor$

$$C(n,k) = \begin{cases} \frac{1}{2^n}, & \text{if } k = 0 \text{ or } k = n \\ \frac{1}{\sqrt{2\pi}} \sqrt{\frac{n}{(n-k)k}} \exp(n \log(\frac{n}{2(n-k)}) + k \log(\frac{n-k}{k})), & \text{otherwise} \end{cases}$$

When applying this robust counterpart formulation to practical problems with large number of uncertain parameters, we can use the feasibility test to identify a-priori which parameters are allowed to take the worst case value, thus providing a guide of assigning appropriate budget parameter to the different constraints.

3.2.4 Comparison of different formulations

For a deterministic mixed integer linear programming (MILP) problem with n variables, m constraints (not counting the bounding constraints) having a total of k uncertain parameters, where j of all the constraints are subject uncertainty and q of all the decision variables are subject to uncertain coefficient, we have the following comparison for the three different robust formulations presented in the previous section in terms of the number of variables, the number of constraints required, the type of formulation and the information obtained:

i) Formulation 1 by Soyster has $n+q$ variables, $m+2q$ constraints and is a linear formulation, but it provides no control over the degree of conservatism of the solution;

ii) Formulation 2 by Ben-Tal and Nemirovski is a second order cone problem (nonlinear). It has $n+2k$ variables, $m+2k$ constraints; and it is able to control the degree of conservatism through the constraint violation probability parameter;

iii) Formulation 3 by Bertsimas and Sim is a linear optimization problem. It has $n+j+k+q$ variables and $m+k+2q$ constraints. It also provides a way to control the degree of solution conservatism through the budget parameter Γ_l.

In summary, Soyster's worst-case formation is the simplest formulation with the smallest number of variables and constraints but it is not able to adjust the solution conservatism, thus the generated solution will often be too pessimistic. Ben-Tal and Nemirovski's formulation provides a level of control for solution conservatism, but it results in a nonlinear formulation, which will cause computational complexity in solving

mixed integer nonlinear programming (MINLP) problems. On the other hand, the robust counterpart optimization formulation proposed by Bertsimas and Sim is a linear formulation which has the flexibility of adjusting the solution conservatism and also does not result in substantial increase in problem size. Note that all the robust counterpart formulations have the same number of binary variables as the original deterministic formulation. In the next section, different uncertainties (price, processing time and demand) in process scheduling problem will be considered using Bertsimas and Sim's robust counterpart formulation whereas comparison of the three robust formulations will be conducted through different examples in section 4.

3.3 Robust scheduling

For the general process scheduling problem, the following deterministic formulation (3.P8) proposed by (Ierapetritou & Floudas, 1998) is used:

Problem (3.P8)

$$\max \sum_{s\in S_p,n} price_s d_{s,n} - \sum_{s\in S_r} price_s (STI_s - STF_s)$$

s.t.

$$\sum_{i\in I_j} wv_{i,j,n} \le 1 \qquad \forall j \in J, \forall n \in N \tag{3.12}$$

$$st_{s,n} = st_{s,n-1} - d_{s,n} - \sum_{i\in I_s} \rho^P_{s,i} \sum_{j\in J_i} b_{i,j,n} + \sum_{i\in I_s} \rho^c_{s,i} \sum_{j\in J_i} b_{i,j,n-1} \qquad \forall s \in S, \forall n \in N \tag{3.13}$$

$$st_{s,n} \le st_s^{\max} \qquad \forall s \in S, \forall n \in N \tag{3.14}$$

$$v_{i,j}^{\min} wv_{i,j,n} \le b_{i,j,n} \le v_{i,j}^{\max} wv_{i,j,n} \qquad \forall i \in I, \forall j \in J_i, \forall n \in N \tag{3.15}$$

$$\sum_n d_{s,n} \ge r_s \qquad \forall s \in S \tag{3.16}$$

$$Tf_{i,j,n} = Ts_{i,j,n} + \alpha_{i,j} wv_{i,j,n} + \beta_{i,j} b_{i,j,n} \qquad \forall i \in I, \forall j \in J_i, \forall n \in N \tag{3.17}$$

$$Ts_{i,j,n+1} \ge Tf_{i,j,n} - H(1 - wv_{i,j,n}) \qquad \forall i \in I, \forall j \in J_i, \forall n \in N \tag{3.18}$$

$$Ts_{i,j,n+1} \ge Tf_{i',j,n} - H(1 - wv_{i,'j,n}) \qquad \forall i, i' \in I_j, \forall j \in J, \forall n \in N \tag{3.19}$$

$$Ts_{i,j,n+1} \ge Tf_{i',j',n} - H(1 - wv_{i,'j',n}) \qquad \forall i, i' \in I_j, i \ne i', \forall j, j' \in J, \forall n \in N \tag{3.20}$$

$$Ts_{i,j,n+1} \ge Ts_{i,j,n} \qquad \forall i \in I, \forall j \in J_i, \forall n \in N \tag{3.21}$$

$$Tf_{i,j,n+1} \ge Tf_{i,j,n} \qquad \forall i \in I, \forall j \in J_i, \forall n \in N \tag{3.22}$$

$$Ts_{i,j,n} \le H \qquad \forall i \in I, \forall j \in J_i, \forall n \in N \qquad (3.23)$$

$$Tf_{i,j,n} \le H \qquad \forall i \in I, \forall j \in J_i, \forall n \in N \qquad (3.24)$$

In the above formulation, the objective function is the profit (different performance measures can be used like makespan), the constraints is similar as the one presented in problem (2.8). Although the above scheduling formulation has been used, the proposed methodology in this chapter is not limited to this model and it can also be applied into other scheduling formulations using either discrete or continuous time models (Floudas & Lin, 2004),(Méndez et al., 2006). In the following, different type of uncertainties including price uncertainty, processing time uncertainty and demand uncertainty are studied. Given the advantage of Bertsimas and Sim's robust formulation, this approach is adopted in this chapter although comparisons with the other models are also provided in section 4.

3.3.1 Price uncertainty

To apply Bertsimas and Sim's formulation for the price uncertainty in the objective function, the original objective is transformed into following:

$$\max \ profit$$

$$\text{s.t.} \quad profit - \sum_{s \in S_p, n} \tilde{price}_s d_{s,n} + \sum_{s \in S_r} \tilde{price}_s (STI_s - STF_s) \le 0 \qquad (3.25)$$

Thus, the uncertain price parameters $\tilde{price}_s$ ($\tilde{price}_s \in [price_s - \hat{price}_s, price_s + \hat{price}_s]$) appear in the left hand side of the new constraint. The number of uncertain parameter in this constraint equals to the number of uncertain prices. If the number of uncertain prices is k, then the budget parameter Γ^p takes value in [0, k]. To apply the robust formulation, this constraint is transformed into the following set of constraints (3.26)~(3.29):

$$profit - \sum_{s \in S_p, n} \tilde{price}_s d_{s,n} + \sum_{s \in S_r} \tilde{price}_s (STI_s - STF_s) + \Gamma^p z^p + \sum_s q_s^p \le 0 \qquad (3.26)$$

$$z^p + q_s^p \ge \hat{price}_s \sum_n d_{s,n}, \quad s \in S_p \qquad (3.27)$$

$$z^p + q_s^p \ge \hat{price}_s (STI_s - STF_s), \quad s \in S_r \qquad (3.28)$$

$$0 \le \Gamma^p \le k, z^p \ge 0, q_s^p \ge 0 \qquad (3.29)$$

Here, only one constraint is subject to uncertainty and the number of uncertain parameters is the number of uncertain prices. The constraints introduced correspond to the constraints in (3.P7). Since $d_{s,n} \geq 0$ and $STI_s - STF \geq 0$, no auxiliary variable is incorporated.

3.3.2 Processing time uncertainty

Let's consider that the processing times take value in the symmetric region $\tilde{T}_{i,j} \in [T_{i,j} - \hat{T}_{i,j}, T_{i,j} + \hat{T}_{i,j}]$. The original duration constraint is reformulated as following:

$$Ts_{i,j,n} - Tf_{i,j,n} + \alpha_{i,j} wv_{i,j,n} + \beta_{i,j} b_{i,j,n} + \tilde{\theta}_{i,j} wv_{i,j,n} \leq 0 \tag{3.30}$$

In equation (3.30), $\alpha_{i,j}$ and $\beta_{i,j}$ are calculated based on nominal processing time: $\alpha_{i,j} = 2T_{i,j}/3$, $\beta_{i,j} = 2T_{i,j}/3(v_{i,j}^{\max} - v_{i,j}^{\min})$. $Tf_{i,j,n}$ represents the lower bound on the finishing time of the task, instead of the exact finishing time as determined by the original duration constraint in (Ierapetritou & Floudas, 1998). In the reformulated constraint (3.30), uncertainty is modeled through the last term $\tilde{\theta}_{i,j} wv_{i,j,n}$. Here, $\tilde{\theta}_{i,j}$ represent the variable part of the processing time parameter, so it takes value in the range $[-\hat{T}_{i,j}, \hat{T}_{i,j}]$ and its nominal value is 0. Since there is only one uncertain coefficient in the reformulated duration constraint, the budget parameter Γ^t for the corresponding robust formulation takes value in [0, 1].

$$Ts_{i,j,n} - Tf_{i,j,n} + \alpha_{i,j} wv_{i,j,n} + \beta_{i,j} b_{i,j,n} + 0 \cdot wv_{i,j,n} + \Gamma^t z_{i,j,n}^t + q_{i,j}^t \leq 0 \tag{3.31}$$

$$z_{i,j,n}^t + q_{i,j}^t \geq \hat{T}_{i,j} wv_{i,j,n} \tag{3.32}$$

$$0 \leq \Gamma^t \leq 1, z_{i,j,n}^t \geq 0, q_{i,j}^t \geq 0 \tag{3.33}$$

Constraints (3.31)-(3.33) correspond to the constraints in (3.P7) respectively. For every duration constraint, one additional variable $z_{i,j,n}^t$ and one constraint are added; for every uncertain parameter $\tilde{\theta}_{i,j}$, one additional variable is added. So totally the additional number of variables is the sum of the number of uncertain duration constraints and number of uncertain processing time parameters.

3.3.3 Demand uncertainty

The demand uncertainty appears on the right hand side of the demand constraints,

$$\sum_n d_{s,n} \geq \tilde{r}_s \tag{3.34}$$

Assuming that the demand parameter takes value in the symmetric region: $\tilde{r}_s \in [r_s - \hat{r}_s, r_s + \hat{r}_s]$. To apply the robust formulation, an additional binary variable with fixed value 1 is added and the constraint is transformed as follows:

$$-\sum_n d_{s,n} + \tilde{r}_s \cdot 1 \leq 0 \tag{3.35}$$

Thus, the demand constraint has only one uncertain coefficient and thus the budget parameter Γ^d takes value in [0, 1]. The following constraints (3.36)-(3.38) are then incorporated in the robust formulation:

$$-\sum_n d_{s,n} + r_s + \Gamma^d z^d + q_s^d \leq 0 \tag{3.36}$$

$$z^d + q_s^d \geq \hat{r}_s \tag{3.37}$$

$$0 \leq \Gamma^d \leq 1, z^d \geq 0, q_s^d \geq 0 \tag{3.38}$$

3.4 Examples

In all the examples presented in this section, the resulted MINLP problem is solved in GAMS using DICOPT solver and the MILP problems are solved in GAMS using CPLEX 10.1 solver in a Pentium PC (3.8GHz, 1G RAM) running in Windows XP operating system.

3.4.1 Example 1

This example is taken from (Ierapetritou & Floudas, 1998) and involves the production of two products using three raw materials (Figure 3.1). Detail process data for this example are shown in Table 3.1. Through this example, we are comparing the three different robust counterpart optimization formulations stated in previous section considering different types of uncertainties and finally the problem is solved using the Bertsimas and Sim's robust formulation with a systematic consideration of all the uncertainties.

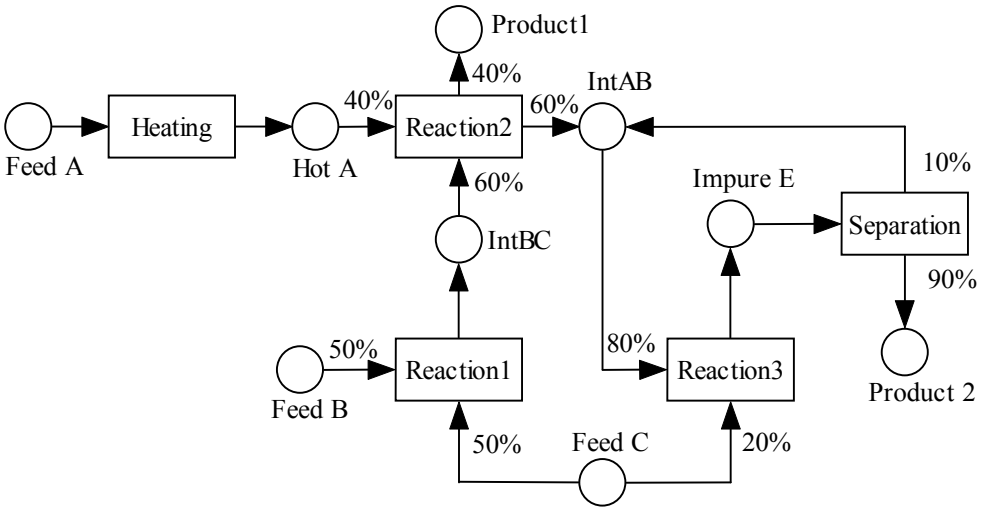


Figure 3.1 State Task Network (STN)representation of Example 1

Table 3.1 Process data for the example 1

Unit	Capacity	Suitability	Processing time
Heater	100	Heating	1.0
Reactor 1	50	Reaction 1,2,3	2.0, 2.0, 1.0
Reactor 2	80	Reaction 1,2,3	2.0, 2.0, 1.0
Sill	200	Separation	1 for product 2, 2 for IntAB
State	Storage capacity	Initial amount	
Feed A	Unlimited	Unlimited	
Feed B	Unlimited	Unlimited	
Feed C	Unlimited	Unlimited	
Hot A	100	0.0	
IntAB	200	0.0	
IntBC	150	0.0	
impure	200	0.0	
Product 1(P1)	Unlimited	0.0	
Product 2(P2)	Unlimited	0.0	

(a) Price uncertainty

In this case, bounded and symmetric uncertainty of raw material cost and product price is assumed. The scheduling horizon is 8 hours and 8 event points are used in the continuous scheduling formulation presented in section 3. The nominal cost of all raw materials is 5 and the nominal prices of product 1 and product 2 are 10, 15, respectively. A 5% variability level is assumed for all the prices and no infeasibility tolerance is considered in the robust formulation.

First, the deterministic schedule using all nominal price values is solved, and the optimal schedule has an optimal profit of 1088.75. Because the objective is profit maximization, the worst case scenario

corresponds to the minimum value of all product prices (9.5, 14.25) and maximum value of raw material costs (5.25, 5.25, 5.25). The resulted profit from Soyster's worst case formulation is 959.56. The extended Ben-Tal's robust formulation for MILP problem proposed by (Lin et al., 2004) is solved with reliability level 28.4% and 10%, which means that the probability that the constraint is violated is at most 28.4% and 10%. Finally, the robust formulation by Bertsimas and Sim is solved using different budget parameters between 0 and 5 (the number of uncertain prices). A detailed comparison of the different robust formulations is shown in Table 3.2 (the optimality gap in CPLEX solver is set as 0.1).

Table 3.2 Comparison of the robust formulations for price uncertainty

	Nominal	Soyster	Ben-Tal		Bertsimas and Sim			
					Γ^p=0	Γ^p=2.5	Γ^p=4.19	Γ^p=5
Objective	1088.75	959.56	981.09	961.73	1088.75	989.63	967.44	959.56
Upper probability of constraint violation	-	-	0.284	0.10	0.695	0.284	0.10	0.031
CPU time (s)	14.4	22.9	64.6	44.4	13.8	33.4	20.2	19.3
Continuous variables	401	401	419		407			
Binary variable	64	64	64		64			
Constraints	884	884	899		889			

As shown in Table 3.2, the result of the deterministic schedule using nominal parameter values is the same as the result of Bertsimas and Sim's formulation when budget parameter is set as 0; the result of the Soyster's worst case formulation is the same as that of Bertsimas and Sim's when budget parameter takes its maximum value. Soyster's formulation has the same problem size as the nominal deterministic formulation because the uncertain parameters are coefficients of positive variables and no auxiliary variable is added. Bertsimas and Sim's robust formulation is relatively more efficient compared to Ben-Tal's formulation because a MILP instead of MINLP problem is solved. Moreover, when the maximum probability of constraint violation is the same, Bertsimas and Sim's robust formulation generates higher profit than Ben-Tal's formulation, which means that Ben-Tal's formulation is more conservative than Bertsimas and Sim's formulation.

(b) Processing time uncertainty

In this case study, the scheduling horizon is 12 hours and 8 event points are used in the continuous scheduling formulation. Let's consider here that all the processing times are uncertain parameters and have a variability of 15%. In this case we assume zero cost for all the raw materials, and product prices are set as 10 for both Product 1 and Product 2.

Ben-Tal's formulation is studied with the reliability level set as 75% and 62.5% and no infeasibility tolerance is assumed for all the formulations. For Bertsimas and Sim's formulation, note that the maximum number of uncertain parameters is 1 for the duration constraints, so $\Gamma^t \in [0, 1]$.

Table 3.3 Comparison of the robust formulations for processing time uncertainty

	Nominal	Soyster	Ben-Tal		Bertsimas and Sim		
					$\Gamma^t=0$	$\Gamma^t=0.5$	$\Gamma^t=1$
Objective	2657.9	2140.9	1676.1	1201.8	2657.9	2423.6	2140.9
Upper probability of constraint violation	-	-	0.75	0.625	0.75	0.625	0.5
CPU time (s)	4.8	58.3	129.2	109.0	4.9	12.2	69.6
Continuous variables	401	401	401		473		
Binary variable	64	64	64		64		
Constraints	884	884	884		948		

The results in Table 3.3 lead to the same conclusions as in Table 3.2. Comparing to deterministic formulation, Soyster's formulation and Ben-Tal's formulation has the same variable and constraint number, because the uncertain parameters are coefficients of binary variables and no auxiliary variable is added. Bertsimas and Sim's formulation has more constraints and variables but it is still more efficient than Ben-Tal's formulation because the linearity of the formulation.

(c) Demand uncertainty

In this case, the scheduling horizon is set as 8 hours and 8 event points are used in the continuous scheduling formulation. Let's consider that both the demand of product 1 and product 2 have 50% variability level, they both take value in [25, 75] and the nominal value is 50. A reliability level of 75% is set for the Ben-Tal's formulation, since smaller reliability level 62.5% cause the problem to be infeasible. Budget parameter value for Bertsimas and Sim's formulation takes value in [0, 1] because only one demand parameter is uncertain in the demand constraints.

Table 3.4 Comparison of the robust formulations for demand uncertainty

	Nominal	Soyster	Ben-Tal		Bertsimas and Sim		
					$\Gamma^d=0$	$\Gamma^d=0.5$	$\Gamma^d=1$
Objective	1088.75	infeasible	942.80	Infeasible	1088.75	688.05	infeasible
Upper probability of constraint violation	-	-	0.75	0.625	0.75	0.625	-
CPU time (s)	29.0	-	60.9	-	28.7	181.8	-
Continuous variables	401	401	401		411		
Binary variables	64	64	64		64		
Constraints	884	884	884		893		

The results in Table 3.4 illustrate similar trends as the previous case studies. Soyster's formulation becomes infeasible here, which means that the "worst-case" demand uncertainty cause infeasible schedules. Correspondingly, Bertsimas and Sim's formulation becomes infeasible when the budget parameter is set at the largest value. So the Bertsimas and Sim's robust formulation will be feasible for different budget parameter in the range of zero to maximum uncertain coefficient of a constraint only if the worst case feasibility is ensured. Furthermore, in this case, Soyster's formulation and Ben-Tal's formulation both have same problem size as the deterministic formulation because the uncertain parameter is on the right hand side of the constraint and it can be viewed as coefficient of binary variable with fixed value 1.

Summarizing, three different kinds of uncertainties in scheduling have been studied and compared. From the results we can see that the size of all the robust formulations do not increase a lot because the increase in the number of constraints and variables is at the same scale as the number of the uncertain parameters. Moreover, since most decision variables in scheduling formulation are positive or binary, we can further reduce the number of auxiliary variables and boundary constraints for the auxiliary variables. Comparing the three different robust formulations, Soyster's worst case formulation is the most conservative formulation and does not have the flexibility of adjusting the degree of conservatism. The other two robust counterpart formulations are able to adjust the solution robustness either through the constraint violation probability or budget parameter. Bertsimas and Sim's formulation involve relative more constraints and continuous variables when addressing processing time and demand uncertainty, but it has more flexibility in controlling the degree of conservatism and also avoids the solution of mixed integer nonlinear optimization problem, thus the solution efficiency is greatly improved; although Ben-Tal's formulation can also adjust the degree of conservatism with the probability of constraint violation, it tends to be a more conservative formulation and thus more likely to generate infeasible problem; on the other hand, the feasibility of Bertsimas and Sim's formulation can be ensured when the feasibility of the worst case formulation is satisfied. So, Bertsimas and Sim's formulation will be adopted in our research as the method of generating robust preventive schedule as shown in following subsection.

(d) Systematically considering all uncertainties

Finally, we consider all uncertain parameters simultaneously including all processing times with variability 15%, the demand of P1 and P2 with variability 50%, and the prices of P1 and P2 with variability level 5%. The scheduling horizon is set as 8 hours and 8 event points are used in the continuous scheduling formulation.

Table 3.5 Solution data for example 2 with all uncertainties

Budget parameter (Γ^p, Γ^d, Γ^t)	(0,0,0)	(0.5,0.3,0.3)	(1,0.3,0.5)	(2,0.3,0.5)
Objective	1088.75	615.5	431.3	402.6
CPU time (s)	26.7	237.2	231.3	330.5
Continuous variables	494			
Binary variables	64			
Constraints	972			

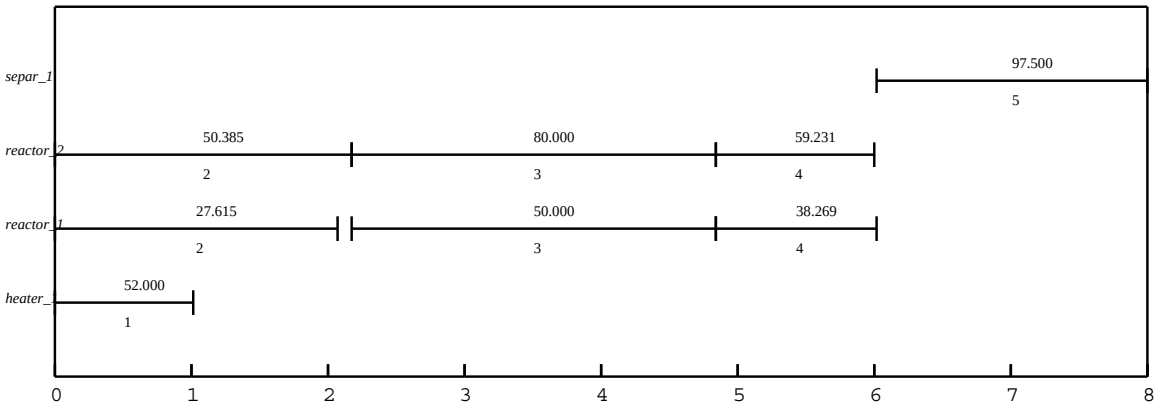


Figure 3.2 Nominal schedule for example 1 (x: hours, y: equipment)

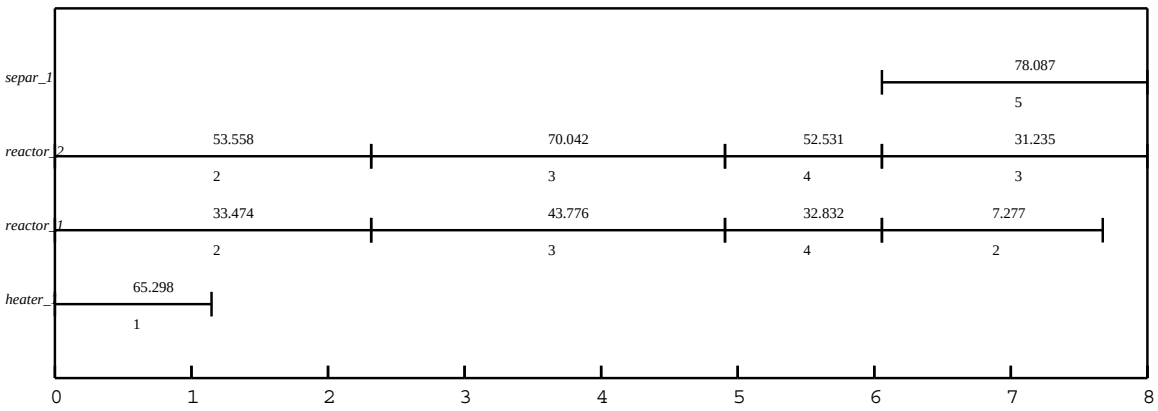


Figure 3.3 Robust schedule for example 1 (Γ^p=0.5, Γ^d=0.3, Γ^t=0.3)

Several different budget parameter combinations are used to solve the problem considering all the uncertainties. The results are summarized in Table 3.5, which shows the relationship between the profit objective and the budget parameters: higher budget parameters result in more conservative solution with

larger feasibility but smaller profit. The profit of the nominal schedule with zero budget parameter value is 1088.75, and is shown in Figure 3.2. When we choose the budget parameter as Γ^{p}=0.5, Γ^{d}=0.3 and Γ^{t}=0.3, which means the corresponding price, demand and duration constraints may be violated with maximum probability 61.3%, 67.5% and 67.5% respectively, the profit reduces to 615.5 (43.5% decrease). Since product 2 (P2) is more valuable than product 1 (3.P1), the production of the nominal schedule (Figure 3.2) leads to a production of 52 units P1 and 87.5 units P2 which aims at producing more P2 for largest profit. On the other hand, the robust schedule is shown in Figure 3.3 which tends to generate a feasible schedule that covers more uncertain range and has a production of 58 units P1 and 70.3 units P2 because the demand of both P1 and P2 is in [25, 75]. Moreover, the robust schedule aims at generating feasible operations considering the processing time variability, e.g., in the separation stage, less amount of materials is processed in the robust schedule (78.087) than in the nominal schedule (97.5) such that the task will finish in the given time horizon considering the processing time variability.

3.4.2 Example 2

This example is taken from (Wu & Ierapetritou, 2003). Here, four products are produced through eight tasks from three feeds. There are nine intermediates in the system as shown in Figure 3.4. Detail process data are shown in Table 3.6. In all, six different units are required for the whole process. This case study is considered since it is larger than the previous example to illustrate the scale-up of the robust-counterpart formulation.

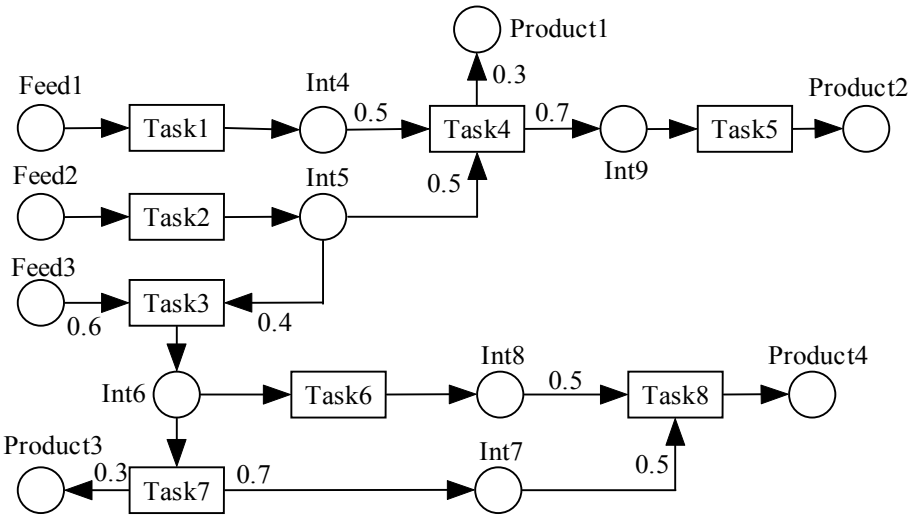


Figure 3.4 STN of example 2

Table 3.6 Process data for Example 2

Unit	Capacity	Suitability	Processing time
Unit1	1000	Task 1	1
Unit2	2500	Task 3,7	1
Unit3	3500	Task 4	1
Unit4	1500	Task 2	1
Unit5	1000	Task 6	1
Unit6	4000	Task 5,8	1
State	Storage capacity	Initial amount	
Feed 1,2,3	Unlimited	0.0	
Int4	1000	0.0	
Int5	1000	0.0	
Int6	1500	0.0	
Int7	2000	0.0	
Int8	0	0.0	
Int9	3000	0.0	
Products 1,2,3,4	Unlimited	0.0	

The scheduling horizon is 18 hours and 15 event points are used in the continuous scheduling formulation. The nominal values of all the processing times are 1; the nominal price of Product1~Product4 are 18, 19, 20 and 21, respectively; the nominal demand of Product1~Product4 are 6000, 8000, 2000 and 8000, respectively. For the uncertainty problem we assume all the processing times have 20% variability level, all the product demands have 30% variability level, and all the product price have 10% variability level.

To generate robust schedules for this problem, the robust optimization formulation proposed by Bertsimas and Sim is also used here. Different budget parameter combinations are considered as shown in Table 3.7 (the optimality gap in CPLEX solver is set as 0.01).

Table 3.7 Solution data of example 2

	Determinisitic formulation	Bertsimas and Sim's Robust formulation		
Budget parameter (Γ^p, Γ^d, Γ^t)	-	(0,0,0)	(1,0.3,0.3)	(2,0.4,0.4)
Objective	738835	738835	674567	634677
CPU time (s)	7.0	12.5	147.7	476.9
Continuous variables	1471	1619		
Binary variables	720	720		
Constraints	2416	2554		

The deterministic scheduling formulation for example 2 is first solved, and the generated schedule is shown in Figure 3.5. Then the robust formulation with different budget parameter combination is used to generate the robust schedules. The robust schedule with minimum budget parameter value equal to 0 is actually same as the nominal deterministic schedule. The other two robust schedules are shown in Figure 3.6 and Figure 3.7.

The production of Product1~Product4 in the nominal schedule are (7522, 16418, 2835, 11180), the production in the robust schedule (Figure 3.6) are (7200, 15096, 2522, 11256), the production in the robust schedule (Figure 3.7) are (7150, 15400, 2459, 10235), compared to the demand uncertainty: (6000, 8000, 2000, 8000)$\cdot(1\pm30\%)$, it can be seen that as the budget parameter value increases, the production of the robust schedule tends to reduce the production so to satisfy the duration requirement under the uncertain processing time with higher robustness; on the other hand, the total profit decreases. Note that when the budget parameters increase leading to more conservative solutions, the computation time becomes longer. This is due to the fact that the feasible space will become smaller as the robustness requirement increases.

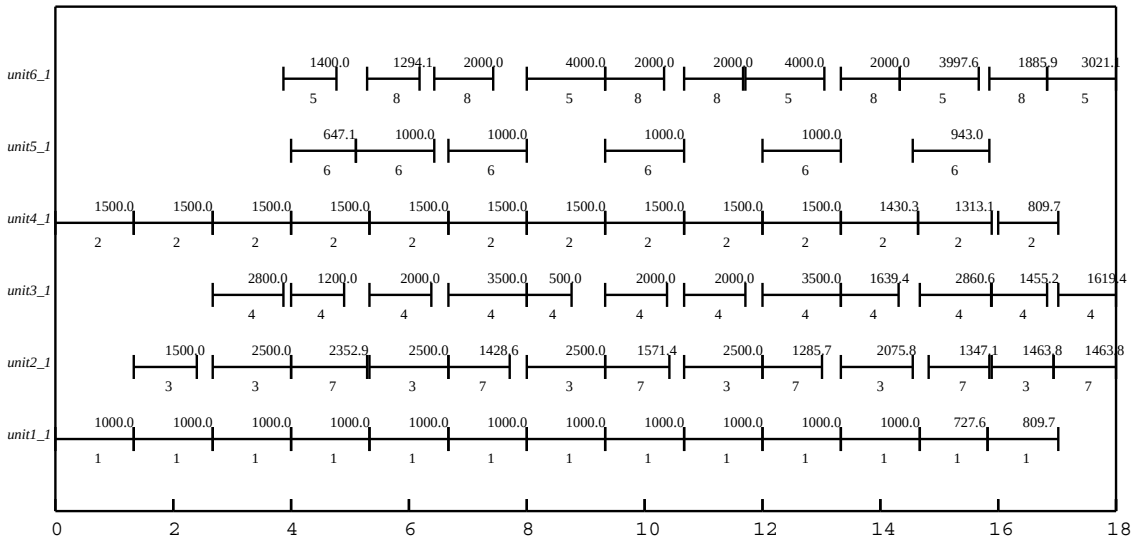


Figure 3.5 Nominal schedule for example 2

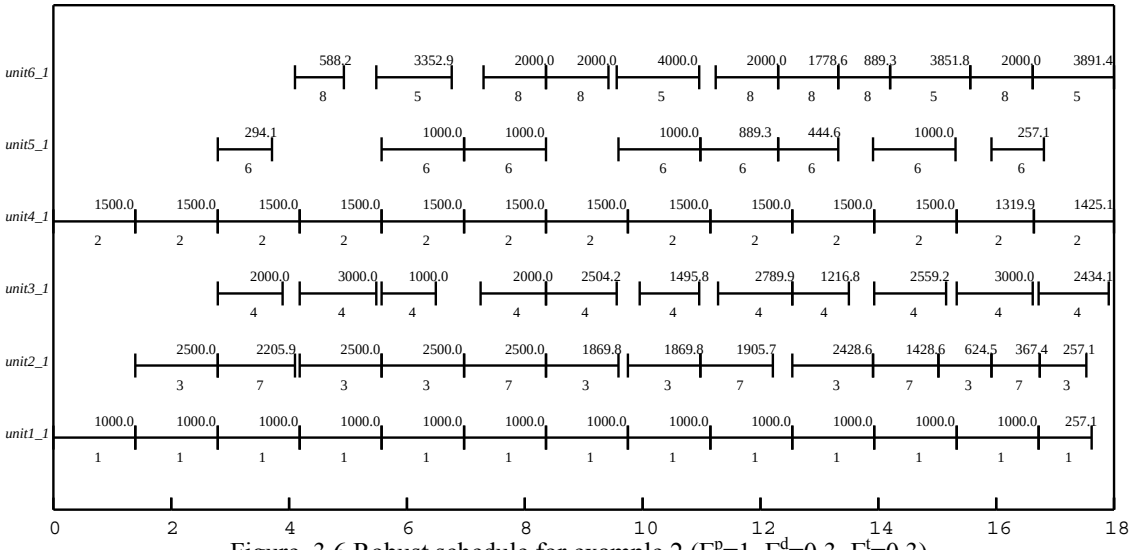


Figure 3.6 Robust schedule for example 2 (Γ^p=1, Γ^d=0.3, Γ^t=0.3)

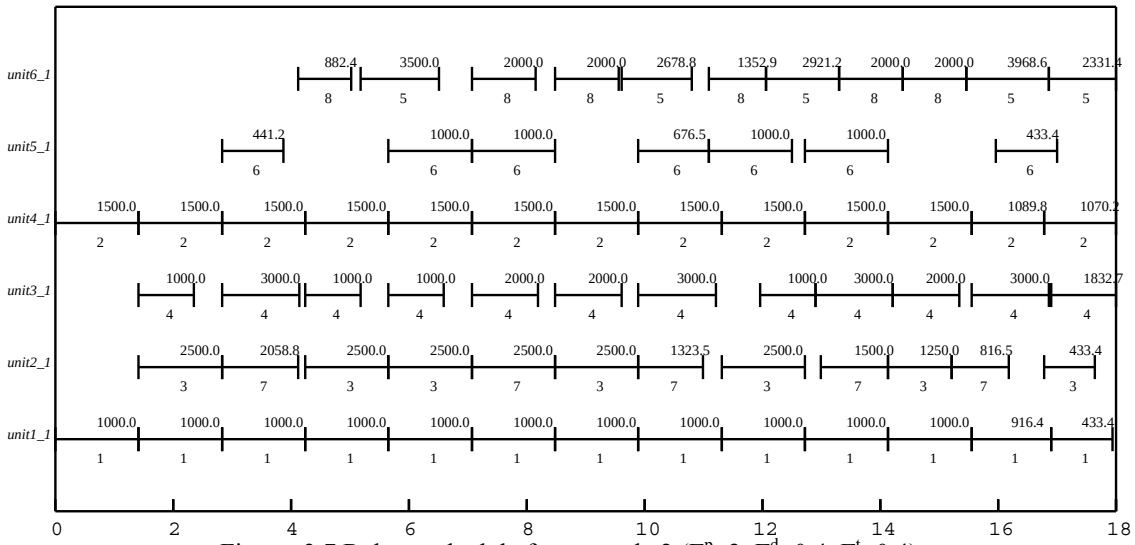


Figure 3.7 Robust schedule for example 2 (Γ^p=2, Γ^d=0.4, Γ^t=0.4)

3.5 Summary

To generate robust preventive schedules that can address the different parameter uncertainties in process scheduling problem, a lot of efforts have been made in the past, especially in the direction of scenario based stochastic scheduling. However, the scenario based methodologies have a main drawback that they cannot avoid the exponential increase of the problem size when the number of parameters increases. Robust counterpart optimization based preventive scheduling avoids this type of complexity. In this work we studied three robust counterpart optimization formulations and compared their performance in uncertain scheduling. The results showed that the "budget parameter" based formulation proposed by (Bertsimas & Sim, 2003) is appropriate for uncertain scheduling problems with its unique advantages that it does not increase substantially the problem size, maintains its linearity, and its ability to control the degree of conservatism for every constraint and guarantee the feasibility for the robust optimization problem with the use of budget parameter.

Nomenclature

M_l	index set for the uncertain coefficients in the l-th constraint
$\tilde{a}_{lm}$	true values of the coefficient parameter
a_{lm}	nominal values of the coefficient parameter
$\hat{a}_{lm}$	variation amplitude of the coefficient parameter
u_m	auxiliary variable incorporated for robust formulation
ξ_{lm}	independent random variable symmetrically distributed in interval [-1, 1]
Ω_l	constant (with constraint violation probability $\kappa_l = e^{-\Omega_l^2/2}$)
κ_l	constraint violation probability
z_{lm}	auxiliary variable incorporated for robust formulation
Γ_l	budget parameter
$\lfloor \Gamma_l \rfloor$	biggest integer not greater than Γ_l
S_l	subset that contains $\lfloor \Gamma_l \rfloor$ uncertain parameters of the l-th constraint
t_l	index that represents an additional uncertain parameter if Γ_l is not an integer
q_{lm}	dual variable

Chapter 4

Reactive Scheduling

Abstract: To address the various disruptive events that occur during process operations, reactive scheduling is commonly used. However, a major limitation of the existing reactive scheduling techniques is the response time, which might cause significant delay while the generation of a new schedule takes place. In this chapter, a novel approach is proposed to improve the efficiency of reactive scheduling and avoid the resolution of a complex optimization problem when uncertain event occurs during the scheduling period. In the proposed method, reactive schedule is obtained from the solution of multiparametric programming problem which is solved ahead of time and covers all possible outcomes of future uncertainty. The multiparametric programming problem is derived from a new reactive scheduling formulation which integrates disruptive events (rush order and machine breakdown) as uncertain parameters in the process modeling.

4.1 Introduction

Reactive scheduling, which is also called rescheduling, takes place when the schedule is implemented based on up-to-date information regarding the state of the system. It requires the modification of the existing schedule during the manufacturing process to adapt to changes (uncertainty) such as rush order arrivals, order cancellations or machine breakdowns. For this type of uncertainty there is not enough information prior to realization of the uncertain parameters that will allow a protective action, so almost all the methods in the literature aim to resolve a rescheduling problem once the disruptive events occur.

The reactive scheduling actions are based on various underlying strategies. It can rely on simple techniques or heuristic rules to seek a quick schedule consistency restoration. One of the earliest efforts in reactive scheduling was reported by (Cott & Macchietto, 1989) who considered fluctuations of processing times and used a shifting algorithm to modify the starting times of processing steps of a batch by the maximum deviation between the expected and actual processing times of all related processing steps.

(Kanakamedala et al., 1994) developed a least-impact heuristic approach with two levels that allows time shifting and unit replacement in multipurpose batch plants. (Huercio et al., 1995) proposed a reactive scheduling technique to deal with variations in task processing times and equipment availability. They generated a set of decision trees using alternative unit assignments, each based on a conflict in the real production schedule caused by a deviation between the real schedule and the nominal schedule. Branches of the trees are then pruned according to heuristic equipment selection rules. (Sanmartí et al., 1997) extended this work to cover unexpected equipment failure. (Rodrigues et al., 1996) also considered uncertain processing times and proposed a rolling-horizon approach which incorporates a look-ahead procedure to avoid possible violations of future due dates. (Honkomp et al., 1997) proposed a reactive scheduling framework for processing time variations and equipment breakdown by coupling a deterministic schedule optimizer with a simulator that introduces stochastic events where two different formulations of time are considered. A number of rescheduling strategies were proposed and heuristics were used to locate critical tasks which can be modified to make the nominal schedule less susceptible to the effects of processing time variability.

On the other hand, a number of the techniques presented in the literature involve a full scheduling of the tasks that have to be executed after the unexpected event occurs through mathematical programming approaches relying mostly on mixed integer linear programming (MILP). (Roslöf et al., 2001) developed an MILP-based heuristic algorithm by iteratively releasing a set of jobs from a nominal schedule and optimally reallocating them, where the complexity of the problem is controlled through the number of simultaneously released jobs. (Ruiz et al., 2001) presented a fault diagnosis system that interacts with a schedule optimizer for multipurpose batch plants to perform reactive scheduling in the event of processing time variability or unit unavailability. (Méndez & Cerdá, 2003) proposed a rescheduling method by first reassigning resources to tasks that still need to be processed and then reordering the sequence of processing tasks for each resource item. They considered start time shifting, local reordering, and unit reallocation of old batches as well as insertion of new batches. This work was extended in (Méndez & Cerdá, 2004) to include limited discrete renewable resources where only start-time shifting, local batch reordering, and resource reallocation of existing batches are allowed. (Vin & Ierapetritou, 2000) considered the rescheduling of multiproduct and multipurpose batch plants in the event of machine breakdown or rush order arrival. Full-scale rescheduling of

each production schedule is avoided by fixing binary variables for a subset of tasks from the original production schedule. The fixing of tasks results in a reduced computational effort required to solve the resulting MILP problem. (Janak & Floudas, 2006) presented a similar framework where the fixed subset of tasks is determined using a detailed set of rules that reflect the production needs and can be modified for different production facilities. By fixing a subset of tasks a reduced computational effort is required to solve the resulting MILP problem.

As shown from the literature, a major consideration for reactive scheduling is the response time. If the computation time is large the production may be significantly delayed while the new schedule is developed. In this chapter, we proposed a framework to solve the reactive scheduling problems using multiparametric programming technique, which will greatly improve the efficiency of the rescheduling approach because the new schedule is obtained from the solution of parametric programming problem which was solved before the occurrences of disruptive events, thus completely avoiding the solution of the rescheduling optimization problem.

Parametric programming serves as an analytic tool by mapping the uncertainties in the optimization problem to optimal alternatives. From this point of view, parametric programming provides the exact mathematical solution of the optimization problem under uncertainty. In the literature, there are not many records on the application of parametric programming in process scheduling problem. (Ryu & Pistikopoulos, 2007) has reported the application of parametric programming to a zero-waiting scheduling problem, where they studied the parametric solution under processing time uncertainty for zero-wait batch processes, but the scheduling formulation does not consider the executed tasks so it is not able to address the reactive scheduling problem. (Li & Ierapetritou, 2007b) proposed an efficient multiparametric programming framework and applied it to general scheduling problem to study the effect of uncertain product demand, price and processing time on preventive scheduling problem. In this chapter, the work is further extended to study the reactive scheduling problem.

The rest of this chapter is organized as follows. A general reactive scheduling formulation is presented in next section for two kinds of uncertainty: rush order and machine breakdown. Then, the multiparametric programming method presented in Chapter 2 will be applied to solve the parametric reactive scheduling problems and the chapter is finally summarized in the last section.

4.2 Reactive scheduling formulation

4.2.1 General idea

The reactive scheduling model studied in this chapter is based on the deterministic model (2.8) presented in Chapter 2. It should be noticed that the proposed methodology for reactive scheduling is not tight to the specific deterministic model. Any schedule modeling framework can be used as long as it can be formulated as a MILP problem such as the ones presented by (Floudas & Lin, 2004), (Méndez et al., 2006), (Maravelias & Grossmann, 2006).

To apply the parametric programming method on reactive scheduling, it is necessary to develop an effective way to model the disruptive uncertainty into the scheduling formulation. An important fact for formulating the reactive scheduling is that the tasks that have already executed or started cannot be changed. In a previously published work (Vin & Ierapetritou, 2000), those binary variables that corresponds to a subset of tasks of the original production schedule that have been executed are fixed while generating the reactive schedule. However this method only solves one reactive schedule after the uncertain event occurs.

Our target in this work is to develop a new reactive scheduling formulation to consider all possible uncertain outcomes by formulating the uncertain events as uncertain parameters into the optimization problem. The basic strategy is to generate a complete reschedule but fix the executed tasks with a set of binary indicator variables $y_{i,j,n}$, which denote whether a task is executed ($y_{i,j,n}$=1) or not ($y_{i,j,n}$=0). The rules and corresponding constraints to identify these indicator variables will be presented in the next two subsections for the specific disruptive event, rush order or machine breakdown. Here, the constraints that ensure that the executed tasks are fixed using the indicator variables ($y_{i,j,n}$) are described as follows:

$$wv_{i,j,n} = wv_{i,j,n}^{old}, \quad \text{if } y_{i,j,n} = 1 \quad \forall i \in I, \forall j \in J_i, \forall n \in N \tag{4.1}$$

$$b_{i,j,n} = b_{i,j,n}^{old} wv_{i,j,n}^{old}, \quad \text{if } y_{i,j,n} = 1 \quad \forall i \in I, \forall j \in J_i, \forall n \in N \tag{4.2}$$

$$Ts_{i,j,n} = Ts_{i,j,n}^{old} wv_{i,j,n}^{old}, \quad \text{if } y_{i,j,n} = 1 \quad \forall i \in I, \forall j \in J_i, \forall n \in N \tag{4.3}$$

$$Tf_{i,j,n} = Tf_{i,j,n}^{old} wv_{i,j,n}^{old}, \quad \text{if } y_{i,j,n} = 1 \quad \forall i \in I, \forall j \in J_i, \forall n \in N \tag{4.4}$$

Constraint (4.1) ensures that if a task i assigned to unit j at event point n has been executed, then the corresponding variable $wv_{i,j,n}$ has to be fixed to the value $wv_{i,j,n}^{old}$ that represents the task in the original schedule. Similarly, constraints (4.2), (4.3) and (4.4) ensure that batch size, task starting and completion time are fixed at the same values as the ones in the original schedule.

The logical constraints (4.1)-(4.4) are transformed to mathematical programming constraints as follows.

$$wv_{i,j,n}^{old} - (1 - y_{i,j,n}) \le wv_{i,j,n} \le wv_{i,j,n}^{old} + (1 - y_{i,j,n}) \qquad \forall i \in I, \forall j \in J_i, \forall n \in N \tag{4.5}$$

$$b_{i,j,n}^{old} wv_{i,j,n}^{old} - b_{i,j}^{UB}(1 - y_{i,j,n}) \le b_{i,j,n} \le b_{i,j,n}^{old} wv_{i,j,n}^{old} + b_{i,j}^{UB}(1 - y_{i,j,n}) \ \forall i \in I, \forall j \in J_i, \forall n \in N \tag{4.6}$$

$$Ts_{i,j,n}^{old} wv_{i,j,n}^{old} - U(1 - y_{i,j,n}) \le Ts_{i,j,n} \le Ts_{i,j,n}^{old} wv_{i,j,n}^{old} + U(1 - y_{i,j,n}) \ \forall i \in I, \forall j \in J_i, \forall n \in N \tag{4.7}$$

$$Tf_{i,j,n}^{old} wv_{i,j,n}^{old} - U(1 - y_{i,j,n}) \le Tf_{i,j,n} \le Tf_{i,j,n}^{old} wv_{i,j,n}^{old} + U(1 - y_{i,j,n}) \ \forall i \in I, \forall j \in J_i, \forall n \in N \tag{4.8}$$

where $b_{i,j}^{UB}$ is the upper bound of the batch size, and U is the upper bound of the scheduling time horizon. Constraint (4.5) is equivalent to constraint (4.1). This can be shown as follows: if $y_{i,j,n} = 1$, constraint (4.5) becomes $wv_{i,j,n}^{old} \le wv_{i,j,n} \le wv_{i,j,n}^{old}$, i.e., $wv_{i,j,n}$ is fixed to $wv_{i,j,n}^{old}$; if on the other hand $y_{i,j,n} = 0$, (4.5) becomes $wv_{i,j,n}^{old} - 1 \le wv_{i,j,n} \le wv_{i,j,n}^{old} + 1$, which is a redundant constraint since it is satisfied for any value of the binary variable $wv_{i,j,n}$. Similarly, constraints (4.6), (4.7) and (4.8) are equivalent to logical constraints (4.2), (4.3) and (4.4), respectively.

In order to determine the value of $y_{i,j,n}$ additional constraints are required depending on the nature of the disruptive event: rush order or machine breakdown. In the next two subsections we specify the rules and constraints that determine the value of $y_{i,j,n}$ and present the complete reactive scheduling formulation.

4.2.2 Rush order

Once a rush order arrives during the scheduling execution process, all the tasks that have already started should be identified as executed. When this rule is implemented on the original schedule solution, the value of $y_{i,j,n}$ can be identified. However this rule should also be implemented on the complete reschedule so that the reactive schedule does not change the schedule history, otherwise the reactive schedule can generate

"wrong" tasks that start before the disruptive event which do not exist in the original schedule. So, we can define the indicator binary variable $y_{i,j,n}$ as follows:

$$y_{i,j,n} = \begin{cases} 1, & \text{if } Ts_{i,j,n}^{old} < T^{rush} \text{ and } Ts_{i,j,n} < T^{rush} \\ 0, & \text{if } Ts_{i,j,n}^{old} \geq T^{rush} \text{ and } Ts_{i,j,n} \geq T^{rush} \end{cases}$$

which can be mathematically formulated as follows:

$$Ts_{i,j,n}^{old} + \varepsilon - U(1 - y_{i,j,n}) \leq T^{rush} \leq Ts_{i,j,n}^{old} + Uy_{i,j,n} \quad \forall i \in I, \forall j \in J_i, \forall n \in N \tag{4.9}$$

$$Ts_{i,j,n} + \varepsilon - U(1 - y_{i,j,n}) \leq T^{rush} \leq Ts_{i,j,n} + Uy_{i,j,n} \quad \forall i \in I, \forall j \in J_i, \forall n \in N \tag{4.10}$$

Constraint (4.9) and (4.10) corresponds to the definition of the binary variables $y_{i,j,n}$. Constraints (4.9) can be verified as follows: if task i in unit j at event point n starts before the rush order arrives ($Ts_{i,j,n}^{old} < T^{rush}$), then $y_{i,j,n}$ must take value 1 as a binary variable because if $y_{i,j,n} = 0$ constraint (4.9) will become $T^{rush} \leq Ts_{i,j,n}^{old}$ which contradicts the fact that $Ts_{i,j,n}^{old} < T^{rush}$, however if $y_{i,j,n} = 1$ constraint (4.9) takes the form $Ts_{i,j,n}^{old} + \varepsilon \leq T^{rush} \leq Ts_{i,j,n}^{old} + U$, which verifies the assumption ($Ts_{i,j,n}^{old} < T^{rush}$) since ε is a small positive number and the inequality on the right hand side is redundant; similarly, if task i in unit j at event point n starts at or after the rush order arrival time ($Ts_{i,j,n}^{old} \geq T^{rush}$), $y_{i,j,n}$ must be 0 since in this case constraint (4.9) satisfies this assumption whereas the value of 1 results in a contradictory conclusion ($Ts_{i,j,n}^{old} < T^{rush}$). Thus the value of $y_{i,j,n}$ is defined by constraint (4.9). Constraint (4.10) defines the variables $y_{i,j,n}$ for the tasks in reactive schedule in the same way.

Furthermore, the demand constraint should be updated as following to account for the new demand in the rush order:

$$\sum_n d_{s,n} \geq r_s^{rush} \qquad \forall s \in S \tag{4.11}$$

where r_s^{rush} corresponds to the updated demand after the rush order arrival.

Thus, the reactive scheduling problem considering rush order uncertainty is formulated with the original constraints from deterministic model and additional constraints defined as above. The reactive scheduling objective can be selected based on preferred performance, e.g., minimizing the makespan to fulfill the updated order. The complete problem formulation is given by Formulation A as follows. It should be noticed

that this formulation covers any case of a rush order arrival including the time of arrival, new orders, and modification or cancellation of existing orders. Also it is not restricted by the number of different products in the order.

Problem 4.A: Reactive scheduling formulation for rush order uncertainty

$$\min H \tag{4.A1}$$

s.t.

$$\sum_{i\in I_j} wv_{i,j,n} \le 1 \qquad \forall j \in J, \forall n \in N \tag{4.A2}$$

$$st_{s,n} = st_{s,n-1} - d_{s,n} - \sum_{i\in I_s} \rho^{P}_{s,i} \sum_{j\in J_i} b_{i,j,n} + \sum_{i\in I_s} \rho^{c}_{s,i} \sum_{j\in J_i} b_{i,j,n-1} \quad \forall s \in S, \forall n \in N \tag{4.A3}$$

$$st_{s,n} \le st_s^{\max} \qquad \forall s \in S, \forall n \in N \tag{4.A4}$$

$$v_{i,j}^{\min} wv_{i,j,n} \le b_{i,j,n} \le v_{i,j}^{\max} wv_{i,j,n} \qquad \forall i \in I, \forall j \in J_i, \forall n \in N \tag{4.A5}$$

$$\sum_n d_{s,n} \ge r_s^{rush} \qquad \forall s \in S \tag{4.A6}$$

$$Tf_{i,j,n} = Ts_{i,j,n} + \alpha_{i,j} wv_{i,j,n} + \beta_{i,j} b_{i,j,n} \qquad \forall i \in I, \forall j \in J_i, \forall n \in N \tag{4.A7}$$

$$Ts_{i,j,n+1} \ge Tf_{i,j,n} - U(1 - wv_{i,j,n}) \qquad \forall i \in I, \forall j \in J_i, \forall n \in N \tag{4.A8}$$

$$Ts_{i,j,n+1} \ge Tf_{i',j,n} - U(1 - wv_{i',j,n}) \qquad \forall i, i' \in I_j, \forall j \in J, \forall n \in N \tag{4.A9}$$

$$Ts_{i,j,n+1} \ge Tf_{i',j',n} - U(1 - wv_{i',j',n}) \qquad \forall i, i' \in I_j, i \ne i', \forall j, j' \in J, \forall n \in N \tag{4.A10}$$

$$Ts_{i,j,n+1} \ge Ts_{i,j,n} \qquad \forall i \in I, \forall j \in J_i, \forall n \in N \tag{4.A11}$$

$$Tf_{i,j,n+1} \ge Tf_{i,j,n} \qquad \forall i \in I, \forall j \in J_i, \forall n \in N \tag{4.A12}$$

$$Ts_{i,j,n} \le H \qquad \forall i \in I, \forall j \in J_i, \forall n \in N \tag{4.A13}$$

$$Tf_{i,j,n} \le H \qquad \forall i \in I, \forall j \in J_i, \forall n \in N \tag{4.A14}$$

$$wv_{i,j,n}^{old} - (1 - y_{i,j,n}) \le wv_{i,j,n} \le wv_{i,j,n}^{old} + (1 - y_{i,j,n}) \quad \forall i \in I, \forall j \in J_i, \forall n \in N \tag{4.A15}$$

$$b_{i,j,n}^{old} wv_{i,j,n}^{old} - b_{i,j}^{UB}(1 - y_{i,j,n}) \le b_{i,j,n} \le b_{i,j,n}^{old} wv_{i,j,n}^{old} + b_{i,j}^{UB}(1 - y_{i,j,n}) \quad \forall i \in I \quad \forall j \in J_i, \quad \forall n \in N \tag{4.A16}$$

$$Ts_{i,j,n}^{old} wv_{i,j,n}^{old} - U(1 - y_{i,j,n}) \le Ts_{i,j,n} \le Ts_{i,j,n}^{old} wv_{i,j,n}^{old} + U(1 - y_{i,j,n}) \quad \forall i \in I, \quad \forall j \in J_i, \quad \forall n \in N, \tag{4.A17}$$

$$Tf_{i,j,n}^{old} wv_{i,j,n}^{old} - U(1 - y_{i,j,n}) \le Tf_{i,j,n} \le Tf_{i,j,n}^{old} wv_{i,j,n}^{old} + U(1 - y_{i,j,n}) \quad \forall i \in I \quad \forall j \in J_i \quad \forall n \in N \tag{4.A18}$$

$$Ts^{old}_{i,j,n} + \varepsilon - U(1 - y_{i,j,n}) \le T^{rush} \le Ts^{old}_{i,j,n} + Uy_{i,j,n} \quad \forall i \in I, \forall j \in J_i, \forall n \in N \tag{4.A19}$$

$$Ts_{i,j,n} + \varepsilon - U(1 - y_{i,j,n}) \le T^{rush} \le Ts_{i,j,n} + Uy_{i,j,n} \quad \forall i \in I, \forall j \in J_i, \forall n \in N \tag{4.A20}$$

4.2.3 Machine breakdown

To incorporate machine breakdown within the reactive scheduling formulation, the following rules should be included: if a unit j^* breaks down at time T^{break} and requires repair/maintenance time of T^{maint}, then:

a) All the tasks in $j \in J, j \neq j^*$ should be identified as executed if they start before T^{break};

b) All the tasks in j^* should be identified as executed if they finish at or before T^{break}.

Note that there are different rules for the breakdown units and for the ones that operate normally. For the unit that is broken, we are enforcing rules on task finishing time but not starting time because once the machine is broken, the tasks that have started must be stopped. So, we define the indicator binary variables as follows:

$$y_{i,j,n} = \begin{cases} 1, \text{ if } \begin{cases} Ts^{old}_{i,j,n} < T^{break} \text{ and } Ts_{i,j,n} < T^{break} & \text{for } \forall j \in J, j \neq j^*, \text{ or} \\ Tf^{old}_{i,j,n} \le T^{break} \text{ and } Ts_{i,j,n} \le T^{break} & \text{for } j = j^* \end{cases} \\ 0, \text{ if } \begin{cases} Ts^{old}_{i,j,n} \ge T^{break} \text{ and } Ts_{i,j,n} \ge T^{break} & \text{for } \forall j \in J, j \neq j^*, \text{ or} \\ Tf^{old}_{i,j,n} > T^{break} \text{ and } Ts_{i,j,n} > T^{break} & \text{for } j = j^* \end{cases} \end{cases}$$

Similar to the rush order case, the rules are implemented in the original schedule to identify the value of $y_{i,j,n}$, and in the reactive schedule to ensure that the complete reschedule does not change the schedule history. The definition of $y_{i,j,n}$ is mathematically equivalent with the following constraints:

$$Ts^{old}_{i,j,n} + \varepsilon - U(1 - y_{i,j,n}) \le T^{break} \le Ts^{old}_{i,j,n} + Uy_{i,j,n}, \forall i \in I, \forall j \in J_i, j \neq j^*, \forall n \in N \tag{4.12}$$

$$Tf^{old}_{i,j^*,n} - U(1 - y_{i,j^*,n}) \le T^{break} \le Tf^{old}_{i,j^*,n} + Uy_{i,j^*,n} - \varepsilon, \forall i \in I, \forall n \in N \tag{4.13}$$

$$Ts_{i,j,n} + \varepsilon - U(1 - y_{i,j,n}) \le T^{break} \le Ts_{i,j,n} + Uy_{i,j,n}, \forall i \in I, \forall j \in J_i, j \neq j^*, \forall n \in N \tag{4.14}$$

$$Tf_{i,j^*,n} - U(1 - y_{i,j^*,n}) \le T^{break} \le Tf_{i,j^*,n} + Uy_{i,j^*,n} - \varepsilon, \forall i \in I, \forall n \in N \tag{4.15}$$

Constraints (4.12) are valid for units that do not break down and represent the definition of $y_{i,j,n}$ in the following way: if task i in unit j at event point n starts before the machine breaks down ($Ts^{old}_{i,j,n} < T^{break}$), since constraint (4.12) expresses that $T^{break} \le Ts^{old}_{i,j,n} + Uy_{i,j,n}$, $y_{i,j,n}$ must be 1 since 0 does not satisfy this

constraint; similarly, if the task i in unit j at event point n start at or after the machine breakdown time ($Ts_{i,j,n}^{old} \geq T^{break}$), because constraint (4.12) expresses the requirement $Ts_{i,j,n}^{old} + \varepsilon - U(1 - y_{i,j,n}) \leq T^{break}$, $y_{i,j,n}$ must be 0 since 1 will generate contradictory results. Constraints (4.13) represent the definition of $y_{i,j,n}$ for units that break down, and can be verified in similar way. Constraints (4.14) and (4.15) are for the tasks in reactive schedule and correspond to the same rules as constraints (4.14) and (4.15) respectively and can be verified in the same way.

Furthermore, we should add constraints to change the starting time for tasks that finish after T^{break} in the machine that breaks down:

$$T^{break} + T^{maint} \leq Ts_{i,j^*,n} + Uy_{i,j^*,n} \qquad \forall i \in I, \forall n \in N \tag{4.16}$$

Constraint (4.16) expresses the requirement that if a task has been identified as one that does not finish before breakdown time ($y_{i,j,n} = 0$), it must start after the unit is fixed. For the case that the task finishes before the breakdown occurs ($y_{i,j,n} = 1$), constraint (4.16) is redundant.

The reactive scheduling formulation incorporating machine breakdown is formulated with the objective of maximizing the profit (or minimizing makespan). The complete formulation is given in model B as follows. Note that this formulation covers all possible machine breakdown events including the breakdown of any unit at any time during the schedule execution that require any time for repair/maintenance.

Problem 4.B: Reactive scheduling formulation for machine breakdown uncertainty

$$\max \sum_{s,n} price_s d_{s,n} \text{ or } \min H \tag{4.B1}$$

$$\text{s.t.} \sum_{i \in I_j} wv_{i,j,n} \leq 1 \qquad \forall j \in J, \forall n \in N \tag{4.B2}$$

$$st_{s,n} = st_{s,n-1} - d_{s,n} - \sum_{i \in I_s} \rho_{s,i}^{P} \sum_{j \in J_i} b_{i,j,n} + \sum_{i \in I_s} \rho_{s,i}^{c} \sum_{j \in J_i} b_{i,j,n-1} \quad \forall s \in S, \forall n \in N \tag{4.B3}$$

$$st_{s,n} \leq st_s^{\max} \qquad \forall s \in S, \forall n \in N \tag{4.B4}$$

$$v_{i,j}^{\min} wv_{i,j,n} \leq b_{i,j,n} \leq v_{i,j}^{\max} wv_{i,j,n} \qquad \forall i \in I, \forall j \in J_i, \forall n \in N \tag{4.B5}$$

$$\sum_{n} d_{s,n} \geq r_s \qquad \forall s \in S \tag{4.B6}$$

$$Tf_{i,j,n} = Ts_{i,j,n} + \alpha_{i,j} wv_{i,j,n} + \beta_{i,j} b_{i,j,n} \qquad \forall i \in I, \forall j \in J_i, \forall n \in N \qquad (4.B7)$$

$$Ts_{i,j,n+1} \geq Tf_{i,j,n} - U(1 - wv_{i,j,n}) \qquad \forall i \in I, \forall j \in J_i, \forall n \in N \qquad (4.B8)$$

$$Ts_{i,j,n+1} \geq Tf_{i',j,n} - U(1 - wv_{i',j,n}) \qquad \forall i, i' \in I_j, \forall j \in J, \forall n \in N \qquad (4.B9)$$

$$Ts_{i,j,n+1} \geq Tf_{i',j',n} - U(1 - wv_{i',j',n}) \qquad \forall i, i' \in I_j, i \neq i', \forall j, j' \in J, \forall n \in N \qquad (4.B10)$$

$$Ts_{i,j,n+1} \geq Ts_{i,j,n} \qquad \forall i \in I, \forall j \in J_i, \forall n \in N \qquad (4.B11)$$

$$Tf_{i,j,n+1} \geq Tf_{i,j,n} \qquad \forall i \in I, \forall j \in J_i, \forall n \in N \qquad (4.B12)$$

$$Ts_{i,j,n} \leq H \qquad \forall i \in I, \forall j \in J_i, \forall n \in N \qquad (4.B13)$$

$$Tf_{i,j,n} \leq H \qquad \forall i \in I, \forall j \in J_i, \forall n \in N \qquad (4.B14)$$

$$wv^{old}_{i,j,n} - (1 - y_{i,j,n}) \leq wv_{i,j,n} \leq wv^{old}_{i,j,n} + (1 - y_{i,j,n}) \quad \forall i \in I, \forall j \in J_i, \forall n \in N \qquad (4.B15)$$

$$b^{old}_{i,j,n} wv^{old}_{i,j,n} - b^{UB}_{i,j}(1 - y_{i,j,n}) \leq b_{i,j,n} \leq b^{old}_{i,j,n} wv^{old}_{i,j,n} + b^{UB}_{i,j}(1 - y_{i,j,n}) \quad \forall i \in I, \forall j \in J_i, \forall n \in N \qquad (4.B16)$$

$$Ts^{old}_{i,j,n} wv^{old}_{i,j,n} - U(1 - y_{i,j,n}) \leq Ts_{i,j,n} \leq Ts^{old}_{i,j,n} wv^{old}_{i,j,n} + U(1 - y_{i,j,n}) \quad \forall i \in I, \forall j \in J_i, \forall n \in N \qquad (4.B17)$$

$$Tf^{old}_{i,j,n} wv^{old}_{i,j,n} - U(1 - y_{i,j,n}) \leq Tf_{i,j,n} \leq Tf^{old}_{i,j,n} wv^{old}_{i,j,n} + U(1 - y_{i,j,n}) \quad \forall i \in I, \forall j \in J_i, \forall n \in N \qquad (4.B18)$$

$$Ts^{old}_{i,j,n} + \varepsilon - U(1 - y_{i,j,n}) \leq T^{break} \leq Ts^{old}_{i,j,n} + Uy_{i,j,n}, \forall i \in I, \forall j \in J_i, j \neq j^*, \forall n \in N \qquad (4.B19)$$

$$Tf^{old}_{i,j^*,n} - U(1 - y_{i,j^*,n}) \leq T^{break} \leq Tf^{old}_{i,j^*,n} + Uy_{i,j^*,n} - \varepsilon, \qquad \forall i \in I, \forall n \in N \qquad (4.B20)$$

$$Ts_{i,j,n} + \varepsilon - U(1 - y_{i,j,n}) \leq T^{break} \leq Ts_{i,j,n} + Uy_{i,j,n}, \forall i \in I, \forall j \in J_i, j \neq j^*, \forall n \in N \qquad (4.B21)$$

$$Tf_{i,j^*,n} - U(1 - y_{i,j^*,n}) \leq T^{break} \leq Tf_{i,j^*,n} + Uy_{i,j^*,n} - \varepsilon, \qquad \forall i \in I, \forall n \in N \qquad (4.B22)$$

$$T^{break} + T^{maint} \leq Ts_{i,j^*,n} + Uy_{i,j^*,n} \qquad \forall i \in I, \forall n \in N \qquad (4.B23)$$

4.3 Examples

With the proposed reactive scheduling formulations, the parametric programming algorithm can be applied to solve the corresponding parametric MILP problem and the relationship between the disruptive events and optimal reactive schedule can be obtained from the parametric solution map. This application is illustrated through the following examples.

Example 1

Example 1 involves the production of two products using three raw materials. The state-task-network (STN) representation of this example is shown in Figure 4.1 and the problem data can be found in Chapter 2.

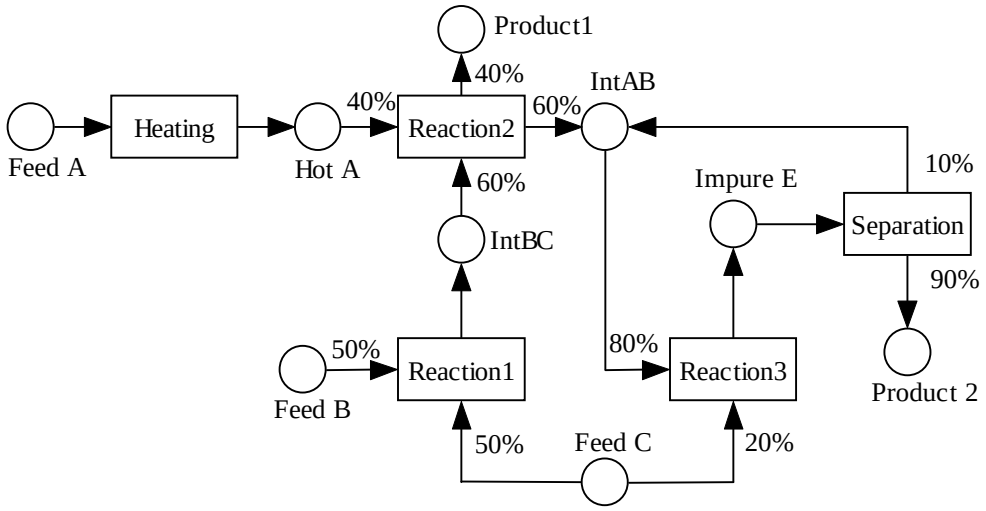


Figure 4.1 State-task-network (STN) representation of example 1

a) Rush order

To study the reactive scheduling problem considering an unexpected rush order, we assume the original deterministic schedule is generated first to satisfy the nominal demand of products P1 and P2 which are both set as 80 units. The deterministic scheduling problem consisting of constraints (4.1)-(4.14) is solved with the objective function of minimizing the makespan. The resulted schedule is shown in Figure 4.2.

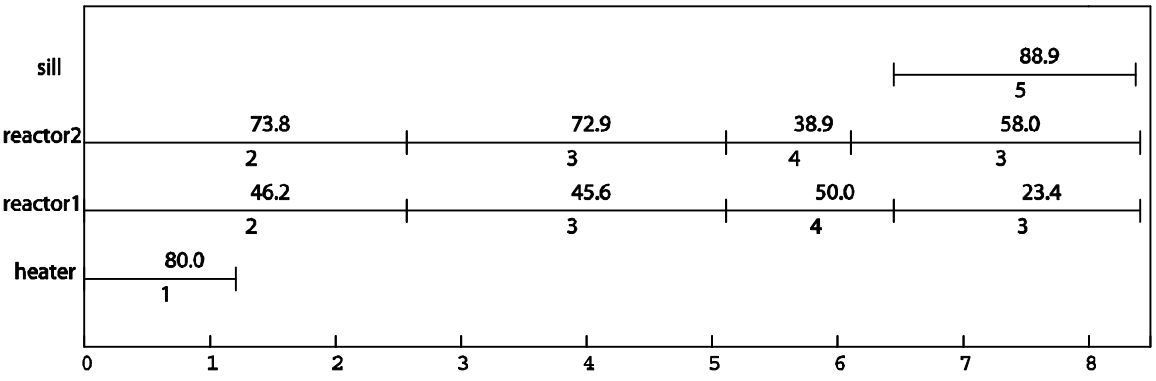


Figure 4.2 Original schedule of example 1 with nominal demand

A rush order of product P1 is investigated with an uncertain demand and an uncertain order arrival time as described in Table 4.1.

Table 4.1 Rush order uncertainty for example 1

	Value	Range
r_{P1}^{rush} (New demand of P1)	θ_1	$70 \le \theta_1 \le 90$
T^{rush} (Order arrival time)	θ_2	$2 \le \theta_2 \le 6$

For the reactive scheduling formulation with rush order uncertainty, since the demand of product is increased to satisfy the additional order, the number of event points used for the original deterministic scheduling might not be enough and can cause the problem to be infeasible. So we need to find the appropriate number of event points for reactive scheduling formulation with the maximum demand using the deterministic formulation and then fix the event point number and solve the parametric problem. For this example, the original number of event points for deterministic formulation is 7, which is the number required for the maximum demand of product P1 (90 units), so the number of event points are fixed at 7 for the reactive scheduling formulation during the multiparametric programming solution process. Then the corresponding multiparametric programming problem is solved and the parametric results are obtained. Figure 4.3 illustrates the exact relationship between the uncertain parameter and the optimal makespan. Figure 4.4 shows the critical regions of the solution and Table 4.2 shows the detail parametric objective in different critical regions.

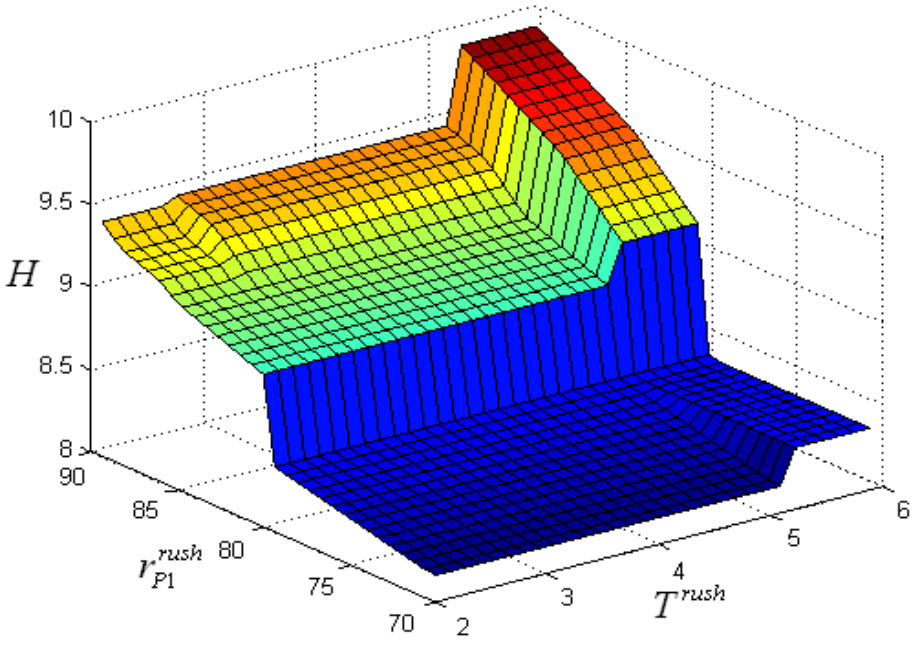


Figure 4.3 Parametric solution of optimal makespan and the rush order

Table 4.2 Parametric objective for example 1 with rush order

Critical Region	Makespan H (hours)	Critical Region	Makespan H (hours)
1	8.1614	7	$0.071 + +0.10667\theta_1$
2	8.3723	8	$5.7892 + 0.041\theta_1$
3	$6.353 + 0.02564\theta_1$	9	$0.0467 + 0.10667\theta_1$
4	$0.6192 + 0.10667\theta_1$	10	$5.769 + 0.04\theta_1$
5	$3.962 + 0.06667\theta_1$	11	$-0.1174 + 0.10667\theta_1$
6	$5.6353 + 0.041\theta_1$		

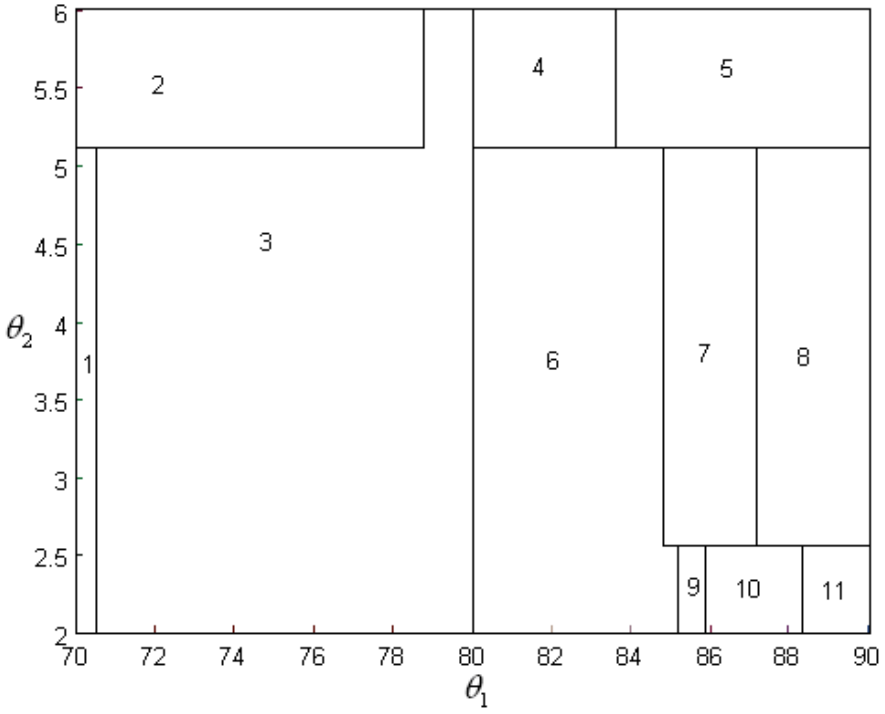


Figure 4.4 Critical regions of example 1 with rush order

As shown in Figures 4.3 and 4.4, the parametric result gives the exact relationship between the uncertain parameter and the scheduling solution, thus the reactive schedule can be obtained explicitly from the parametric solution once the rush order arrives. For example, if a rush order arrives at t=2.2 hour $(\theta_2 = 2.2)$ with 7 units additional demand of P1, thus the new demand of P1 is 87 units $(\theta_1 = 87)$ and the reactive schedule can be obtained from the parametric solution for $(\theta_1, \theta_2) = (87, 2.2)$, which corresponds to critical region 10 with a parametric objective of $5.769 + 0.04\theta_1$ and the integer solution is shown in Table 4.3. Thus the optimal makespan can be evaluated by $5.769 + 0.04 * 87 = 9.249$ and the corresponding schedule is obtained as shown in Figure 4.5.

Table 4.3 Integer solution of critical region 10

$\{(i,j,n) \mid wv_{i,j,n} = 1\}$	$\{(i,j,n) \mid y_{i,j,n} = 1\}$
heating.heater.n0	heating.heater.n0
heating.heater.n1	heating.heater.n1
heating.heater.n2	rxn1.rtr1.n0
rxn1.rtr1.n1	rxn1.rtr1.n1
rxn1.rtr1.n3	rxn1.rtr2.n0
rxn1.rtr2.n1	rxn1.rtr2.n1
rxn2.rtr1.n2	rxn2.rtr1.n0
rxn2.rtr1.n5	rxn2.rtr2.n0
rxn2.rtr2.n2	rxn3.rtr1.n0
rxn2.rtr2.n5	rxn3.rtr2.n0
rxn3.rtr1.n4	sepn.sill.n0
rxn3.rtr2.n3	sepn.sill.n1
sepn.sill.n5	sepn.sill.n2
	sepn.sill.n3

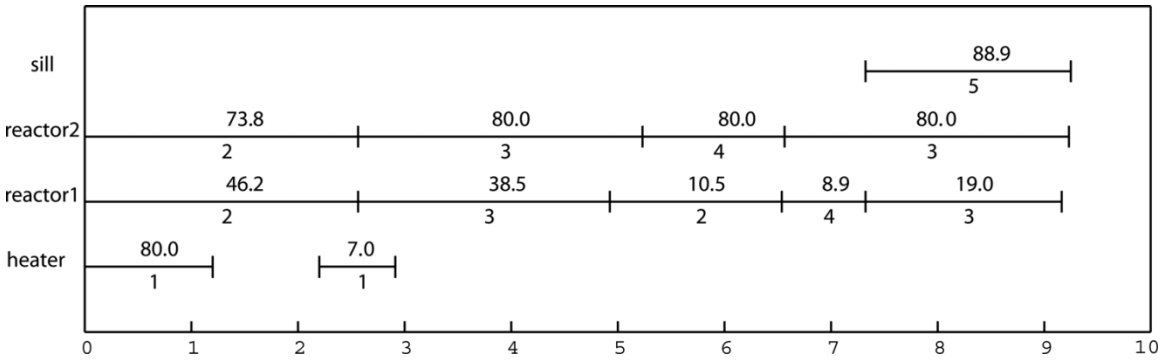


Figure 4.5 Reactive schedule for example1 with rush order at t = 2.2 h

b) Machine breakdown

In order to study the case when machine breakdown occurs, we consider the problem of maximizing the profit in a given makespan (8 hours) and there is no requirement on the demand of product P1 and P2. Note that these assumptions are not necessary and any other optimization objective and demand constraints can be used. The original schedule is obtained as shown in Figure 4.6 which results in a maximum profit of 1498.2. The we assume that reactor 2 breaks down during the scheduling execution period and the breakdown time and the time of maintenance are assumed uncertain as shown in Table 4.4.

Table 4.4 Machine breakdown uncertainty for example 1

	Value	Range
T^{break} (Breakdown time)	θ_1	$1 \le \theta_1 \le 7$
T^{maint} (Maintenance time)	θ_2	$0.5 \le \theta_2 \le 2.5$

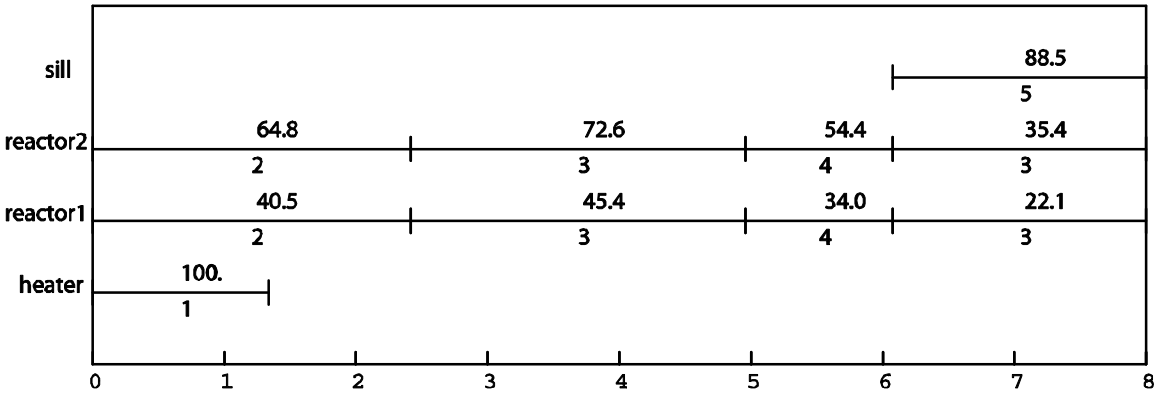


Figure 4.6 Original schedule of example 1, fixed H = 8 h

To illustrate the parametric solution, the parametric objective is shown in Figure 4.7 and the details of the solution are given in Table 4.5. The critical regions of the solution are shown in Figure 4.8. Computational data for the solution process is shown in Table 4.9. From the parametric solution, it can be observed that the final optimal profit will decrease with the increase of the maintenance time if the same unit breaks down at the same time. Also it should be noticed that the problem will become infeasible if the machine breakdown time and the maintenance time increase beyond certain limit.

Table 4.5 Parametric objective for example 1 with machine breakdown

Critical Region	Optimal profit	Critical Region	Optimal profit
1	920.5	9	576.2
2	$2015-363.2q_1-363.2q_2$	10	866.6
3	$1674.3-240q_1-240q_2$	11	$2466.6-240q_1-240q_2$
4	$1403.1-150q_1-150q_2$	12	1008.2
5	1058.9	13	$5001-706.7q_1-706.7q_2$
6	896.2	14	1356.6
7	$1856.2-240q_1-240q_2$	15	$2521.6-428.1q_1-428.1q_2$
8	$2176.2-240q_1-240q_2$		

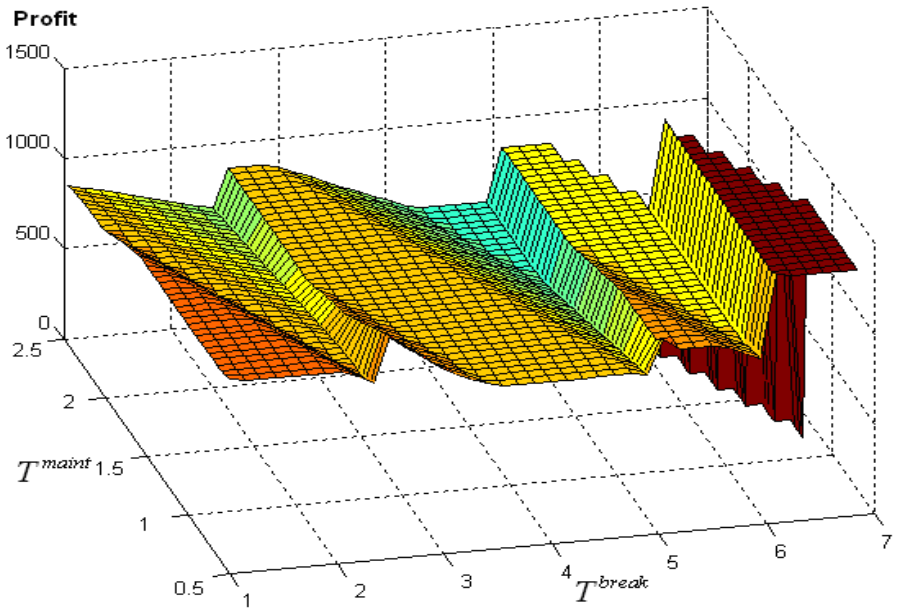


Figure 4.7 Parametric solution of maximum profit and machine breakdown parameter

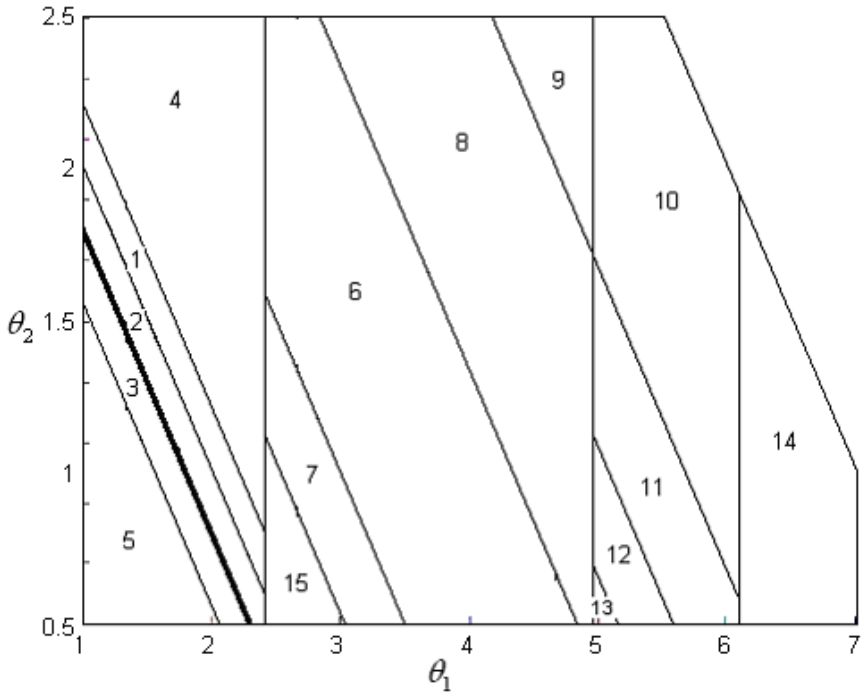


Figure 4.8 Critical region of the example 1 with machine breakdown

Once the parametric solution is obtained, the reactive schedule can be directly determined once the event occurs. For example, a reactive schedule for the reactor 2 breakdown at t=2.5 hour with 1 hour maintenance can be obtained by mapping the parameter value $(\theta_1, \theta_2) = (2.5,1)$ in the critical region 15. The corresponding

reactive schedule is shown in Figure 4.9. The resulted profit is $2521.6-428.1*2.5-428.1*1=1023.3$, which corresponds to a big decrease compared to the original profit (1498.2) because of the machine breakdown.

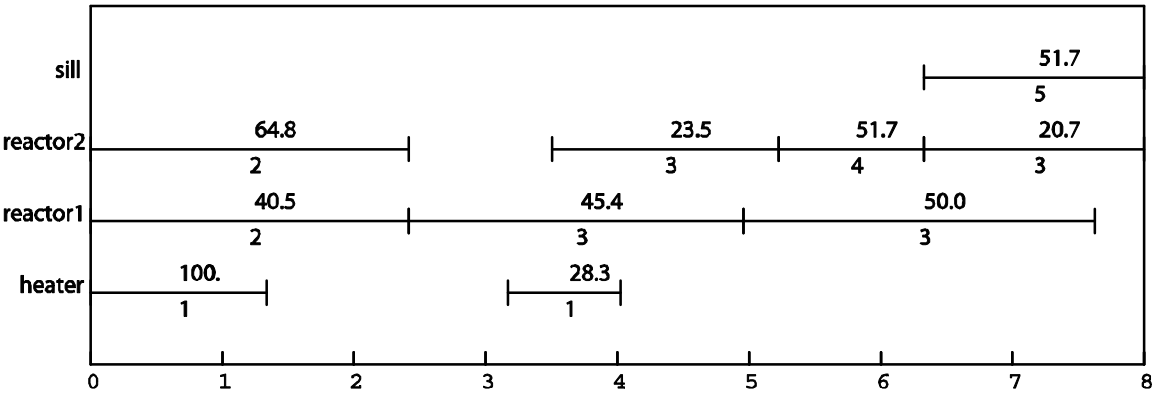


Figure 4.9 Reactive schedule for reactor 2 breakdown at t = 2.5 h, maintenance time = 1 h

Example 2

In example 2, four products are produced through eight tasks from three feeds and there are nine intermediates in the system. In all, six different units are required for the whole process. The STN representation of this process is shown in Figure 4.10 and the problem data can be found in Chapter 2. Through this example, we are studying the application of the proposed method on consecutive uncertainties for reactive scheduling. Specifically, we assume that the first disruptive event is a rush orders for products P1 and P2, and the second disruptive event is that unit 2 breaks down and needs maintenance.

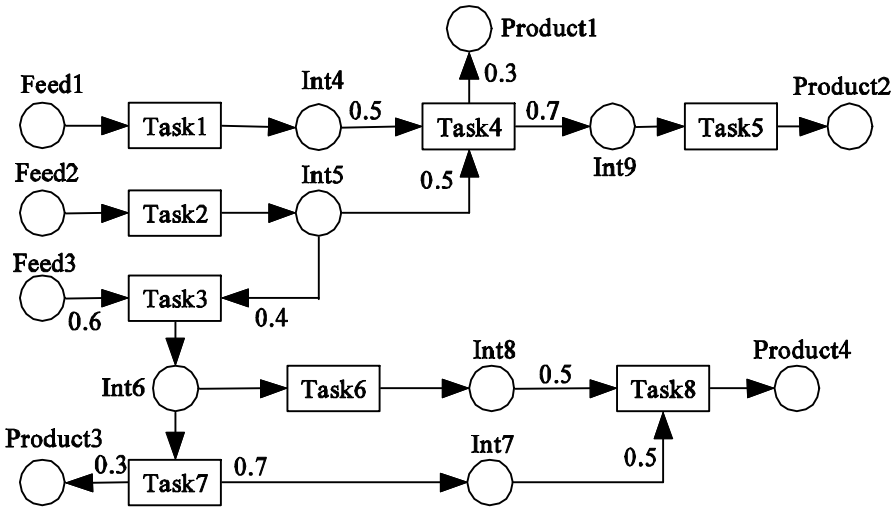


Figure 4.10 STN representation of example 2

First, we solve the deterministic scheduling problem with the objective of minimizing makespan to satisfy the nominal demand of products as: P1=600, P2=1400, P3=300, P4=1000. The original deterministic schedule is solved with 7 event points and the schedule is shown in Figure 4.11, which has a minimum makespan of 4.45 hours.

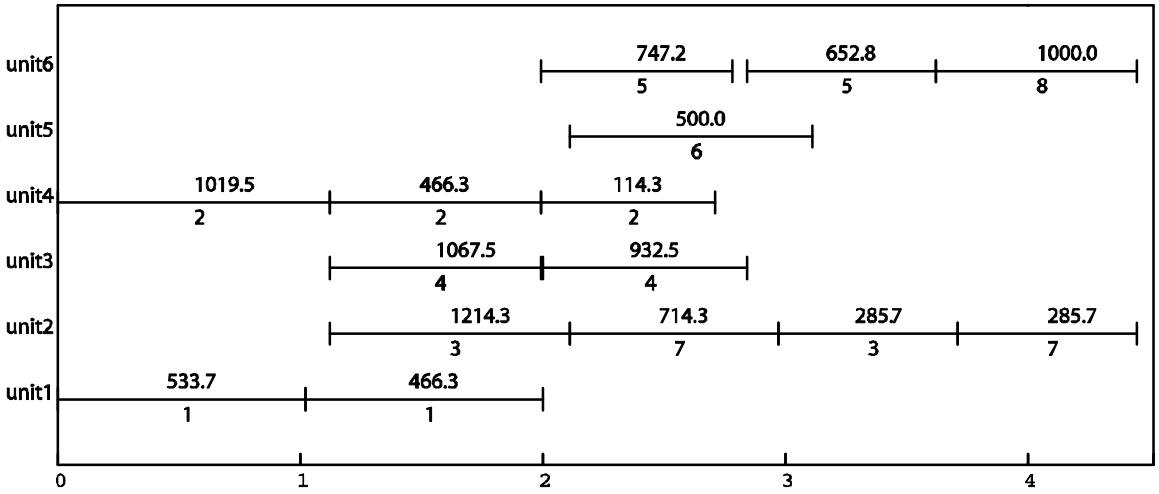


Figure 4.11 Original schedule for example 2

To address the upcoming rush order uncertainty, we can start solving the parametric problem for rush order soon after we get the original schedule. Using the maximum demand of the new order, 9 event points are identified to be necessary for the reactive scheduling formulation. The uncertain event is described as shown in Table 4.6. Then the multiparametric programming problem is solved with the critical regions illustrated in Figure 4.12.

Table 4.6 Rush order uncertainty for example 2

	Value	Range
r_{P1}^{rush} (New order of P1)	q_1	$500 \le P_1 \le 800$
r_{P2}^{rush} (New order of P2)	q_2	$1200 \le P_2 \le 1600$
T^{rush} (Order arrival time)	q_3	$1 \le q_2 \le 4$

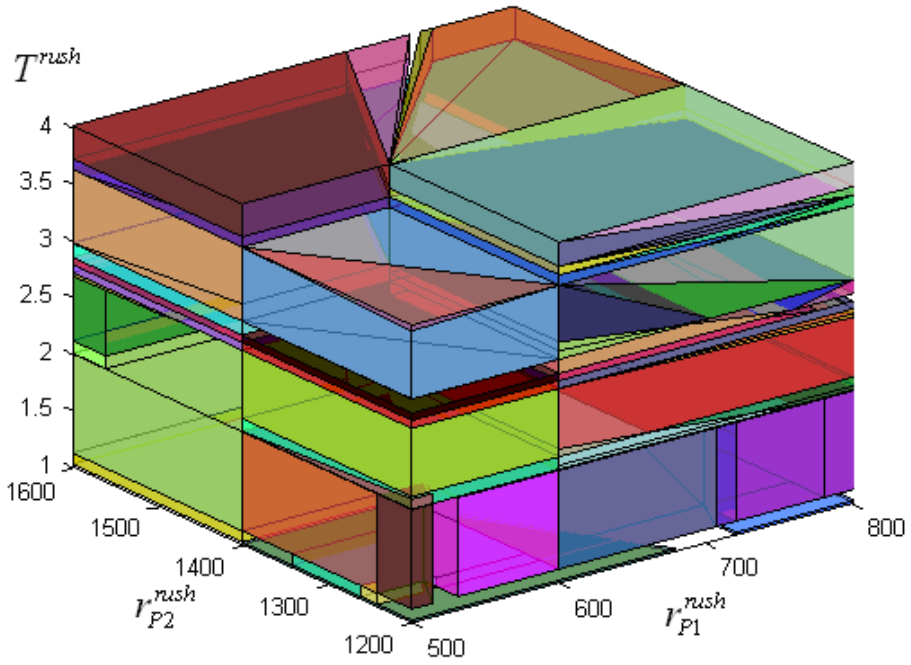


Figure 4.12 Critical region of example for rush order uncertainty

Having obtained the parametric solution, we can generate a reactive schedule as soon as the rush order arrives. For example, if the demand of product P1 increases to 750($q_1 = 750$), and the demand of P2 increases to 1500 ($q_2 = 1500$) at time t=1.5 hour ($q_3 = 1.5$), the parametric solution for $(q_1, q_2, q_3) = (750, 1500, 1.5)$ can be found directly from the parametric result. Figure 4.13 illustrates the new schedule which has a makespan of 4.74 hours.

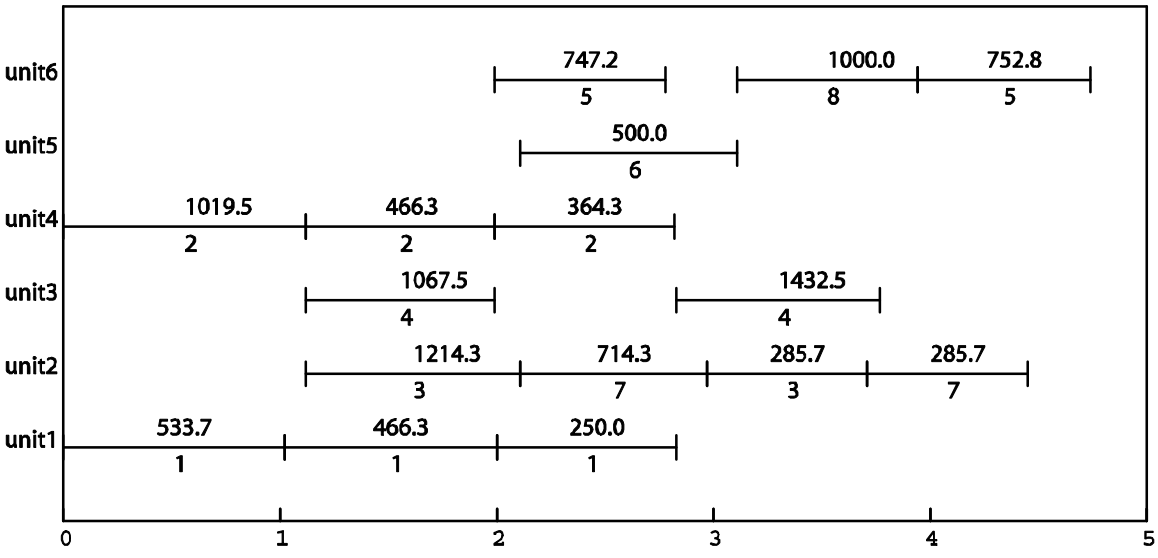


Figure 4.13 Reactive schedule for rush order at t = 1.5 h

Soon after the reactive schedule is executed, a new parametric reactive scheduling problem is solved to deal with future unexpected events. The machine breakdown uncertainty considered here is defined in Table 4.7. The critical regions of the parametric solution are shown in Figure 4.14, whereas the detail parametric objectives are shown in Table 4.8.

Table 4.7 Machine breakdown uncertainty for example 2

	Value	Range
T^{break} (Breakdown time)	θ_1	$3 \le \theta_1 \le 5$
T^{maint} (Maintenance time)	θ_2	$0.5 \le \theta_2 \le 2.5$

Table 4.8 Parametric objective for example 2 with machine breakdown

Critical Region	Makespan H (hours)
1	4.74
2	$3.74 + 0.062\theta_1 + 0.062\theta_2$
3	$1.49 + \theta_1 + \theta_2$
4	$0.74 + \theta_1 + \theta_2$
5	$2.37 + \theta_1 + \theta_2$
6	$4.102 + 0.0656\theta_1$
7,8	$2.48 + \theta_1 + \theta_2$

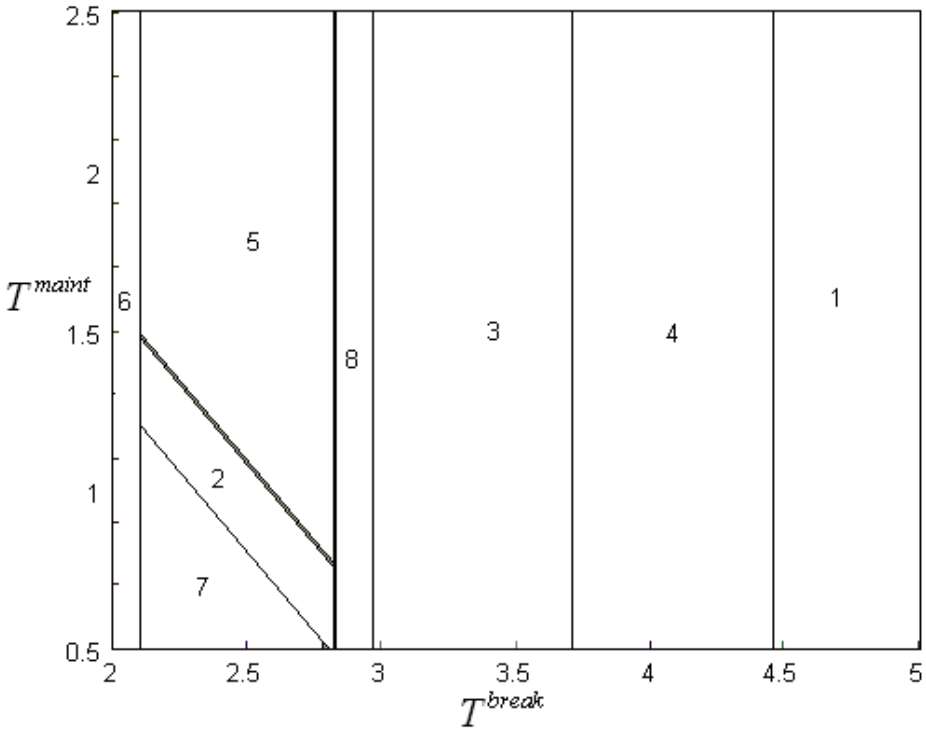


Figure 4.14 Critical region of example 2 with machine breakdown.

After we obtain the parametric solution, we can address the upcoming machine breakdown. For example, if unit 2 breaks down at $T^{break} = 3$ hour, and requires $T^{maint} = 1.5$ hour, it corresponds to $(q_1, q_2) = (3, 1.5)$, so the reactive schedule can be obtained from the parametric result and it is shown in Figure 4.15.

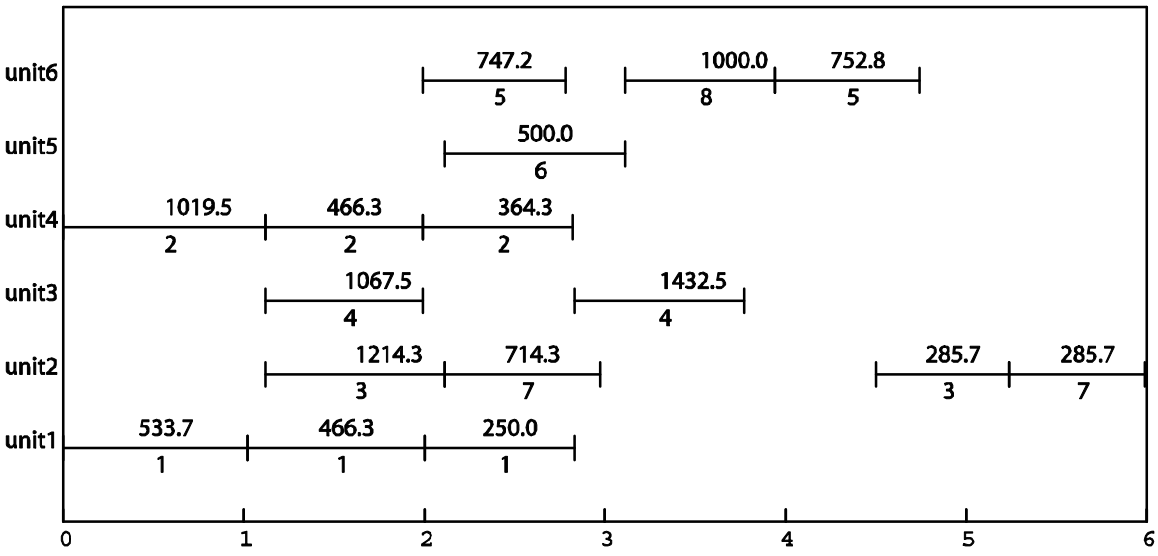


Figure 4.15 Reactive schedule for unit 2 breakdown at t = 3 h, maintenance time = 1.5 h

Following this dynamic way of addressing uncertainty, multiple disruptive uncertainties in the scheduling process can be addressed. The only requirement is that upon the arrival of a new disruptive event, the corresponding parametric solution that covers this uncertain event has been retrieved. In this example, the first parametric problem is solved in 2300 CPU sec, and the second parametric problem is solved in 1442 CPU sec. In both cases we test 5000 points uniformly distributed in the uncertain space and the parametric solution can cover all the given uncertain space (Figures 4.12, 4.14) except the infeasible operation areas. Detail computational statistics are given in Table 4.9. Furthermore, during the process of solving the parametric programming problem, the uncertain space that represents the near future can be solved at the beginning so that the earlier disruptive events can be covered by the parametric solution.

Table 4.9 Computational statistics for the examples

	Example 1		Example 2	
	Rush order	Machine breakdown	Rush order	Machine breakdown
Reactive scheduling model				
Constraints	1360	1596	2278	2358
Continuous	576	673	1316	1370
Binaries	280	320	864	864
Number of testing points	5000	5000	5000	5000
Number of critical regions*	35	49	89	18

Average iterations per point	3	3	3	2
Average CPU time per point (sec)**	7	5	6	5

* Equal to the number of points used to apply the multiparametric programming, since some of these critical regions might belong to a larger nonconvex critical region, the final critical region illustrated in the chapter is post-processed result after union operation.

** The average time is for multiparametric programming solution process for a point

4.4 Summary

A new methodology for reactive scheduling is proposed in this chapter. Different to any existing method, this chapter provides a direct mapping approach to generate the reactive schedule with the parametric solution. It greatly improves the efficiency of reactive scheduling because the reactive schedule is obtained by checking from a set of parametric solutions which is solved ahead of time but not solve a rescheduling problem after the uncertainty occurs. The proposed methodology is designed to address single disruptive event. However, consecutive uncertainties can be addressed through the repetitive application of the method. It is worthwhile to note here that the number of critical regions of multiparametric MILP problem increases with the size of the uncertain space (number of the uncertain parameters), so complete coverage of the uncertain space needs considerable computational effort. However, the parametric solution generated using the proposed method provides a way to derive the possible reactive decision with existing computational ability before the uncertain event, which make it possible to save time in making reactive decision. Once the realized uncertainty is not covered by the current solution, the reactive schedule can be directly solved through the developed reactive scheduling formulation. Further improvements on the proposed method lie on developing parallel algorithm to solve the multiparametric programming problem to further save the computation time.

Chapter 5

Integration of Planning and Scheduling

Abstract: In this chapter, augmented Lagrangian method is applied to solve the full-space integration problem which takes a block angular structure. To resolve the non-separability issue in the augmented Lagrangian relaxation, we study the traditional method which approximates the cross-product term through linearization and also propose a new decomposition strategy based on two-level optimization. The results from case study show that the augmented Lagrangian method is effective in solving the large integration problem and generating a feasible solution. Furthermore, the proposed decomposition strategy based on two-level optimization can get better feasible solution than the traditional linearization method.

5.1 Introduction

Production planning and scheduling belong to different decision making levels in process operations, they are also closely related since the result of planning problem is the production target of scheduling problem. In process industry, the commonly used planning and scheduling decision making strategy generally follows a hierarchical approach, in which the planning problem is solved first to define the production targets and the scheduling problem is solved next to meet these targets. However, there exists a big disadvantage in this traditional strategy since there is no interaction between the two decision levels, i.e., the planning decisions generated might cause infeasible scheduling subproblems. At the planning level, the effects of changeovers and daily inventories are neglected, which tends to produce optimistic estimates that can not be realized at the scheduling level, i.e., a solution determined at the planning level does not necessarily lead to feasible schedules. Moreover, the optimality of the planning solution cannot be ensured because the planning level problem might not provide an accurate estimation of the production cost, which should be calculated from detailed tasks determined by the scheduling problem.

Therefore, it is important and necessary to develop methodologies that can effectively integrate production planning and scheduling. However, since production planning and scheduling are dealing with different time scales, the major challenge for the integration lies in the large problem size of the resulted optimization model. A direct way for addressing the integrated planning and scheduling problems is to formulate a single simultaneous planning and scheduling model that spans the entire planning horizon of interest. However, when typical planning horizons are considered, the size of this detailed model becomes intractable, because of the potential exponential increase in the computation time. To overcome the above difficulty, most of the work appeared in the literature aim at decreasing the problem scale through different types of problem reduction methodologies and developing efficient solution strategies as summarized by (Grossmann et al., 2002), (Maravelias & Sung, 2008). Generally, the existing work in the area of planning and scheduling integration can be summarized as follows.

The first type of methods is based on decomposition in a hierarchical way through iterative solution procedure. Through a hierarchical decomposition of the integration problem, detailed scheduling constraints are not incorporated into the upper level aggregate planning model, on the other hand, information is passed from the aggregate planning problem to a set of detailed scheduling problems and these scheduling problems are separated based on the temporal decomposition. Thus, the problems that need to be solved include a relative simple planning problem and a series of scheduling subproblems. To ensure the feasibility and optimality of the solution, it is further necessary to develop effective algorithms to improve the solution using additional cuts in the planning level within an iterative solution framework (Papageorgiou & Pantelides, 1996); (Bassett, Pekny et al., 1996); (Munawar & Gudi, 2005); (Erdirik-Dogan & Grossmann, 2006). The second type of method, which is also called rolling horizon approach, considers a relative rough model for the far future planning periods in the integrated planning and scheduling model, i.e., detailed scheduling models are only used for a few early periods and aggregate models are used for later periods. The production targets for the early periods are directly implemented, while the production targets for the later periods are updated along with the rolling horizon. Applications of this kind of strategy can be found in (Dimitriadis et al., 1997); (Sand et al., 2000); (Wu & Ierapetritou, 2007); (Verderame & Floudas, 2008). Thirdly, for the cases where there is no plant and market variability, campaign mode can be applied to generate an easy to implement and profitable process operations plan. In a periodic scheduling framework, the planning and scheduling

integration problem is replaced by establishing an operation schedule and executing it repeatedly (Zhu & Majozi, 2001); (Castro et al., 2003); (Wu & Ierapetritou, 2004). Other than using the detailed scheduling model in the integrated planning, surrogate methods aim at deriving the scheduling feasibility and production cost function first and then incorporating them into the integrated problem. This avoids the disadvantage of large scale and complex model which directly incorporate the detailed scheduling model into aggregating planning model as shown in (Sung & Maravelias, 2007).

Except the different methods for the integrated planning and scheduling summarized above, another approach is based on the study of the special structure of the mathematical programming model for the integration problem and aims at developing efficient decomposition techniques to solve the optimization problem directly. Lagrangian relaxation is an approach that is often applied to models with a block angular structure. In such models, distinct blocks of variables and constraints can be identified and they are linked through a few "linking" constraints and variables. To our knowledge, Lagrangian relaxation has been widely applied onto planning and scheduling problems for different applications including unit commitment in power industry (Padhy, 2004), midterm production planning (Gupta & Maranas, 1999), and combined transportation and scheduling (Equi et al., 1997), etc. However, the major drawback of Lagrangian relaxation method is that there is duality gap between the solution of the Lagrangian dual problem and the solution of original problem, and often the feasibility of the solution needs to be recovered through heuristic steps. So it is often only used as the bounding step in the branch and bound framework. The disadvantage of Lagrangian relaxation can be avoided by augmented Lagrangian relaxation (ALR) method, which has been used in several applications in areas such as power generation scheduling (Carpentier et al., 1996), multidisciplinary design (Tosserams et al., 2008), etc. One drawback of ALR method is the non-separability of the relaxed problem, which has also received wide attention in the literature. In this chapter, we propose to apply the ALR method on the planning and scheduling integration problem which takes a block angular model structure, and also propose a new decomposition strategy to address the non-separability issue in the ALR solution procedure, which can be used to decompose the relaxed problem exactly without any approximation technique as presented in the literature.

The content of this chapter is organized as follows. The problem formulation of the integrated planning and scheduling problem is first presented in section 5.2. The general augmented Lagrangian solution method

is presented in section 5.3. Detail reformulation and decomposition strategies for the planning and scheduling integration problem are presented in section 5.4. The proposed method is studied in section 5.5 through a case study and the chapter concludes in section 5.6.

5.2 Problem structure

Production planning model is used to predict production targets and material flow over several months (up to one year), it is generally takes a simplified representation of the production and formulated as linear problem. Scheduling models on the other hand are more detailed assuming that key decisions (production targets) have been made. To integrate these two different decision-making problems, the simplest way is to formulate a full space optimization model, where in every period of the planning horizon, the scheduling constraints are incorporated into the model, while keeping the inventory connecting constraints between the planning decision and scheduling decisions. In this work, we formulate the production planning and scheduling integration problem as follows.

$$\min \quad \sum_t \sum_{s\in S_P} h_s Inv_s^t + \sum_t \sum_{s\in S_P} u_s U_s^t + \sum_t \sum_i \sum_j \sum_n (FixCost_i w_{ijn}^t + VarCost_i b_{ijn}^t) \tag{5.1a}$$

s.t.

$$Inv_s^t = Inv_s^{t-1} + P_s^t - D_s^t \qquad \forall s \in S_P, \forall t \tag{5.1b}$$

$$U_s^t = U_s^{t-1} + Dem_s^t - D_s^t \qquad \forall s \in S_P, \forall t \tag{5.1c}$$

$$st_{s,n=N}^t - stin_s^t = P_s^t \qquad \forall s \in S_P, \forall t \tag{5.1d}$$

$$stin_s^t = Inv_s^{t-1} \qquad \forall s \in S_P, \forall t \tag{5.1e}$$

$$\sum_{i\in I_j} wv_{i,j,n}^t \le 1 \qquad \forall j \in J, \forall n \in N, \forall t \tag{5.1f}$$

$$v_{i,j}^{\min} wv_{i,j,n}^t \le b_{i,j,n}^t \le v_{i,j}^{\max} wv_{i,j,n}^t \qquad \forall i \in I, \forall j \in J_i, \forall n \in N, \forall t \tag{5.1g}$$

$$Tf_{i,j,n}^t = Ts_{i,j,n}^t + \alpha_{i,j} wv_{i,j,n}^t + \beta_{i,j} b_{i,j,n}^t \qquad \forall i \in I, \forall j \in J_i, \forall n \in N, \forall t \tag{5.1h}$$

$$Ts_{i,j,n+1}^t \ge Tf_{i,j,n}^t - H(1 - wv_{i,j,n}^t) \qquad \forall i \in I, \forall j \in J_i, \forall n \in N, \forall t \tag{5.1i}$$

$$Ts_{i,j,n+1}^t \ge Tf_{i',j,n}^t - H(1 - wv_{i',j,n}^t) \qquad \forall i, i' \in I_j, \forall j \in J, \forall n \in N, \forall t \tag{5.1j}$$

$$Ts_{i,j,n+1}^t \ge Tf_{i',j',n}^t - H(1 - wv_{i',j',n}^t) \qquad \forall i, i' \in I_j, i \ne i' \ \forall j, j' \in J, \forall n \in N, \forall t \tag{5.1k}$$

$$Ts^t_{i,j,n+1} \geq Ts^t_{i,j,n} \qquad \forall i \in I, \forall j \in J_i, \forall n \in N, \forall t \qquad (5.1l)$$

$$Tf^t_{i,j,n+1} \geq Tf^t_{i,j,n} \qquad \forall i \in I, \forall j \in J_i, \forall n \in N, \forall t \qquad (5.1m)$$

$$Ts^t_{i,j,n} \leq H \qquad \forall i \in I, \forall j \in J_i, \forall n \in N, \forall t \qquad (5.1n)$$

$$Tf^t_{i,j,n} \leq H \qquad \forall i \in I, \forall j \in J_i, \forall n \in N, \forall t \qquad (5.1o)$$

$$st^t_{s,n} = st^t_{s,n-1} - \sum_{i \in I_s} \rho^C_{s,i} \sum_{j \in J_i} b^t_{i,j,n} + \sum_{i \in I_s} \rho^P_{s,i} \sum_{j \in J_i} b^t_{i,j,n-1} \quad \forall s \in S, \forall n \in N, \forall t \qquad (5.1p)$$

$$st^t_{s,n=1} = stin^t_s - \sum_{i \in I_s} \rho^C_{s,i} \sum_{j \in J_i} b^t_{i,j,n=1} \qquad \forall s \in S, \forall t \qquad (5.1q)$$

$$st^t_{s,n} \leq st^{\max}_s \qquad \forall s \in S, \forall n \in N \qquad (5.1r)$$

In the above model, the objective function (5.1a) is the total cost composed by three parts: inventory cost, backorder cost and production cost, where the inventory cost and backorder cost are calculated based on the inventory and backorder amount and the given unit cost parameter (h_s , u_s); the production cost of different planning periods is composed by a fixed part which represents the basic cost of a task, and a dynamic part which is proportional to the amount of material processed (batch size).

The constraints of the above integration model can be divided into planning level and scheduling level. Equations (5.1b) and (5.1c) represent the planning level constraints, among them, equations (5.1b) represent the inventory balance and equations (5.1c) represent the backorder balance. Among the constraints of the scheduling level, equations (5.1d) express the requirement that the planning solutions P^t_s generated from upper planning level is the production targets for different planning periods. Equations (5.1e) represent the connection constraints for the initial product inventory for different planning periods. Equations (5.1f)-(5.1r) represent scheduling constraints which can be referred from Chapter 2. If we denote the scheduling decision variables ($wv^t_{i,j,n}, b^t_{i,j,n}, Tf^t_{i,j,n}, Ts^t_{i,j,n}, st^t_{s,n}, stin^t_s$) for planning period t using the vector y^t , then the structure of the above integrated planning and scheduling model can be illustrated as shown in Figure 5.1.

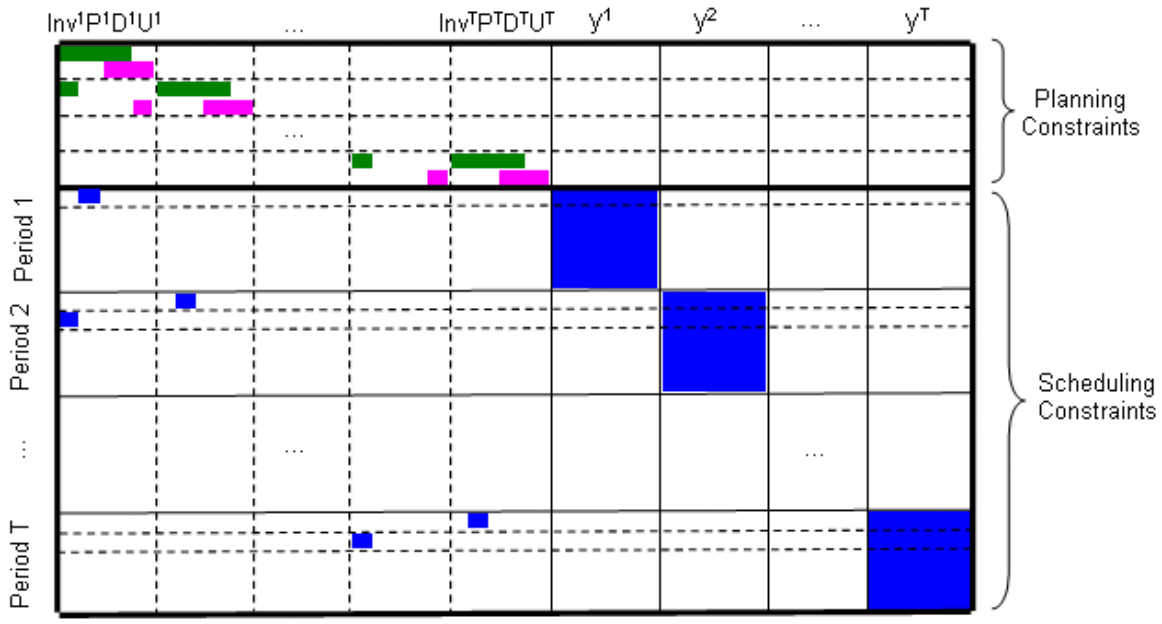


Figure 5.1 Constraint matrix structure of the integration model

In the above constraint matrix, the part on the top of the matrix corresponds to the planning constraints, and the lower part is composed by scheduling constraints for different planning periods. It can be observed that the integration model takes a block angular structure and the blocks are linked through planning decision variables. As stated in the introduction section, Lagrangian relaxation is a typical approach that is often applied to this type of models with a block angular structure. However, to avoid the drawback of classical Lagrangian relaxation, augmented Lagrangian method is applied in this work.

5.3 Augmented Lagrangian Optimization algorithm

Observing the special constraint structure of the integrated planning and scheduling problem as shown in Figure 5.1, we can reformulate the problem into a decomposable structure through the introduction of auxiliary duplicate variables PP_s^t for the production target P_s^t, and II_s^t for the inventory variables Inv_s^t, respectively. The following reformulated problem which is equivalent to the original problem (5.1) can be derived:

$$\min \quad \sum_t \sum_{s\in S_P} h_s Inv_s^t + \sum_t \sum_{s\in S_P} u_s U_s^t + \sum_t \sum_i \sum_j \sum_n (FixCost_i w_{ijn}^t + VarCost_i b_{ijn}^t) \tag{5.2a}$$

$$s.t. \quad Inv_s^t = Inv_s^{t-1} + P_s^t - D_s^t \qquad \forall s \in S_P, \forall t \tag{5.2b}$$

$$U_s^t = U_s^{t-1} + Dem_s^t - D_s^t \qquad \forall s \in S_P, \forall t \qquad (5.2c)$$

$$P_s^t = PP_s^t \qquad \forall s \in S_P, \forall t \qquad (5.2d)$$

$$Inv_s^t = II_s^t \qquad \forall s \in S_P, \forall t \qquad (5.2e)$$

$$st_{s,n=N}^t - stin_s^t = PP_s^t \qquad \forall s \in S_P, \forall t \qquad (5.2f)$$

$$stin_s^t = II_s^{t-1} \qquad \forall s \in S, \forall t \qquad (5.2g)$$

$$y^t \in Y \qquad \forall t \qquad (5.2h)$$

In the above model, (5.2d) and (5.2e) are the coupling constraints which link the different scheduling and planning constraints block. In the problem reformulation (5.2) we have made a compact representation of the scheduling constraints (5.1f)-(5.1r) as (5.2h) for the sake of simplicity. Thus the constraint matrix structure of the above problem is shown as Figure 5.2.

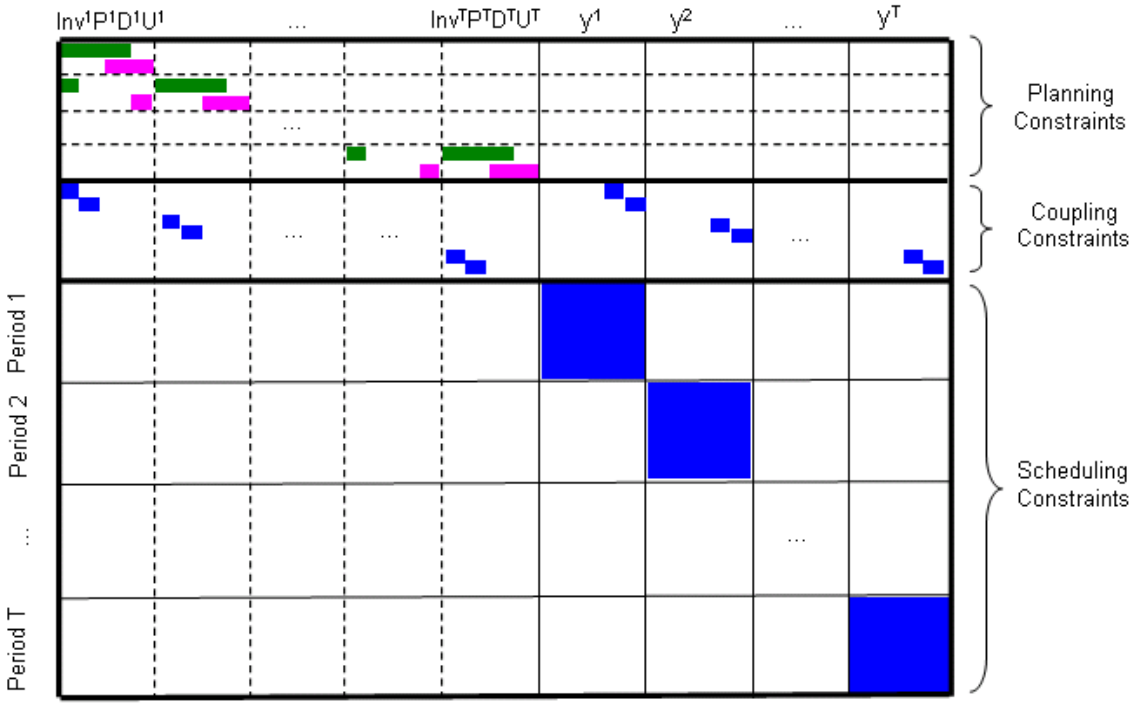


Figure 5.2 Constraint matrix structure of the reformulated model

With the above reformulation, the resulted model (5.2) is decomposed into a planning subproblem and a number of scheduling subproblems once the coupling constraints (5.2d) and (5.2e) are relaxed. In this work, the augmented Lagrangian algorithm is applied to solve the integration planning and scheduling problem.

Specifically, equality constraints (5.2d) and (5.2e) are relaxed and the following augmented Lagrangian relaxation problem is obtained:

$$f(\lambda,\mu,\sigma) =$$

$$\begin{aligned}\min \sum_t \sum_{s\in S_P} h_s Inv_s^t + \sum_t \sum_{s\in S_P} u_s U_s^t + \sum_t \sum_i \sum_j \sum_n (FixCost_i w_{ijn}^t + VarCost_i b_{ijn}^t) \\ + \sum_t \sum_{s\in S_P} \lambda_s^t (P_s^t - PP_s^t) + \sum_t \sum_{s\in S_P} \mu_s^t (Inv_s^t - II_s^t) + \sigma \sum_t \sum_{s\in S_P} \{(P_s^t - PP_s^t)^2 + (Inv_s^t - II_s^t)^2\}\end{aligned} \quad (5.3a)$$

$$s.t. \quad Inv_s^t = Inv_s^{t-1} + P_s^t - D_s^t \qquad \forall s \in S_P, \forall t \quad (5.3b)$$

$$U_s^t = U_s^{t-1} + Dem_s^t - D_s^t \qquad \forall s \in S_P, \forall t \quad (5.3c)$$

$$st_{s,n=N}^t - stin_s^t = PP_s^t \qquad \forall s \in S_P, \forall t \quad (5.3d)$$

$$stin_s^t = II_s^{t-1} \qquad \forall s \in S_P, \forall t \quad (5.3e)$$

$$y^t \in Y \qquad \forall t \quad (5.3f)$$

Thus the solution of the original planning and scheduling integration problem (5.1) is transformed into the solution of the following augmented Lagrangian dual problem $\max_{\lambda,\mu,\sigma} f(\lambda,\mu,\sigma)$. In particular, we propose the following algorithm for the planning and scheduling integration problem:

Step 1. Initialization. Set bounds for multipliers: $[\lambda_{\min}, \lambda_{\max}]$, $[\mu_{\min}, \mu_{\max}]$. Choose initial multiplier and penalty parameter value $\lambda_s^t = 0$, $\mu_s^t = 0$, $\sigma = 1$, set $k = 1$, $\varepsilon > 0$ (e.g., 0.1), $\alpha > 1$ (e.g., 2.2), $\beta \in (0,1)$ (e.g., 0.4);

Step 2. Compute an approximate solution of the augmented Lagrangian relaxation problem through decomposition technique as described in detail in the next section, get solution Inv, II, P, PP and objective value $f(\lambda,\mu,\sigma)$. Define consistency function value vector $g = \left[Inv - II \quad P - PP\right]^T$, if $\| g \| < \varepsilon$, then stop, (λ,μ,σ) is a solution; otherwise, go to step 3.

Step 3. Update multipliers:

$\lambda_s^t = \min\{\max\{\lambda_{\min}, \lambda_s^t + \sigma(Inv_s^t - II_s^t)\}, \lambda_{\max}\}$, $\mu_s^t = \min\{\max\{\mu_{\min}, \mu_s^t + \sigma(P_s^t - PP_s^t)\}, \mu_{\max}\}$.

If $\| g \|_{(k)} \geq \beta \| g \|_{(k-1)}$, set $\sigma = \alpha\sigma$; otherwise keep σ unchanged. Set $k = k + 1$, go to step 2.

Although the convergence properties of the ALR algorithm proved by (Andreani et al., 2008) (Appendix B) are based on the assumption of the continuous first derivative of the objective function $f(x)$ and upper level constraints $h(x), g(x)$, it is worth to point out that these properties are remained for the mixed integer linear programming problem studied in this paper. The reason is that the mixed integer problem is always able to be transformed into its equivalent continuous counterpart because binary variable $wv_{i,j,n}^{t} \in \{0,1\}$ can be replaced by continuous relaxation $0 \le wv_{i,j,n}^{t} \le 1$ and adding complementarity constraints $wv_{i,j,n}^{t}(1 - wv_{i,j,n}^{t}) = 0$, so when the above algorithm is applied onto the mixed integer programming problem (5.2), similar convergence properties can still be ensured.

Note that in the above solution algorithm, it is necessary to solve a series of augmented Lagrangian relaxation problems (5.3). However, the objective function of the relaxation problem (5.3) contains cross product terms $P_s^t PP_s^t$ and $Inv_s^t I_s^t$ which are non-separable, thus it is still hard to solve the relaxation problem unless it is decomposed because it is almost as hard as the original problem (5.1). So, in next subsection several decomposition strategies are presented to decompose the relaxation problem and reduce the computational complexity.

5.4 Decomposition strategy

As presented above, in the augmented Lagrangian solution framework, there is an upper level which aims at finding the optimal Lagrangian multipliers and penalty parameters to solve the augmented Lagrangian dual problem. In every iteration of the method of multipliers, an augmented Lagrangian relaxation problem (5.3) needs to be solved with fixed Lagrangian multipliers λ, μ and penalty parameter σ. The relaxation problem is solved in a lower level using different decomposition strategies. There are several techniques in the literature that resolve the issue of separability in the augmented Lagrangian solution method: the Diagonal Quadratic Approximation (DQA) method (Ruszczynski, 1995); the Block Coordinate Decent (BCD) method which is also known as the "nonlinear Gauss-Seidel" method (Bertsekas, 2003); the Alternating Direction method (Bertsekas & Tsitsiklis, 1989), which is an extreme case of the BCD method by taking only a single

BCD iteration; the separable augmented Lagrangian algorithm (Hamdi et al., 1997), etc. All of those methods generate an approximate decomposable version of the original relaxation problem then solve it through decomposition. In this subsection, we present the Diagonal Quadratic Approximation method for comparison and also propose a new method based on two-level optimization of the relaxation problem.

5.4.1 Diagonal Quadratic Approximation

Diagonal Quadratic Approximation method addresses the nonseparable issue through linearizing the cross product quadratic term ($P_s^t PP_s^t$, $Inv_s^t II_s^t$) around the tentative solution $\overline{P_s^t}, \overline{PP_s^t}, \overline{Inv_s^t}, \overline{II_s^t}$ and get separable approximation (also called) as following

$$(P_s^t - PP_s^t)^2 \approx (P_s^t - \overline{PP_s^t})^2 + (PP_s^t - \overline{P_s^t})^2 - (\overline{P_s^t} - \overline{PP_s^t})^2$$

Thus with above substitution for the nonseparable term, the original relaxed problem (5.3) can be rewritten as the following decomposable form

$$f(\lambda, \mu, \sigma) = f_P + \sum_t f_S^t$$

where f_P represents optimal objective of the following planning subproblem (5.4)

$$\begin{aligned} f_P = \min_{P,Inv,D,U} \; & \sum_t \sum_s h_s Inv_s^t + \sum_t \sum_s u_s U_s^t \\ & + \sum_s \sum_t \lambda_s^t P_s^t + \sigma \sum_s \sum_t \{(P_s^t - \overline{PP_s^t})^2 - (\overline{P_s^t} - \overline{PP_s^t})^2\} \\ & + \sum_s \sum_t \mu_s^t Inv_s^t + \sigma \sum_s \sum_t \{(Inv_s^t - \overline{II_s^t})^2 - (\overline{Inv_s^t} - \overline{II_s^t})^2\} \end{aligned} \tag{5.4a}$$

$$s.t. \quad Inv_s^t = Inv_s^{t-1} + P_s^t - D_s^t \qquad \forall s \in S_P, \forall t \tag{5.4b}$$

$$U_s^t = U_s^{t-1} + Dem_s^t - D_s^t \qquad \forall s \in S_P, \forall t \tag{5.4c}$$

and f_S^t represent optimal objectives of the following scheduling subproblems (5.5)

$$\begin{aligned} f_S^t = \min_{PP^t, II^t, y^t} \; & \sum_i \sum_j \sum_n (FixCost_i w_{ijn}^t + VarCost_i b_{ijn}^t) - \sum_s \lambda_s^t PP_s^t + \sigma \sum_s (PP_s^t - \overline{P_s^t})^2 \\ & - \sum_s \mu_s^t II_s^t + \sigma \sum_s (II_s^t - \overline{Inv_s^t})^2 \end{aligned} \tag{5.5a}$$

$$s.t. \quad st_{s,n=N}^t - stin_s^t = PP_s^t \qquad \forall s \in S_P \tag{5.5b}$$

$$stin_s^t = II_s^{t-1} \qquad \forall s \in S_P \qquad (5.5c)$$

$$y^t \in Y \qquad (5.5d)$$

In the DQA solution method, subproblem (5.4) and subproblems (5.5) are solved alternately with updated value of the tentative solution until a given iteration limit is reached or the relative change in the objective function value of the relaxation problem for two consecutive inner loop iterations is smaller than some user-defined termination tolerance. Sometimes, considering the fact that high accuracy of the subproblem solutions is not necessary in the early iterations when the Lagrangian multipliers are far from its optimal value and the computational effort is wasted, it is more desirable to quickly update the Lagrangian multiplier to move toward its optimal value. This can be achieved by limiting the total number of inner loop iterations in DQA by treating it as user-specified parameter to reduce the computational cost for solving the inner loop (Li et al., 2008).

Among the above subproblems, the planning subproblem is a quadratic programming problem, whereas the scheduling subproblems are mixed integer quadratic programming problems, all of them can be solved through standard QP/MIQP solvers such as CPLEX 10. Also notice that the feasibility of the subproblems can be ensured since the auxiliary variables are not constrained in the subproblem. Furthermore, an important fact regarding those subproblems is that they can be solved in parallel, thus the solution efficiency can be greatly improved.

5.4.2 Two-Level optimization

Different from those methods that use an approximation to make the objective separable, we propose a new method to address the non-separability issue in the augmented Lagrangian method. First, the augmented relaxation problem (5.3) can be rewritten to the following equivalent form:

$f(\lambda, \mu, \sigma) =$

$$\min_{P,Inv,D,U} \sum_t \sum_{s\in S_P} h_s Inv_s^t + \sum_t \sum_{s\in S_P} u_s U_s^t + \left\{ \begin{array}{ll} \min\limits_{y,II,PP} & \sum\limits_t \sum\limits_i \sum\limits_j \sum\limits_n (FixCost_i w_{ijn}^t + VarCost_i b_{ijn}^t) \\ & + \sum\limits_{s\in S_P} \sum\limits_t \lambda_s^t (P_s^t - PP_s^t) + \sigma \sum\limits_{s\in S_P} \sum\limits_t (P_s^t - PP_s^t)^2 \\ & + \sum\limits_{s\in S_P} \sum\limits_t \mu_s^t (Inv_s^t - II_s^t) + \sigma \sum\limits_{s\in S_P} \sum\limits_t (Inv_s^t - II_s^t)^2 \\ \text{s.t.} & st_{s,n=N}^t - stin_s^t = PP_s^t \qquad \forall s \in S_P, \forall t \\ & stin_s^t = II_s^{t-1} \qquad \forall s \in S_P, \forall t \\ & y^t \in Y \qquad \forall t \end{array} \right\} \tag{5.6a}$$

$$\text{s.t.} \quad Inv_s^t = Inv_s^{t-1} + P_s^t - D_s^t \qquad \forall s \in S_P, \forall t \tag{5.6b}$$

$$U_s^t = U_s^{t-1} + Dem_s^t - D_s^t \qquad \forall s \in S_P, \forall t \tag{5.6c}$$

Problem (6) can be simplified as follows:

$$f(\lambda, \mu, \sigma) =$$

$$\min_{I,P,D,U} \sum_t \sum_{s\in S_P} h_s Inv_s^t + \sum_t \sum_{s\in S_P} u_s U_s^t + \sum_t q^t(P, Inv) \tag{5.7a}$$

$$\text{s.t.} \quad Inv_s^t = Inv_s^{t-1} + P_s^t - D_s^t \qquad \forall s \in S_P, \forall t \tag{5.7b}$$

$$U_s^t = U_s^{t-1} + Dem_s^t - D_s^t \qquad \forall s \in S_P, \forall t \tag{5.7c}$$

where $q^t(P, Inv)$ is further defined by the following optimization subproblems:

$$q^t(P, Inv) =$$

$$\begin{aligned} \min_{y,II,PP} \quad & \sum_i \sum_j \sum_n (FixCost_i w_{ijn}^t + VarCost_i b_{ijn}^t) \\ & + \sum_{s\in S_P} \lambda_s^t (P_s^t - PP_s^t) + \sum_{s\in S_P} \mu_s^t (Inv_s^t - II_s^t) + \sigma \sum_{s\in S_P} \{(P_s^t - PP_s^t)^2 + (Inv_s^t - II_s^t)^2\} \end{aligned} \tag{5.8a}$$

$$\text{s.t.} \quad st_{s,n=N}^t - stin_s^t = PP_s^t \qquad \forall s \in S_P \tag{5.8b}$$

$$stin_s^t = II_s^{t-1} \qquad \forall s \in S_P \tag{5.8c}$$

$$y^t \in Y \tag{5.8d}$$

With the above reformulation strategy, the solution of the relaxation problem (5.3) can be transformed into the solution of nonlinear problem (5.7) which takes an implicit objective function and the evaluation of the objective function needs the solution of a series of subproblems (5.8).

The difference between the DQA strategy and the proposed two-level strategy lies on the fact that the DQA strategy actually solves an approximation version of the relaxation problem (5.3). However the later

strategy solves the exact problem (5.3) using a two-level optimization. In particular, the two-level optimization strategy solves the augmented Lagrangian relaxation problem by further reformulating it into two levels: in the first level, the relaxation problem is solved only with respect to the planning decision variables through an iterative algorithm, whereas in every iteration, a set of scheduling subproblems needs to be solved with fixed planning decision variables in the second level as shown in Figure 5.3. Notice that in the two-level strategy, the relaxation problem and the scheduling subproblems are in different levels, and in the DQA strategy, the planning subproblem and scheduling subproblem are solved in the same level but alternately.

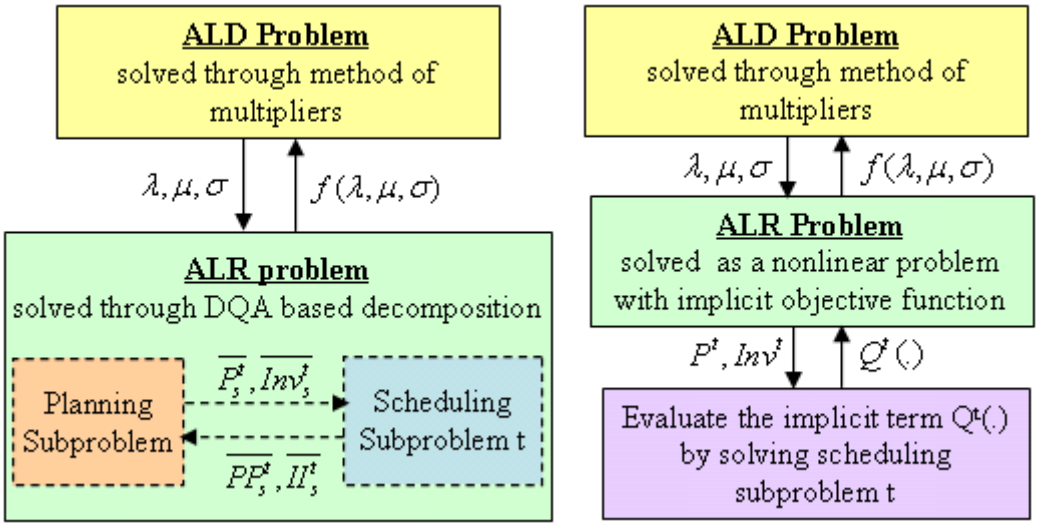


Figure 5.3 Illustration of the decomposition strategy: (left) DQA; (right) Two-level

Finally, it should be mentioned that in the DQA method, the solution generated by solving subproblem (5.4) and (5.5) alternately is actually an approximate solution of the original relaxation problem (5.3). As explained previously, the theory provided by (Andreani et al., 2008) provides support for this kind of approximation method. Similarly, we can use this idea in the two-level optimization strategy as follows. It is known that $q^t(P, Inv)$ is generally a nonsmooth function of P, Inv because of the integrality restrictions. Theoretically, nonsmooth optimization method should be used to ensure the optimality of the solution. However, considering the difficulty of solving the nonsmooth problem (5.7) and due to the fact that an optimal solution is not necessary to ensure the convergence of the algorithm, we propose to use a continuous

solver to solve problem (5.7) to get a solution which is feasible but not optimal. In the next section, we make a comparative study on the two different decomposition strategies in the ALR solution framework.

5.5 Examples

The augmented Lagrangian algorithm and different decomposition strategies are studied in this section through an example production problem. All the computations in this example are performed on a dual-core system with 2.8GHz CPU and 1Gb RAM. In this example, two products P1 and P2 are produced through three processing stages utilizing three materials (Kondili et al., 1993). The state-task-network (STN) representation of this example is shown in Figure 5.4 and the problem data can be found from Chapter 3.

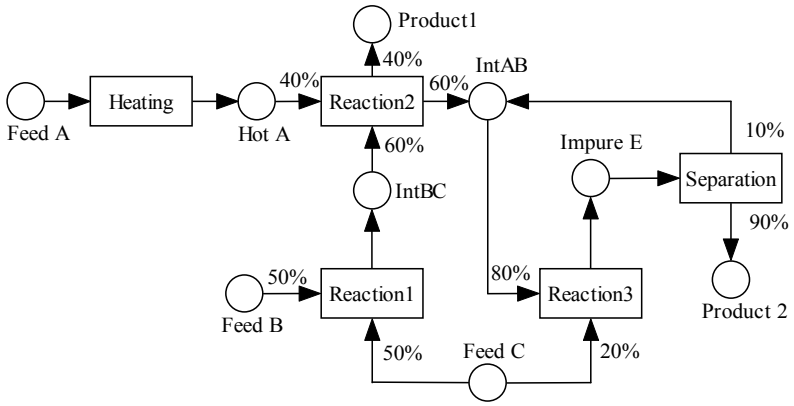


Figure 5.4 State-Task-Network (STN) representation of the motivation example

Considering the production planning and scheduling integration problem for the above production process, we divide the planning horizon into a number of planning periods with equal time length. In every planning period, an 8-hour scheduling problem is considered and 6 event points are used in the continuous time scheduling model as shown in model (5.1). Note that this number of event points is determined ahead of time with an objective of maximizing the production in the scheduling horizon of fixed 8-hour. Within such a time horizon and event point scheme, the resulted scheduling model can be efficiently and quickly solved through standard MILP solver such as CPLEX 10.

In the following, to study the augmented Lagrangian algorithm, we test six different cases of the planning and scheduling integration problem. Those six cases take different number of planning periods from 5 to 90 and the detail demand data can be referred from Figure 5.5 (e.g., for the 5-period case, the demand data are the first five data in the figure). Cost data for this problem can be found from Table 5.1.

Table 5.1 Cost data for the example

	Fixed cost	Variable cost
Heating	150	1
Reaction1,2,3	100, 100, 100	0.5, 0.5, 0.5
Separation	150	1
	Inventory cost	Backorder cost
P1, P2	10, 10	100, 100

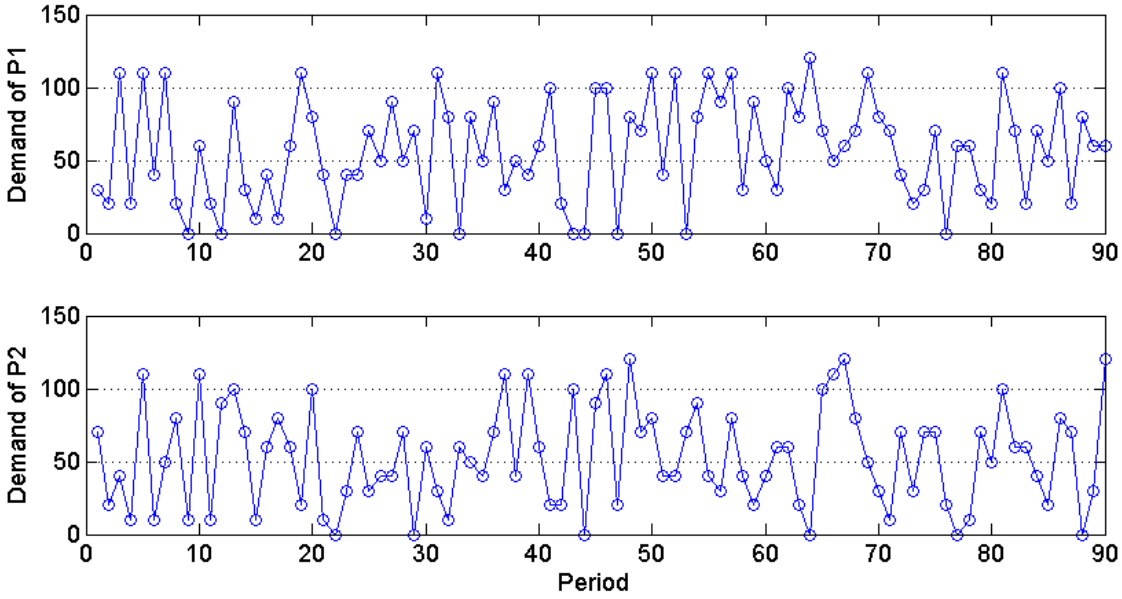


Figure 5.5 Demand data for 90 periods

Before the application of the augmented Lagrangian algorithm on the problem, we study the direct solution of the full space problem (5.1) using standard MILP solver CPLEX 10. The statistical data for the full space integrated planning and scheduling model with six different cases of planning periods and the results of direct solution method are shown in Table 5.2. It is observed that the problem is generally very difficult to be solved to optimality as the number of period increases and it becomes intractable when the number of periods is large (90 in this example).

Table 5.2 Model statistics and direct solution for full space model

Number of periods	Binary variables	Continuous variables	Constraints	Time	Best solution	Gap
5	600	2006	3847	3600*	**6576.8**	4.76%
10	1200	4001	7692	3600*	**13357.6**	14.04%
15	1800	5996	11537	7200*	**18985.9**	19.60%
30	3600	11981	23072	7200*	**35217.5**	17.49%
45	5400	17966	34607	10800*	**53960.3**	13.10%
90	10800	35921	69212	intractable	-	-

* Terminated because resource limit (time) is reached

The augmented Lagrangian method is then applied on this example and different decomposition strategies presented in section 4 are compared. First, for the DQA based decomposition strategy, we studied two different versions of the method and the results are shown in Table 5.3. The first version uses only one iteration for the solution of the relaxation problem and the other version uses increasing iteration limit (equal to the index of the outer iteration, noted as 'k-iteration' in the following) for the solution of the relaxation problem. In the solution procedure, CPLEX 10 is used in GAMS platform to solve both the planning subproblem (QP problem) and the scheduling subproblems (MIQP problems).

Table 5.3 Result of the DQA method

	With one iteration ($k_{\text{inner}}^{\max}=1$)					With two iterations ($k_{\text{inner}}^{\max}=k_{\text{outer}}$)				
T	k	time	F	$\lambda g+\sigma\|g\|^2$	$\|g\|$	k	time	f	$\lambda g+\sigma\|g\|^2$	$\|g\|$
5	11	62	**6925.2**	-3.9	0.39	14	554	**6775.3**	104.8	0.81
10	15	135	**13684.7**	-0.6	0.58	14	899	**13525.8**	42.8	0.80
15	15	157	**21113.8**	72.9	0.80	20	2151	**20000.9**	83.4	0.72
30	33	657	**39312.6**	157.1	0.84	17	3427	**36568.3**	3.6	0.50
45	33	1023	**59135.4**	164.0	0.88	21	7338	**55753.1**	-71.1	0.60
90	34	2462	**126894.2**	44.5	0.93	25	24510	**122122.1**	-101.4	0.76

Then, the proposed two-level optimization strategy is applied for the solution of the augmented Lagrangian problem. To address the implicit objective function (5.7a), we use the nonlinear programming solver KNITRO (Waltz & Plantenga, 2006) in MATLAB platform to solve the inner optimization problem (5.6) with the maximum iteration limit set as 50. Scheduling subproblems (MIQP) are solved using CPLEX 10 in GAMS. Note that although problem (5.7) is generally nonsmooth and KNITRO is a solver for smooth optimization problem, it is used here to obtain a feasible solution to the corresponding problem. We test the same group of problems as with the DQA approach and the computation results are shown in Table 5.4.

Table 5.4 Result of the two-level method

T	k	time	f	$\lambda g+\sigma\|g\|^2$	$\|g\|$
5	8	1245	**6648.2**	0.2	0.01
10	8	4983	**13371.9**	-79.9	0.71
15	9	6459	**19535.1**	-163.7	0.75
30	8	8443	**36223.4**	-1.2	0.03
45	9	12243	**54977.4**	36.9	0.42
90	9	37875	**121274.4**	-10.3	0.29

In all the computation results shown in Tables 5.3 and 5.4, column 'T' represents the number of planning periods, column 'k' represents the number of outer iterations in the augmented Lagrangian method, column 'time' represents the time used in seconds for the computation, column 'f' represents the final value of the

augmented Lagrangian function, column '$\lambda g+\sigma\|g\|^2$' represents the value of the augmented and penalty term in the augmented Lagrangian function, '$\|g\|$' represents the norm of the consistency constraint function value vector. Figure 5.6 presents the solution procedure of the augmented Lagrangian method for the 90-periods problem.

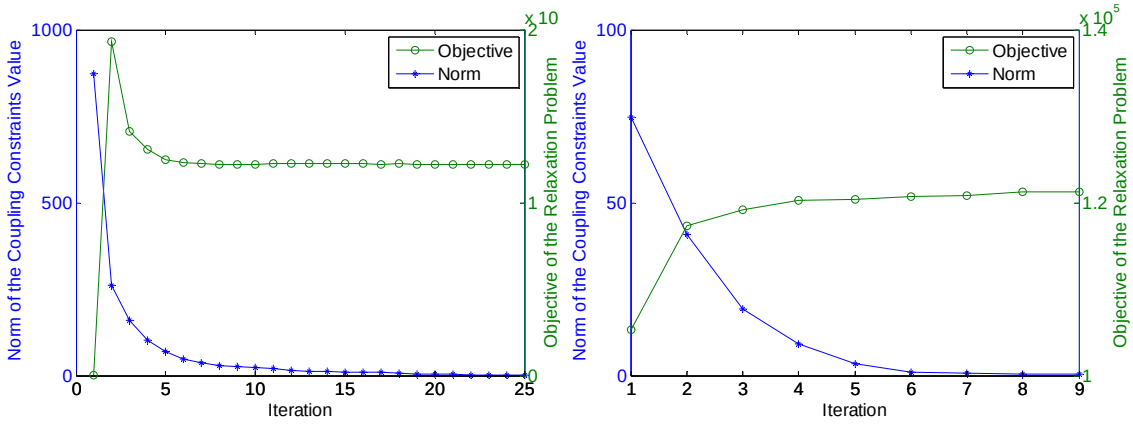


Figure 5.6. Solution procedure: (left) DQA with k-iteration; (right) Two-level optimization

From the above results, it can be observed that the augmented Lagrangian algorithm converges to a feasible solution of the original problem since the norm value of the coupling constraints always converges to zero. Note that this property is independent of the decomposition strategy used. To illustrate the feasibility of the solution, we also plot the solution of the production data along with the scheduling feasibility boundary which is generated through parametric programming technique (Li & Ierapetritou, 2007a) for the 90-periods case in Figure 5.7. It is observed that the solution data points are all inside the feasibility boundary.

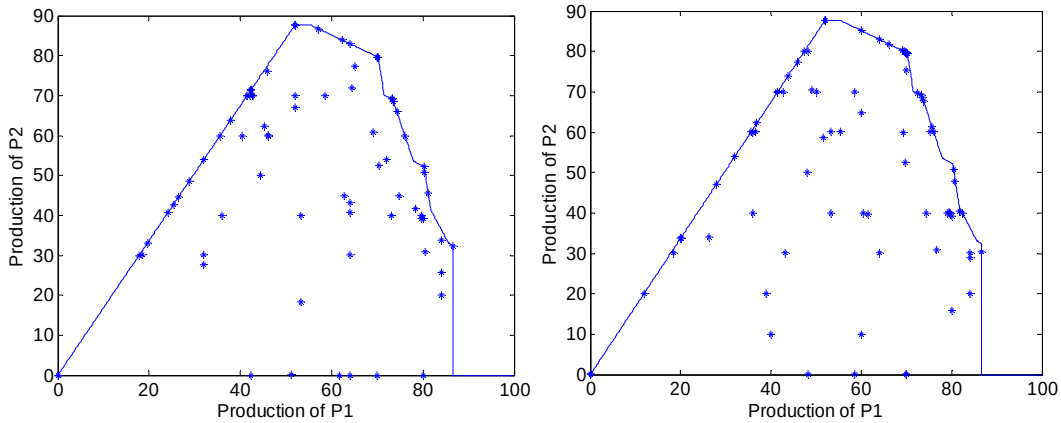


Figure 5.7 Feasibility of solution: (left) DQA with k-iteration; (right) two-level optimization

Although both decomposition strategies ensure the convergence of the solution, it is worth noticing that the efficiency and quality of the solution as analyzed in the following. First, for the DQA strategy, it is observed from the two different versions in Table 5.3 that if more iterations are used for the solution of the relaxation problem, generally less outer iterations will be required. However the increased computational complexity does not reflect obvious quality improvement of the final solution.

Second, for the two-level decomposition method, it can be observed that it takes relative small number of outer iterations and can get feasible solutions which are better than the results of DQA method. On the other hand, the computation time needed for the two-level optimization method is more than the time needed for the DQA approach with fixed one iteration, but comparable to the DQA with increasing iteration limits. However, the quality of the solution for two-level method is better than all the DQA cases, i.e., although more or comparable computation time is required, better solution is achieved by using the two-level optimization strategy.

The results for this problem are shown in Figures 5.8-5.10. In particular Figure 5.8 illustrates the production of products P1 and P2. As shown from this figure the production in the solution produced by DQA method is more compared with the solution from the two-level approach. Figure 5.9 illustrate the inventory of products P1 and P2. As shown by the figure the inventory amount in the solution of the DQA method is more than that of the two-level optimization method, leading to higher inventory cost. Finally as shown from Figure 5.10 that illustrates the backorder amount, the backorder amount in the solution of two-level case is almost zero for all periods, but the solution of DQA takes a relative large backorder in the 7^{th} period. Thus it can be observed from these results that the quality of the DQA solution is inferior compared to the solution of the two-level strategy.

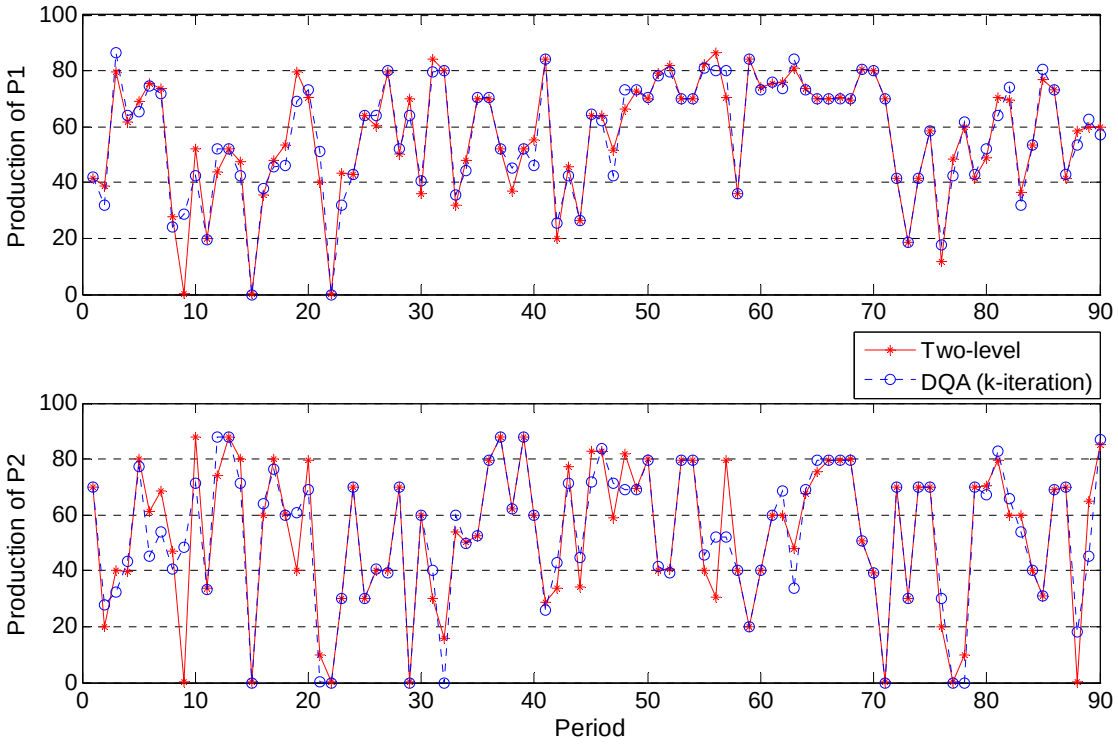


Figure 5.8 Production profile of the solution

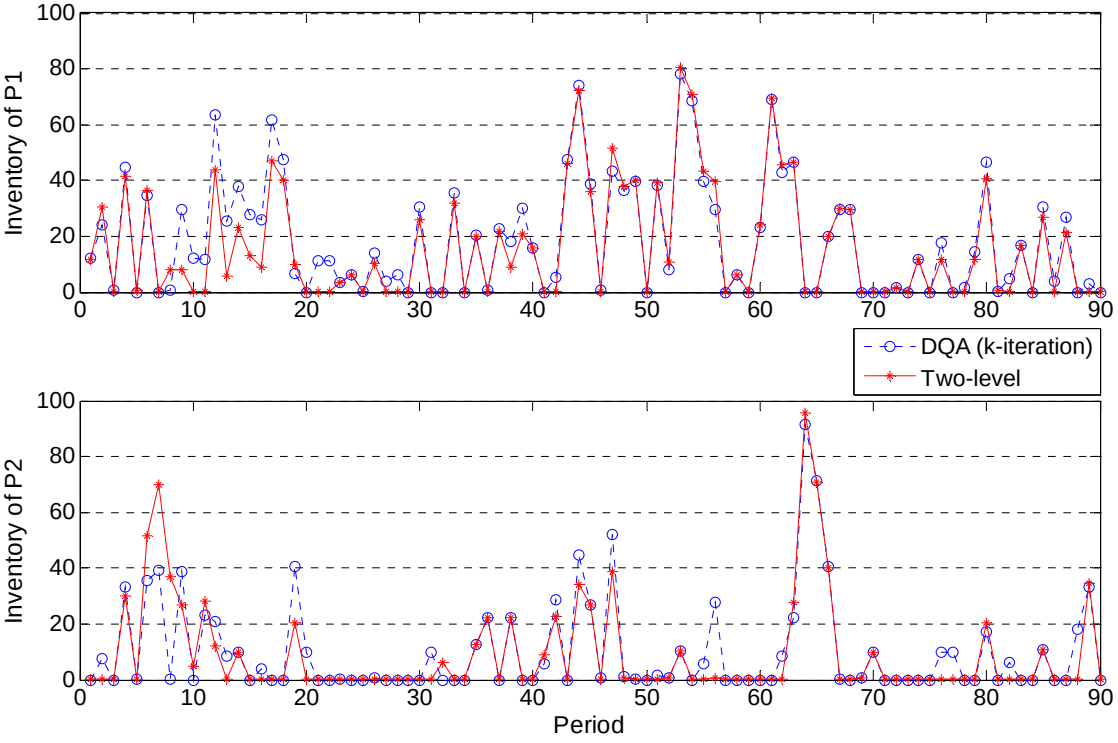


Figure 5.9 Inventory profile of the solution

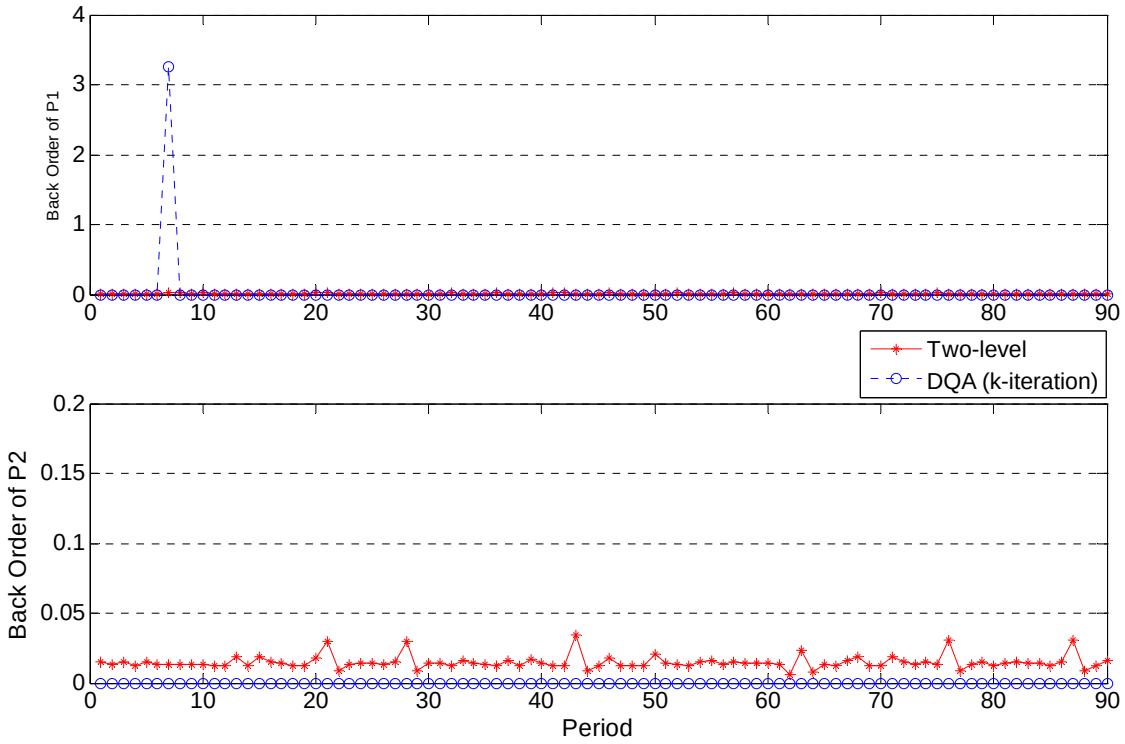


Figure 5.10 Backorder profile of the solution

5.6 Summaries

To address the problem of integrated production planning and scheduling, a decomposition algorithm based on augmented Lagrangian is proposed in this chapter. Based on the special structure of the optimization model, auxiliary variables and coupling constraints for the linking variables are first introduced, the coupling constraints are then relaxed and the resulted augmented Lagrangian relaxation problem is solved through decomposition technique. We also propose a new decomposition strategy based on two-level optimization of the relaxation problem and compare its performance with traditional approximation based decomposition strategy. The results from a case study show that the augmented Lagrangian method can effectively generate feasible solution for the original problem, and the new decomposition strategy can generate better feasible solution than the traditional approximation based method with the trade-off of using more or comparable computation efforts. Furthermore, it is also worth noticing that the computation time in the method is mostly spent on the solution of scheduling subproblems. By realizing that the subproblems can be further solved in parallel, we can reduce further the computation time through parallel computing.

The main advantages of the augmented Lagrangian method are: (a) the convergence of the algorithm is ensured without the need to solve the relaxation problem to optimality; (b) it can be easily parallelized; and (c) it is able to avoid the duality gap. Furthermore, it can be also used within a bounding procedure since a feasible solution is always ensured. In summary, the augmented Lagrangian method is appropriate for the solution of the planning and scheduling integration problem. Future work will include improving the solution of the relaxation problem to find the global optimal solution of the original problem.

Nomenclature

Planning part

t	planning periods $(1,...,T)$
Inv_s^t	inventory level of state s at the end of planning period t
P_s^t	production target of state s in planning period t
D_s^t	delivery of product s in planning period t
U_s^t	backorder of product s in planning period t
Dem_s^t	demand of product s in planning period t
h_s	inventory unit cost of state s
u_s	backorder unit cost of product s

Scheduling part

$i \in I$	task index and sets
I_s	tasks which produce or consume state s
I_j	tasks which can be performed in unit j
$j \in J$	unit index and sets
J_i	units which are suitable for performing task i
$n \in N$	event points representing the beginning of a task
$s \in S$	state index and sets
S_P	index set for products
$wv_{i,j,n}$	binary, whether or not task i in unit j start at event point n
$st_{s,n}$	continuous, amount of state s at event point n
$\rho_{s,i}^P$, $\rho_{s,i}^C$	proportion of state s produced, consumed by task i, respectively
$b_{i,j,n}$	continuous, amount of material undertaking task i in unit j at event point n
$st_s^{\max}$	available maximum storage capacity for state s
$stin_s^t$	initial inventory for state s in planning period t
$v_{i,j}^{\min}$, $v_{i,j}^{\max}$	minimum amount, maximum capacity of unit j when processing task i
$Tf_{i,j,n}$	continuous, time at which task i finishes in unit j while it starts at event point n
$Ts_{i,j,n}$	continuous, time at which task i starts in unit j at event point n
$\alpha_{i,j}$, $\beta_{i,j}$	constant, variable term of processing time of task i in unit j respectively
H	scheduling time horizon

Chapter 6

Rolling Horizon Optimization

Abstract: Rolling horizon method has been proposed to address the integrated production planning and scheduling optimization problem. Since the method can generally result in small-scale optimization model and fast solution, it has received quite a few applications in realistic industrial planning and scheduling problems. In this chapter, we first pointed out that the incorporation of valid production capacity information into the planning model can improve the solution quality. Then we proposed a novel method to derive the production capacity model representing the scheduling problem based on parametric programming technique. A heuristic process network decomposition strategy is further applied to reduce the computational effort needed for complex realistic process networks. Several case studies illustrate the efficiency of the proposed methodology in improving the solution quality of rolling horizon method for integrated planning and scheduling optimization.

6.1 Introduction

Production planning and scheduling are two important decision making levels in process operations. Traditionally, planning and scheduling have been performed separately (Kallrath, 2002). The planning problem is typically solved to predict production targets and material flow over a mid-term horizon (e.g. several months) to satisfy the customer demand. The scheduling problem is usually addressed after the production planning problem has been solved. The data generated by the production planning problem are input data to the scheduling problem. The purpose of the scheduling problem is to transform the production plan into a feasible schedule of all the production operations within a short-term time horizon (e.g., several days).

However, treating the planning and scheduling activities separately can lead to lower efficiency of the operations performed in the production plant. The aggregate production targets supplied by the planning

model often overestimate the production capacity of the plant and may result in infeasible scheduling operations because they are made without consideration of short-term operational restrictions. Given the importance of providing realistic production targets, a production planning model should take into account not only the customer demands but also the production capacity of the plant. To address this issue, the integration of planning and scheduling has been proposed by the process systems engineering community(Maravelias & Sung, 2008). The integration aims to address the inaccuracies within the planning model by allowing for the two-way interaction between planning and scheduling models. Ideally, an integrated model should include not only the medium-term capacity utilization and production level decisions but also the short-term production sequence and unit assignment decisions. One simple approach is to use a scheduling model over the entire planning time horizon, which takes into account the production capacity of the plant. However, this approach results in problems of unrealistic size, which is often computationally intractable.

To ensure that the integration can be addressed efficiently, planning models are often formulated through various types of aggregation or relaxation schemes, and the integration problem is often solved through decomposition algorithms (Maravelias & Sung, 2008), (Grossmann et al., 2002). In the literature, there are a number of decomposition based methods like the hierarchical decomposition (Bassett, Dave et al., 1996); (Munawar & Gudi, 2005); (Erdirik-Dogan & Grossmann, 2006)), periodic scheduling (Schilling & Pantelides, 1999); (Zhu & Majozi, 2001); (Castro et al., 2003); (Wu & Ierapetritou, 2004), mathematical programming based decomposition (Li & Ierapetritou, 2009), as well as methods that are based on the rolling horizon idea which is widely studied (Kreipl & Pinedo, 2004) because it can significantly reduce the computational requirements. The method is based on iteratively solving the integrated problem in a rolling time horizon mode. In every iteration, the detailed scheduling requirements are imposed only for the current or several recent planning periods. In the next iteration, the new planning decision is updated with all the previous executed decisions fixed. This mode is repeated until all the planning periods are considered. The above idea is supported by the fact that planning decisions for far future could not be accurate enough due to the unpredicted future uncertainty. So it is reasonable to consider a relative rough model for far future planning periods in the aggregate planning model. Thus the rolling horizon approach results in reduced size models and lower computation cost.

Rolling horizon method has received a lot of studies in the literature. (Rodrigues et al., 1996) use a rolling horizon (rolling out a predefined schedule) approach to take account of due-date changes and equipment unavailability to resolve infeasibilities. (Dimitriadis et al., 1997) presented an RTN-based rolling horizon algorithm for medium term scheduling of multipurpose plants. (Sand et al., 2000) use a rolling horizon approach, in combination with a Lagrangian relaxation algorithm, for the solution of a two-level hierarchical planning and scheduling problem. (Wu & Ierapetritou, 2007) decompose the planning time horizon into three stages with various durations. The scheduling problem is solved after the solution of planning model to ensure a feasible production schedule for the current period. (Sand & Engell, 2004) use a rolling horizon, two-stage stochastic programming approach to schedule an expandable polystyrene plant that is subject to uncertainty in processing times, yields, capacities and demands. (Verderame & Floudas, 2008) solve the integration problem utilizing the medium-term scheduling model for large-scale batch plants and a forward rolling horizon approach. Rolling horizon method has also been applied to address the long term and medium term scheduling problem (Lin et al., 2002), (Janak et al., 2006), (Shaik et al., 2007), (Shaik et al., 2009). For those scheduling problems, a rolling-horizon based decomposition scheme is used and usually two sub-problems are solved. At the upper-level, a variant of the model is used to find the optimal number of products, and the length of the time horizon to be considered for solving the short-term scheduling problem at the lower level. At the lower level, short-term scheduling of continuous processes using unit-specific event-based continuous-time representation are applied.

Although rolling horizon framework has received a lot of attention in the literature, a major drawback of most existing methods is that they often rely on the simplistic or rather poor representation of the scheduling problem within the aggregate part. Within such a modeling framework, rolling horizon method is generally efficient in the computational manner. However, the method only ensures the feasibility of the final solution. As what we will show in this chapter, production capacity information representing the scheduling problem can have great effect on the final solution's quality. In the literature, (Sung & Maravelias, 2007) have proposed to derive the feasible production regions for scheduling problem through a computational geometry method, and then incorporate it into the rolling horizon planning model. In this work, we are proposing a new method to derive the production capacity constraints based on short-term scheduling model through

parametric programming, which can be used in the rolling horizon framework to greatly improve the final solution's quality.

The content of this chapter is organized as follows. The rolling horizon solution framework and model are presented in section 6.2, which is further studied by a motivation example illustrating the necessity of applying production capacity information to improve the solution quality. In section 6.3, we present a parametric programming based method which is able to generate the accurate boundary of the production capacity region of scheduling problem, and also a heuristic process network decomposition strategy to reduce the computation complexity. In section 6.4, we illustrated the application of the proposed method on several complex problems. The chapter concludes in section 6.5 with a summary of the presented work.

6.2 Rolling horizon framework

Rolling horizon methods solve the planning and scheduling integration problem within a sequence of iterations, each of which models only part of the planning horizon in detail, while the rest of the horizon is represented in an aggregate manner. The rolling horizon solution framework involves successively solving each scheduling sub-horizon and carrying over any unsatisfied demand to the following sub-horizon. In principle, this approach produces feasible planning and scheduling solutions with a significant reduction of the computational requirements.

Generally, discrete time representation is used for the planning time domain. Consider the planning and scheduling integration problem over a time horizon H. In order to integrate both planning and scheduling into the optimization model, H is divided into a number of planning periods, $t = 1 \ldots T$. The length of the planning horizon is typically in the order of few months. In the rolling horizon framework, both uniform and non-uniform planning periods of fixed or varying length can be applied. For example, in production planning, it is often required to determine weekly production targets for the first 2-4 weeks, while monthly production targets are sufficient for subsequent periods. Hence, we can consider the non-uniform planning periods ranging from weeks to months.

To describe the scheduling model, discrete and continuous models can both be applied. In a discrete time representation (which assumes that an event can occur only at the boundaries of each time interval), every planning period in the time horizon is divided into a number of predefined scheduling periods, k =

1…*K*. The length of a scheduling period is typically in the order of hours. In the continuous time representation, an event can occur at any instant within the whole planning horizon. This makes the model more flexible and decreases the total number of variables.

In the following paragraphs, we present a general planning model and a continuous time representation based scheduling model, which forms a basis for the rolling horizon framework studied in this chapter. It should be pointed out however that for long-term or medium-term scheduling problems, similar formulation idea can be applied whereas a planning problem is not involved.

Planning model:

$$\min \quad TotalCost = \sum_t \left(\sum_s h_s I_s^t + \sum_s u_s U_s^t + \sum_s v_s P_s^t \right) \tag{6.1a}$$

$$s.t. \quad I_s^t = I_s^{t-1} + P_s^t - D_s^t \qquad \forall s \in S_P, \forall t \tag{6.1b}$$

$$U_s^t = U_s^{t-1} + Dem_s^t - D_s^t \qquad \forall s \in S_P, \forall t \tag{6.1c}$$

$$P_s^t = \overline{P}_s^t \qquad \forall t \in T^{pre} \tag{6.1d}$$

$$f(P_s^t) \le 0 \qquad \forall s \in S_P, \forall t \tag{6.1e}$$

$$P_s^t, I_s^t, D_s^t, U_s^t \ge 0 \qquad \forall s \in S_P, \forall t$$

The above planning model is similar to the one given in (Sung & Maravelias, 2007). In the problem, the objective function is the total cost which is composed by three parts: inventory cost, backorder cost and production cost. Equation (6.1b) represents the inventory balance and equation (6.1c) represents the backorder balance. Equation (6.1d) fixes those planning decision that have been "executed" by the scheduling model. Constraints (6.1e) represents the production capacity constraints.

Scheduling model:

$$\min \quad \sum_s (\varepsilon_s^+ + \varepsilon_s^-) + \gamma \cdot \text{ProdictionCost} \tag{6.2a}$$

$$\text{s.t.} \quad \text{ProductionCost} = \sum_i \sum_j \sum_n (FixCost_i w_{i,j,n} + VarCost_i b_{i,j,n}) \tag{6.2b}$$

$$P_s - \overline{P_s} = \varepsilon_s^+ - \varepsilon_s^- \qquad \forall s \in S_P \tag{6.2c}$$

$$st_{s,n=N} - stin_s = P_s \qquad \forall s \in S_P \qquad (6.2d)$$

$$\sum_{i \in I_j} wv_{i,j,n} \le 1 \qquad \forall j \in J, \forall n \in N \qquad (6.2e)$$

$$v_{i,j}^{\min} wv_{i,j,n} \le b_{i,j,n} \le v_{i,j}^{\max} wv_{i,j,n} \qquad \forall i \in I, \forall j \in J_i, \forall n \in N \qquad (6.2f)$$

$$Tf_{i,j,n} = Ts_{i,j,n} + \alpha_{i,j} wv_{i,j,n} + \beta_{i,j} b_{i,j,n} \qquad \forall i \in I, \forall j \in J_i, \forall n \in N \qquad (6.2g)$$

$$Ts_{i,j,n+1} \ge Tf_{i,j,n} - H(1 - wv_{i,j,n}) \qquad \forall i \in I, \forall j \in J_i, \forall n \in N \qquad (6.2h)$$

$$Ts_{i,j,n+1} \ge Tf_{i',j,n} - H(1 - wv_{i',j,n}) \qquad \forall i, i' \in I_j, \forall j \in J, \forall n \in N \qquad (6.2i)$$

$$Ts_{i,j,n+1} \ge Tf_{i',j',n} - H(1 - wv_{i',j',n}) \qquad \forall i, i' \in I_j, i \ne i', \forall j, j' \in J, \forall n \in N \qquad (6.2j)$$

$$Ts_{i,j,n+1} \ge Ts_{i,j,n} \qquad \forall i \in I, \forall j \in J_i, \forall n \in N \qquad (6.2k)$$

$$Tf_{i,j,n+1} \ge Tf_{i,j,n} \qquad \forall i \in I, \forall j \in J_i, \forall n \in N \qquad (6.2l)$$

$$Ts_{i,j,n} \le H \qquad \forall i \in I, \forall j \in J_i, \forall n \in N \qquad (6.2m)$$

$$Tf_{i,j,n} \le H \qquad \forall i \in I, \forall j \in J_i, \forall n \in N \qquad (6.2n)$$

$$st_{s,n} = st_{s,n-1} - \sum_{i \in I_s} \rho_{s,i}^{C} \sum_{j \in J_i} b_{i,j,n} + \sum_{i \in I_s} \rho_{s,i}^{P} \sum_{j \in J_i} b_{i,j,n-1} \qquad \forall s \in S, \forall n \in N \qquad (6.2o)$$

$$st_{s,n} \le st_s^{\max} \qquad \forall s \in S, \forall n \in N \qquad (6.2p)$$

$$wv_{i,j,n} \in \{0,1\}, b_{i,j,n}, st_{s,n}, Tf_{i,j,n}, Ts_{i,j,n} \ge 0$$

The objective (6.2a) aims at finding a feasible schedule minimizing the sum of the absolute difference between the result generated by the planning model $\overline{P_s^t}$ and feasible schedule, plus a weighted production cost (γ is a weight coefficient). Here, the production cost is calculated in (6.2b) by a fixed part which represents the basic cost of a task, and a dynamic part which is proportional to the amount of material processed (batch size). Among the constraints of the scheduling level problem, equation (6.2c) uses slack variables to evaluate the difference between given production target and actual production amount. Equation (6.2d) defines the actual production amount. The rest constraints has the similar meaning as explained in problem (2.8). It should be noticed that although the above scheduling formulation derived from (Ierapetritou

& Floudas, 1998) has been used, the proposed methodology in this chapter is not limited to this model because the proposed solution method is appropriate to general MILP scheduling model.

Based on the above planning and scheduling formulations, the following algorithm describes a rolling horizon algorithm of the solution of the integrated planning and scheduling problem.

Rolling horizon algorithm

Step 1 Set the first planning period as "current period", solve the planning problem.

Step 2. Using the production target solution obtained from step 1, solve scheduling problem in current period. If "current period" is the last planning period of the problem, stop. Otherwise, go to step 3.

Step 3. Fix the production target in current period at the values obtained in step 2 and solve a new planning problem; update the current period index; go to step 2.

In the following, an example is used to illustrate the above algorithm and also the effect of production capacity constraints on the quality of the final solution.

Motivation Example

In this motivation example, two products P1 and P2 are produced through three processing stages utilizing three materials. The state-task-network (STN) representation of this example is shown in Figure 6.1. Detail data for this problem can be found from Chapter 3.

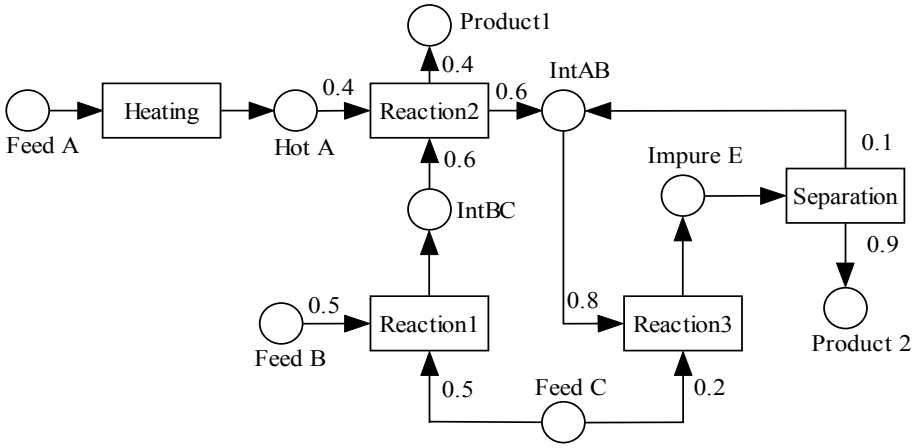


Figure 6.1 State Task Network for example 1

For the sake of simplicity in illustrating the results, 5 planning periods are considered in this problem. The scheduling horizon is set as 8 hours and 7 event points are used for the continuous time scheduling formulation (6.2). The cost and demand data for this problem are shown in Table 6.1.

Table 6.1 Cost and demand data for motivation example

	Fixed cost	Variable cost
Heating	150	1
Reaction1,2,3	100, 100, 100	0.5, 0.5, 0.5
Separation	150	1
	Unit inventory cost	10, 10
P1, P2	Unit backorder cost	100, 100
	Unit production cost (used in planning model)	1.5, 1.5
$\begin{bmatrix} Dem_{P_1} \\ Dem_{P_2} \end{bmatrix}$	$\begin{bmatrix} 0 & 60 & 140 & 30 & 100 \\ 50 & 0 & 55 & 65 & 150 \end{bmatrix}$	

Based on the given process network and corresponding production recipe, simple valid inequalities can be derived to describe the production capacity. First, we can identify a production limit from the scheduling model by reformulating the scheduling problem objective function as maximizing the production of certain product $P_s^U = \max \ (st_{s,n=N} - stin_s)$, and setting the demand of other products as free variables. The solution of this problem leads to the determination of an upper bound of the possible production target of this product as $P_1 \le 86.67$, $P_2 \le 87.75$. Second, for this production problem, a valid inequality representing an upper bound for the ratio between production amount of Product 2 (P2) and production amount of Product 1 (P1) can be derived through examination of production recipe: for every P1 units of Product 1 been processed, the maximum possible amount of Product 2 to be produced can be determined from the mass balance equation: $P_2 = (1.5P_1 + P_2 / 9) \times 1.25 \times 0.9$. By solving this equation, we have the following valid inequality: $P_2 \le 1.93P_1$. Thus those simple production capacity constraints can be incorporated into the planning model as shown in equation (6.1e).

Based on the above analysis, the rolling horizon algorithm is applied to solve this example, the solution procedure for the case with and without capacity model are both listed in Table 6.2, where the "planning result" column denotes the solution P_s^t obtained from planning problem (6.1) and the "scheduling result" column denote the solution P_s obtained through the solution of scheduling problem (6.2). The final solution is shown in Table 6.3.

Table 6.2 Comparison of the rolling horizon solution procedures

k	Without capacity information		With simple capacity constraints	
	Planning result	Scheduling result	Planning result	Scheduling result
1	$\begin{bmatrix} 0 & 60 & 140 & 30 & 100 \\ 50 & 0 & 55 & 65 & 150 \end{bmatrix}$	$\begin{bmatrix} 29.6 \\ 50 \end{bmatrix}$	$\begin{bmatrix} 27 & 86.5 & 86.5 & 45.3 & 84.7 \\ 50 & 7.5 & 87.5 & 87.5 & 87.5 \end{bmatrix}$	$\begin{bmatrix} 29.6 \\ 50 \end{bmatrix}$
2	$\begin{bmatrix} 29.6 & 60 & 140 & 30 & 100 \\ 50 & 0 & 55 & 65 & 150 \end{bmatrix}$	$\begin{bmatrix} 30.4 \\ 0 \end{bmatrix}$	$\begin{bmatrix} 29.6 & 83.9 & 86.5 & 45.3 & 84.7 \\ 50 & 7.5 & 87.5 & 87.5 & 87.5 \end{bmatrix}$	$\begin{bmatrix} 83.9 \\ 7.5 \end{bmatrix}$
3	$\begin{bmatrix} 29.6 & 30.4 & 140 & 30 & 100 \\ 50 & 0 & 55 & 65 & 150 \end{bmatrix}$	$\begin{bmatrix} 77.5 \\ 55 \end{bmatrix}$	$\begin{bmatrix} 29.6 & 83.9 & 86.5 & 45.3 & 84.7 \\ 50 & 7.5 & 87.5 & 87.5 & 87.5 \end{bmatrix}$	$\begin{bmatrix} 70.2 \\ 79.6 \end{bmatrix}$
4	$\begin{bmatrix} 29.6 & 30.4 & 77.5 & 92.5 & 100 \\ 50 & 0 & 55 & 65 & 150 \end{bmatrix}$	$\begin{bmatrix} 74.5 \\ 65 \end{bmatrix}$	$\begin{bmatrix} 29.6 & 83.9 & 70.2 & 59.8 & 86.5 \\ 50 & 7.5 & 79.6 & 87.5 & 87.5 \end{bmatrix}$	$\begin{bmatrix} 59.8 \\ 85.3 \end{bmatrix}$
5	$\begin{bmatrix} 29.6 & 30.4 & 77.5 & 74.5 & 100 \\ 50 & 0 & 55 & 65 & 150 \end{bmatrix}$	$\begin{bmatrix} 70.2 \\ 79.6 \end{bmatrix}$	$\begin{bmatrix} 29.6 & 83.9 & 70.2 & 59.8 & 86.5 \\ 50 & 7.5 & 79.6 & 85.3 & 87.5 \end{bmatrix}$	$\begin{bmatrix} 70.2 \\ 79.6 \end{bmatrix}$

Table 6.3 Comparison of the final solution results

	Without capacity constraints	With simple capacity constraints
Production	$\begin{bmatrix} 29.6 & 30.4 & 77.5 & 74.5 & 70.2 \\ 50 & 0 & 55 & 65 & 79.6 \end{bmatrix}$	$\begin{bmatrix} 29.6 & 83.9 & 70.2 & 59.8 & 70.2 \\ 50 & 7.5 & 79.6 & 85.3 & 79.6 \end{bmatrix}$
Inventory	$\begin{bmatrix} 29.6 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$	$\begin{bmatrix} 29.6 & 53.5 & 0 & 13.5 & 0 \\ 0 & 7.5 & 32.1 & 53.4 & 0 \end{bmatrix}$
Backorder	$\begin{bmatrix} 0 & 0 & 62.5 & 17.9 & 47.7 \\ 0 & 0 & 0 & 0 & 70.4 \end{bmatrix}$	$\begin{bmatrix} 0 & 0 & 16.3 & 0 & 16.3 \\ 0 & 0 & 0 & 0 & 17.9 \end{bmatrix}$
Production cost	5312.8	6144.4
Inventory cost	296.3	1886.6
Backorder cost	19849.7	5059.2
Total cost	**25458.8**	**13090.2**

From the above results, it can be observed that the total cost for the case without production capacity information is much higher than that of the case with simple capacity constraints, and the difference is mainly due to increased backorder cost. The reason is that in the model without production capacity constraints, the planning model cannot predict that the future production will not satisfy the demand, but assumes that the future demand will be always satisfied with enough production capacity, so it tends to produce as close as possible to the demand in current period so as to minimize the inventory cost. Due to this production capacity limitation that is not accurately considered, the model without the capacity constraints results in higher backorder cost, and thus higher total cost.

The above simple implementation of a rolling horizon approach for this motivating example illustrates that the incorporation of production capacity constraints representing production capacity at the planning level problem can result in huge savings in terms of backorder cost (74% reduction) and significant reduction

of the overall cost (49% reduction). Thus it is important to point out that the consideration of production capacity information will improve the quality of the overall solution for the case of a rolling horizon solution procedure.

In this example problem, for illustrative purposes the production capacity information included represented an approximation. A more accurate method based on parametric programming is presented in the next section, which will further improve the quality of the solution.

6.3 Production capacity model derivation

Although the simple production capacity constraints derived from production recipe can improve the quality of the solution in the rolling horizon method, however, they are still an approximation of the exact production capacity of a short-term scheduling problem because they only represent the mass balance information.

In this chapter, we are proposing to develop more accurate production capacity model through parametric programming. From a mathematical point of view, the exact production capacity region for the scheduling problem (6.2) is the projection of the scheduling problem's feasible region onto the subspace spanned by the planning variables, i.e., the production targets P_s. The parametric programming method can evaluate the exact production capacity by exploring the boundary of the production capacity region segment by segment. The details of the method are presented in the follows.

6.3.1 Parametric programming

The production capacity information can be completely described by the boundary (plane or line) of the production feasibility region. Theoretically, to retrieve production capacity information from the scheduling model, we can set the objective of the scheduling problem as maximizing the production of a certain product s^* and set the production amount of the other products at specific fixed values as follows

$$\max \quad P_{s*}=st_{s*,n=N}-stin_{s*} \tag{6.3a}$$

$$\text{s.t.} \quad \text{(2b)-(2o)} \tag{6.3b}$$

$$st_{s,n=N}-stin_s=P_s,\ \forall s\in S_P, s\neq s* \tag{6.3c}$$

Thus by enumerating all possible values of the production amount for products $s \in S_P, s \neq s^*$, we can identify a set of points which is on the boundary (plane or line) of the production capacity region. However, formulating the boundary plane or line requires the evaluation of infinite number of points and correspondingly the solution of infinite number of scheduling problems, which is obviously impossible. However, we can apply the parametric programming algorithm from our previous work (Li & Ierapetritou, 2007b) to identify the parametric solution which represents the boundary of the production capacity region.

Parametric programming approach generates the optimal solution map of an optimization problem with uncertain parameters. From this point of view, parametric programming provides the exact mathematical solution of the optimization problem under parameter variability (Pistikopoulos et al., 2007). In the parametric programming method, the production amount of all the "other" products are viewed as uncertain parameters and their values can vary within a given range. The complete solution of the parametric programming problem is composed by the complete set of critical regions and optimal value functions described with respect to uncertain parameters. The critical region is defined as the range of parameter values where the same solution remains optimal. Thus we only need to evaluate a set of critical regions and optimal value functions to represent the boundary of the production capacity region.

To apply parametric programming, the original scheduling formulation (6.2) is rewritten as the following general compact form:

$$\min \quad cx \tag{6.4a}$$

$$\text{s.t.} \quad Ax + By = b + E\theta \tag{6.4b}$$

$$\theta \in [\theta^L, \theta^U] \tag{6.4c}$$

$$x \geq 0,\ y \in \{0,1\} \tag{6.4d}$$

where y represents the binary decision variables wv; x represents the continuous variables b, st, Tf, Ts; θ represent the production amount of the products P_s, $\forall s \in S_P, s \neq s^*$; $[\theta^L, \theta^U]$ represents an initial given range for those parameters θ, which can be determined by solving problem (6.3) without constraints (6.3c). Based on the above formulation, we can apply the parametric programming algorithm presented in Chapter 2 and a segment of the production capacity boundary can be derived once a set of values of the uncertain parameters are assigned. By varying the values of those parameters, the complete boundary can be identified.

To illustrate the application of the parametric programming algorithm for the solution of the production capacity region, the motivation example presented in section 6.2 is studied in the next subsection.

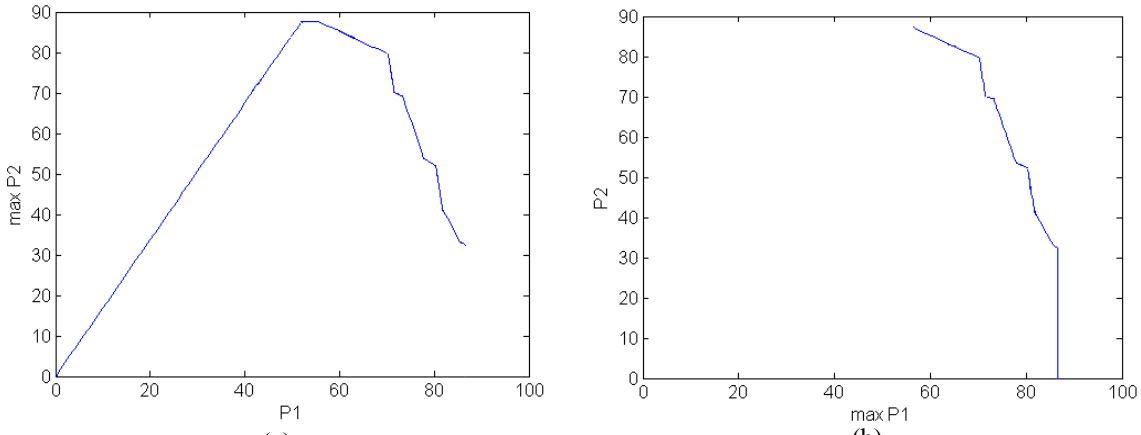
6.3.2 Application of parametric programming in motivation example.

To apply the parametric programming approach on the motivation example, we first set the production amount of P1 as parameter and the scheduling objective as maximizing the production of P2. Similarly, production of P1 can be maximized by viewing production of P2 as parameter. Then the application of the above parametric programming algorithm can result in the parametric solutions listed in Table 6.4.

Table 6.4 Parametric solution for the motivation example

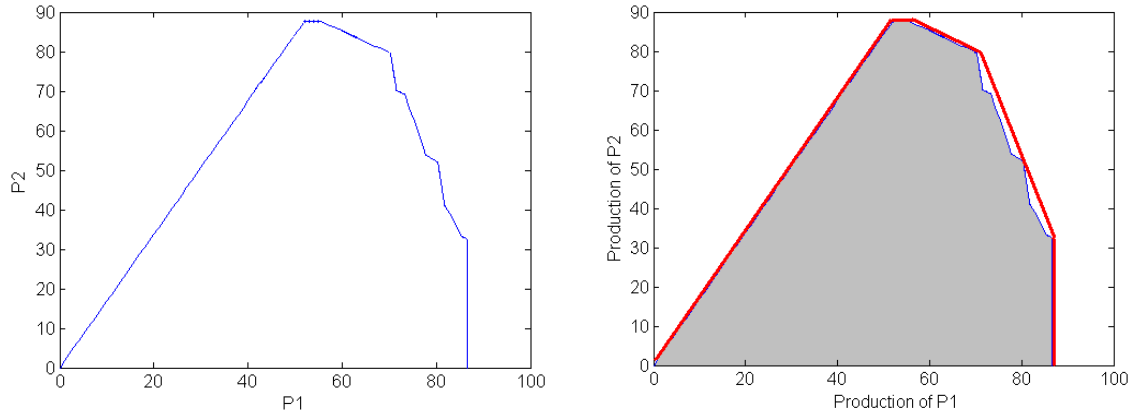
	$\max P_2$	Range of P_1		$\max P_1$	Range of P_2
1	$1.6875P_1$	$0 \le P_1 \le 52$	1	86.67	$0 \le P_2 \le 32.4$
2	87.75	$52 \le P_1 \le 55.25$	2	$136.67 - 1.543P_2$	$32.4 \le P_2 \le 33.07$
3	$117.785 - 0.5436P_1$	$55.25 \le P_1 \le 70.19$	3	$101.25 - 0.472P_2$	$33.07 \le P_2 \le 41.25$
4	$585.0 - 7.2P_1$	$70.19 \le P_1 \le 71.52$	4	$87.5 - 0.1389P_2$	$41.25 \le P_2 \le 52.24$
5	$104.62 - 0.4829P_1$	$71.52 \le P_1 \le 73.32$	5	$166.67 - 1.654P_2$	$52.24 \le P_2 \le 53.65$
6	$317.647 - 3.388P_1$	$73.32 \le P_1 \le 77.92$	6	$93.75 - 0.295P_2$	$53.65 \le P_2 \le 69.22$
7	$100.746 - 0.6045P_1$	$77.92 \le P_1 \le 80.24$	7	$216.67 - 2.071P_2$	$69.22 \le P_2 \le 70.09$
8	$630 - 7.2P_1$	$80.24 \le P_1 \le 81.77$	8	$81.25 - 0.1389P_2$	$70.09 \le P_2 \le 79.63$
9	$214.412 - 2.118P_1$	$81.77 \le P_1 \le 85.63$	9	$216.67 - 1.8395P_2$	$79.63 \le P_2 \le 87.75$
10	$88.56 - 0.648P_1$	$85.63 \le P_1 \le 86.67$			

The above parametric solutions are also illustrated in Figure 6.2. Notice that figure 6.2(b) corresponds to the case of maximizing the production of P1. The figure was rotated to enable the combination of those two results as shown in Figure 6.3(a).



(a) (b)

Figure 6.2 Illustration of the parametric solution



(a) (b)

Figure 6.3 Combined nonconvex production capacity region and its convex hull

As shown in Figure 6.3(a), the derived production capacity region is nonconvex. To incorporate this information into the planning model (6.1), the following constraints with auxiliary binary variables can be formulated based on big-*M* relaxation:

$$\sum_k y_k = 1 \tag{6.5a}$$

$$P_2 \le 1.6875 P_1 + M(1 - y_1) \tag{6.5b}$$

$$0 - M(1 - y_1) \le P_1 \le 52 + M(1 - y_1) \tag{6.5c}$$

where M is a big positive constant and y_k are the binary variables. Notice that constraints (6.5b)-(6.5c) are just for the first segment of the capacity boundary. Similar constraints can be also formulated for all of the other segments.

On the other hand, to avoid the incorporation of binary variables and mixed integer linear constraints into planning model (6.1), we can evaluate the convex hull of the derived nonconvex production capacity region as shown in figure 6.3(b) which can be described by the following linear inequalities:

$$\begin{bmatrix} 0 & 0.02550 \\ 0.01519 & 0.02795 \\ 0.03988 & 0.00004 \\ 0.04993 & 0.01833 \\ 0.05034 & 0.01629 \\ -0.03047 & 0.01806 \\ -0.00000 & -0.02060 \end{bmatrix} \begin{bmatrix} P_1 \\ P_2 \end{bmatrix} \le \begin{bmatrix} 2.23785 \\ 3.29274 \\ 3.45814 \\ 4.96497 \\ 4.89101 \\ 0 \\ 0 \end{bmatrix}$$

Based on the nonconvex and convex representation of the production capacity, the rolling horizon algorithm is solved again and the following results can be obtained as shown in Table 6.5.

Table 6.5 Rolling horizon solution with production capacity model from parametric solution

	With convex model	With nonconvex model
Production	$\begin{bmatrix} 43.1 & 86.7 & 70.2 & 59.8 & 70.2 \\ 50 & 25.5 & 79.6 & 85.3 & 79.6 \end{bmatrix}$	$\begin{bmatrix} 43.1 & 86.7 & 70.2 & 59.8 & 70.2 \\ 50 & 25.5 & 79.6 & 85.3 & 79.6 \end{bmatrix}$
Inventory	$\begin{bmatrix} 43.2 & 69.8 & 0 & 29.8 & 0 \\ 0 & 25.5 & 50.1 & 70.4 & 0 \end{bmatrix}$	$\begin{bmatrix} 43.1 & 69.8 & 0 & 29.8 & 0 \\ 0 & 25.5 & 50.1 & 70.3 & 0 \end{bmatrix}$
Backorder	$\begin{bmatrix} 0 & 0 & 0.007 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.004 \end{bmatrix}$	$\begin{bmatrix} 0 & 0 & 0.003 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.009 \end{bmatrix}$
Production cost	6423.3	6423.3
Inventory cost	2886.9	2886.7
Backorder cost	1.1	1.2
Total cost	**9311.3**	**9311.2**

Comparing the results in Tables 6.3 and 6.5, it is observed that the solution quality is greatly improved with the production capacity model derived from parametric programming technique, i.e., the more accurate production capacity information cause the cost reduction. Furthermore, for the two cases in Table 6.5, there are not big differences in the solution obtained. The reason is that the two different capacity representations actually do not differ too much as shown in Figure 6.3(b).

6.3.3 Capacity model derivation through process network decomposition

For a process network with more than two products, the exact production capacity region has to be described in a higher dimension space. Furthermore, the exact nonconvex boundary of the capacity region will involve more facets. To exactly describe the nonconvex region, large number of auxiliary binary variables will be

necessary. When those binary variables are incorporated into the planning model, the resulted planning problem will be mixed integer linear problem with a large number of binary variables which is hard to solve.

Moreover, for the parametric programming algorithm, although it can be performed off-line, the required computation effort increases significantly for higher dimension problem (i.e., the number of products is large) since it is proportional to the number of critical regions (Li & Ierapetritou, 2007b). To avoid the large computational complexity of parametric programming method for high dimensionality problems, we can use the following heuristic network decomposition strategy and apply the parametric programming method on the sub-networks which involve relative small number of products. With a decomposed network, the complexity of the scheduling model for the sub-network is also reduced, which will also facilitate the solution of the parametric problem.

Based on the STN representation of the production process, we propose the following guiding rules for the process network decomposition:

1) identify key connecting intermediate products in the process network;
2) identify sub-networks such that the resulted scheduling model is able to solve efficiently.

The above network decomposition is rather a heuristic strategy, which should be studied based on specific problem but the basic principle is to generate scheduling problems which can be solved more efficiently by decreasing the number of parameters (products) appeared in the scheduling problem.

The different production capacity relationships that can be obtained from this kind of decomposition include:

1) the production capacity information between any two products in each sub-network;
2) the production capacity information between the connecting products;
3) the production capacity information between certain group of products.

Notice that the production capacity information between certain products in a sub-network can be viewed as the projection of the exact production feasibility space onto certain dimensions of the original feasible production space. Thus the parametric solution between certain groups of products provides valid overestimation of the feasibility space.

It should be pointed out that in the above network decomposition procedure, the same processing unit can appear in different sub-networks if two tasks sharing the same unit are separated. Thus the production capacity is actually overestimated through the above decomposition strategy and the derived production

capacity region is the relaxation of the actual production capacity region. So the capacity information derived from the network decomposition strategy provides a valid approximation of the production feasibility and can be incorporated into the planning and scheduling integration model. The process network decomposition idea will be shown through case studies in the next section.

6.4 Case studies

In this section, two examples which involve more complex process networks than the motivation example are studied to test the proposed solution framework. All the computations in this work are performed on a PC with 2.8GHz CPU and 1Gb RAM and the MILP problems are solved using CPLEX 10.1 solver in GAMS.

Example 1.

In the following example derived from (Kondili, 1987), four products are produced through eight tasks from three feeds in the process. Six different units are required for the whole process. The STN representation of this process is shown in Figure 6.4 and the problem data can be found in (Li & Ierapetritou, 2009).

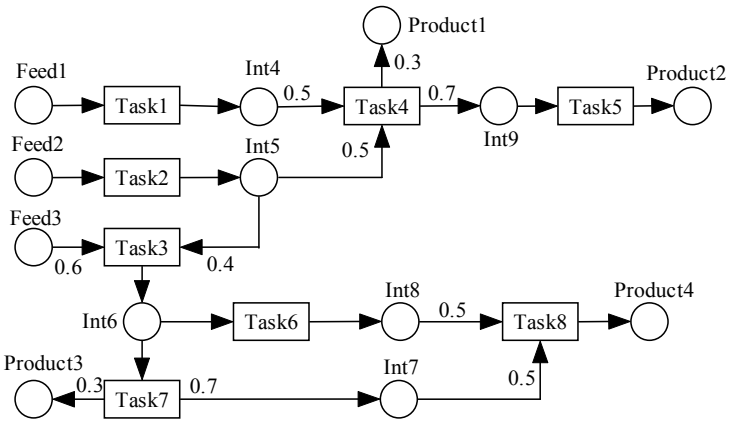


Figure 6.4 STN representation of example 1

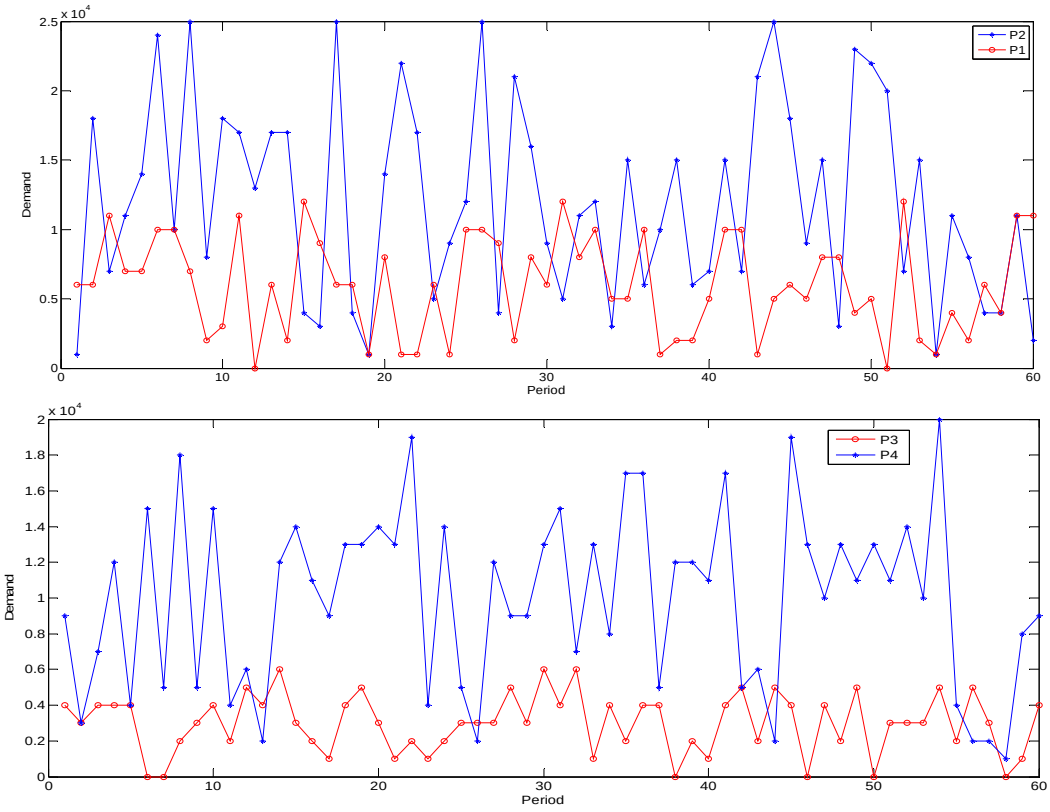


Figure 6.5 Demand data

In this example, we are studying a two-month production planning and scheduling integration problem. The time horizon is divided into 60 planning periods and in every planning period a 24-hour scheduling problem is considered. The demand data are illustrated in Figure 6.5. For every scheduling problem, 15 event points are used, which results in a MILP model with 2579 constraints, 1361continuous, and 768 binary variables. To decrease the solution complexity of the short-term scheduling problem, here we first decompose the process network into two sub-networks as shown in Figure 6.6. Notice that the two sub-networks are connected through intermediate product Int5. Based on this decomposition, each sub-network only involves two products and the complexity of the scheduling problem is reduced.

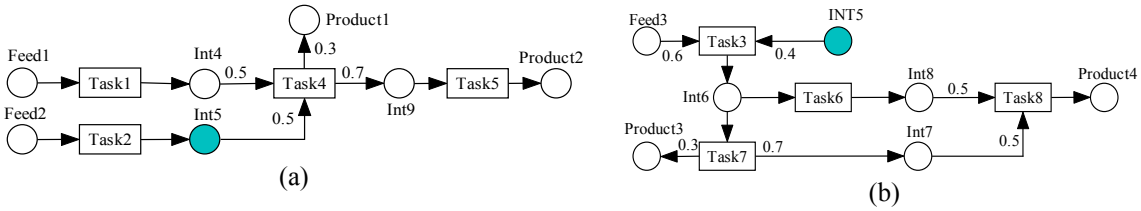


Figure 6.6 Sub-network 2 (assume INT 5 with infinite supply)

Before we apply the parametric programming method to derive the capacity model, simple inequalities representing the upper bound for the ratio between two products can be derived as follows: in the first sub-network, for every 30 units of P1 been processed, the maximum amount of P2 that can be produced is 70, so we have the valid inequality: $P_2 \le 7P_1/3$; similarly, we have $P_4 \le 2\times(7P_3/3)$ from the second sub-network.

To derive more accurate production capacity information, parametric programming method is applied next. For the first sub-network, we solve the problem of maximizing production of P2 (P1) with the production of P1 (P2) as parameter. By aggregating all the parametric solutions, we can identify the production capacity boundary between products P1 and P2 as shown in Figure 6.7(a). The convex hull is also plotted on the same figure using red color. Similarly, the production capacity region between P3 and P4 is shown in Figure 6.7(b) (the inner boundary of the grey area). Furthermore, the convex hulls of those nonconvex capacity regions are also shown as the outside red boundaries. From those results, it can be observed that the parametric solution based solution contain more accurate capacity information than those simple inequalities derived from direct mass balance calculation.

The parametric solution for the sub-networks only provides production capacity information between products in those sub-networks. To describe the production feasibility between different sub-networks, we can further apply similar method in the original process network, by grouping the products. For example, we can set the scheduling objective as maximizing the total production of P1 and P2 and set the total production of P3 and P4 as uncertain parameter. In this way we can identify the production capacity information between the two sub-networks as shown in Figure 6.7(c).

Table 6.6 illustrates the solution of the rolling horizon method for two different cases: one is the case without any capacity constraints and the other is the case with convex hull capacity model in the planning problem (notice that since there is no big difference between the nonconvex capacity region and convex hull in Figure 6.7, we directly apply the convex hull model to avoid the addition of auxiliary binary variables and mixed integer linear constraints).

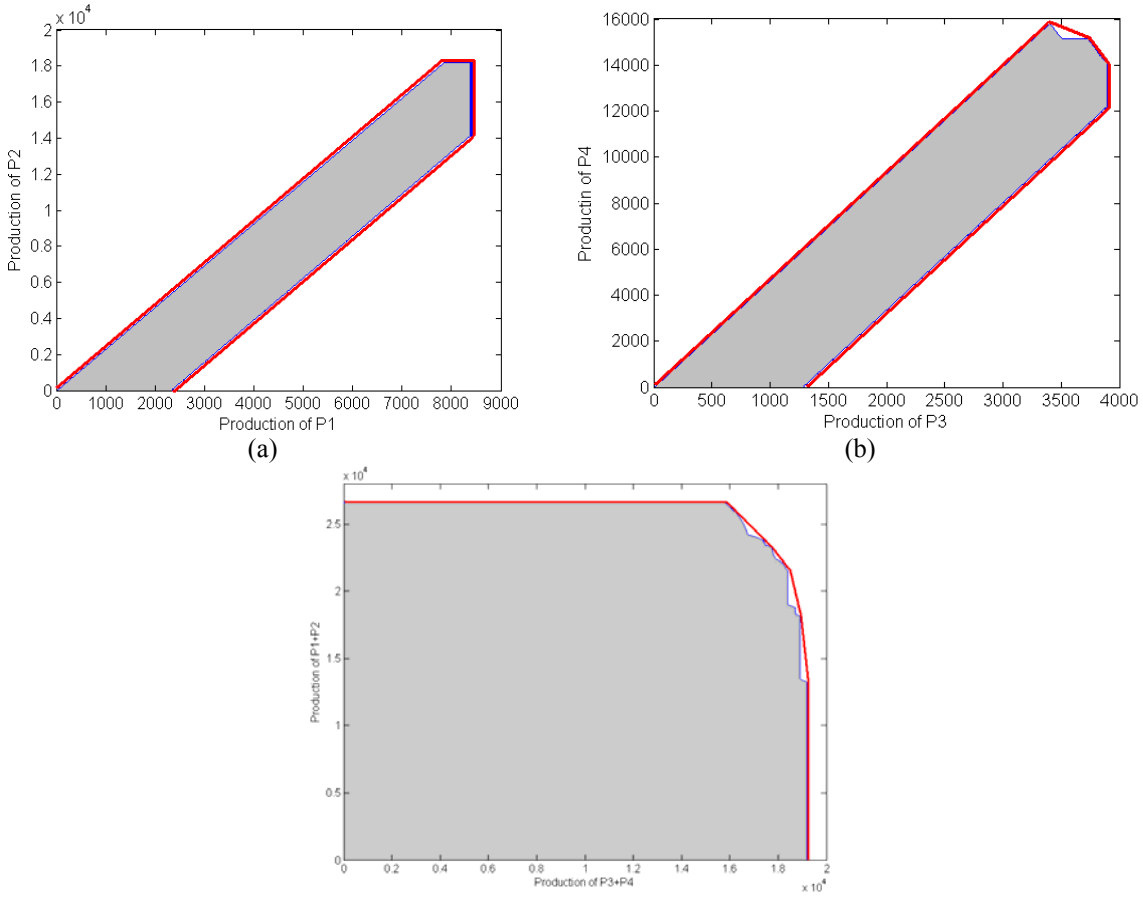


Figure 6.7 Production capacity region and the convex hull
(a) Between P1 and P2; (b) Between P3 and P4; (c) Between P1+P2 and P3+P4

Table 6.6 Solution of the example 1

	Without capacity model	With convex model of the capacity constraints
Production cost	3,295,638	3,383,932
Inventory cost	556,053	1,900,957
Backorder cost	26,187,860	11,770
Total cost	**30,039,550**	**5,296,659**
CPU time (sec)	360	460

It can be observed from the results that the rolling horizon method can solve the problem relatively fast. Comparing the quality of the final solution, we can see that the convex capacity model greatly improve the final solution's quality. The production targets are shown in Figure 6.8 (without capacity constraints) and Figure 6.9 (with convex capacity constraints).

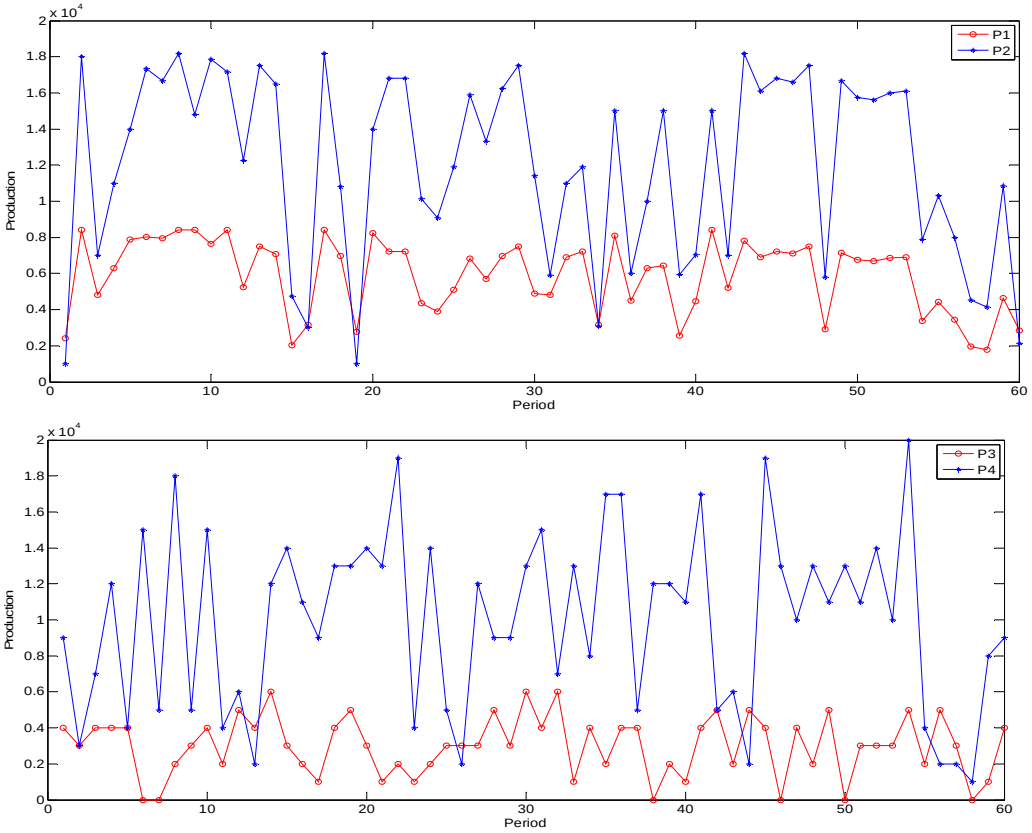


Figure 6.8 Production target solution (without capacity constraints)

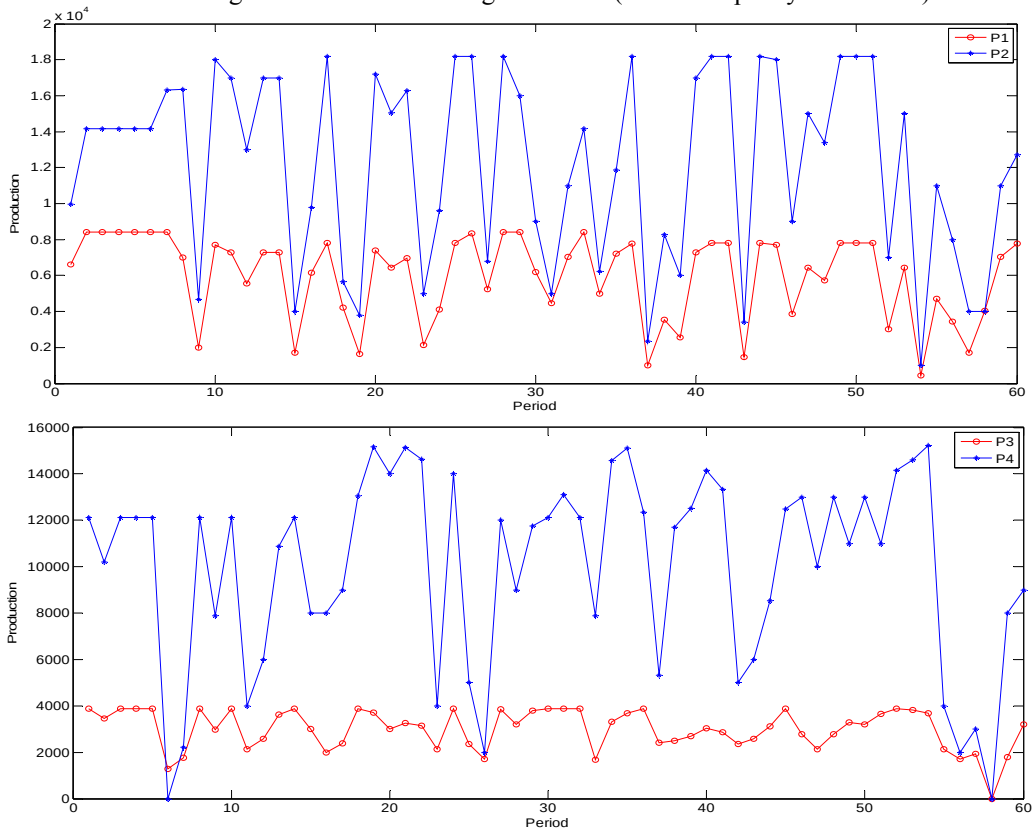


Figure 6.9 Production target solution (with capacity constraints)

From the results in Figures 6.8 and 6.9, it can be observed that the production targets in the solution with capacity constraints is always within the feasible production capacity region, e.g., the production amounts of P3 are always below 4000. But the production target solutions of the case without capacity constraints can violate the production feasibility, e.g., for some periods the production of P3 is more than 4000. Those infeasible production targets will result in more cost since much more backorder is resulted because of the "overestimation" of production capacity. This can be further illustrated through the following backorder amount profile. For the case without capacity constraints, the backorder amounts in the solution for P1 and P2 are shown in Figure 6.10 (the backorder amounts of P3 and P4 are all zeros). For the case with capacity constraints, the backorder of P1, P2 and P4 at all the periods are zero, only product P3 has a backorder amount of 117.7 at the first period. This verified the difference between the final total cost for the two different cases (with and without capacity constraints) studied in this problem.

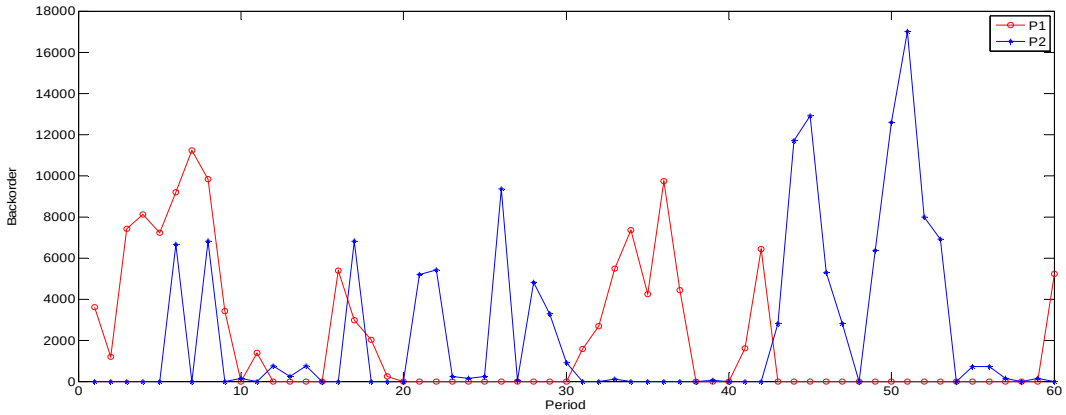


Figure 6.10 Backorder amount in the solution (without capacity constraints)

Example 2

The process in this example can be found in (Kallrath, 2002) and it is a benchmark problem which has stimulated the development of algorithms for scheduling problems in process industry. The STN is shown in Figure 6.11 for this example which is more complicated compared to the previous example. To decrease the scheduling model complexity, we assume no cleaning restrictions in this study.

In this example, we are studying a three-month production planning and scheduling integration problem, where 30 planning periods are considered, and in every planning period a 96-hour scheduling problem is studied. Notice that we have select 11 event points for every scheduling problem.

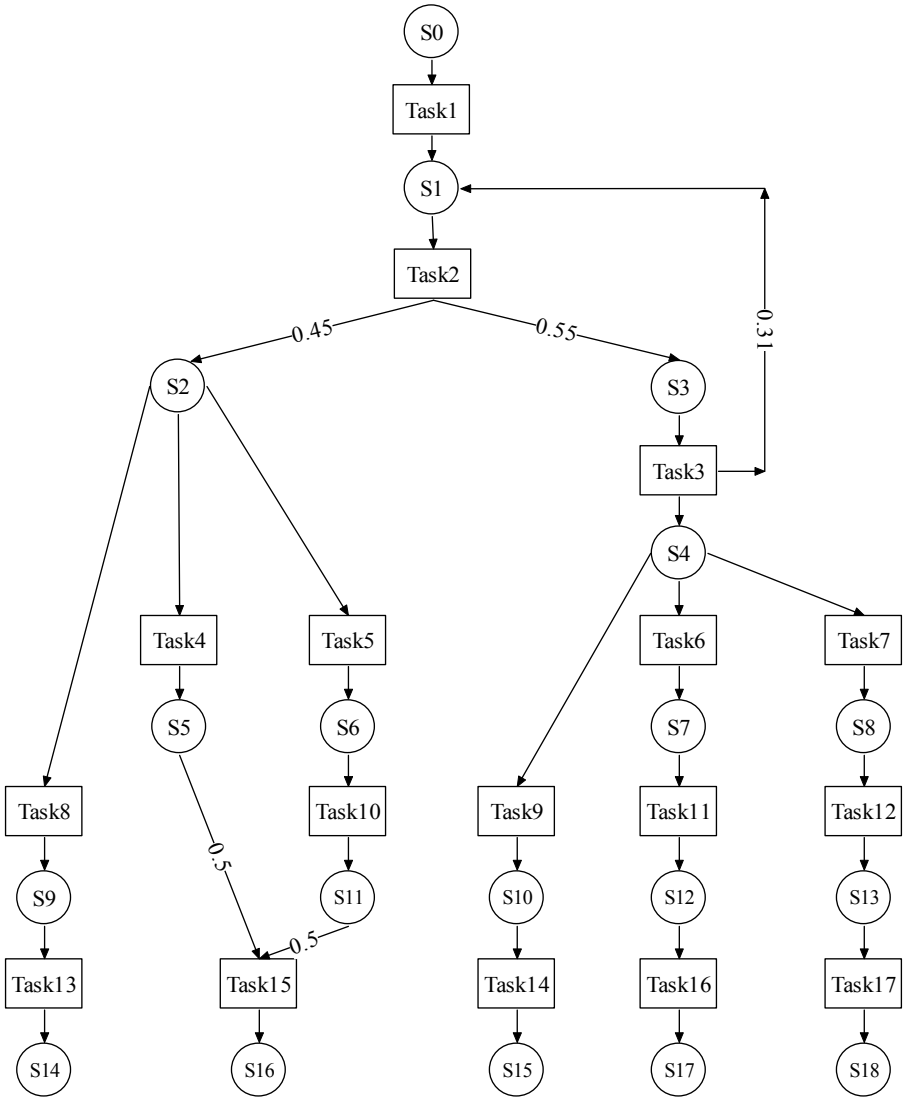


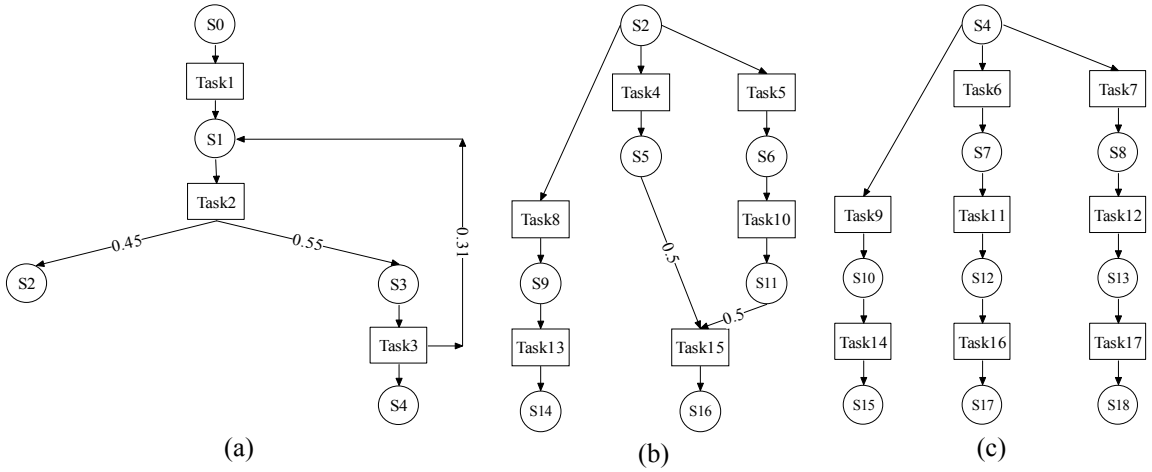
Figure 6.11 STN representation of example 2

Since the given process network in this example is complex, the resulted scheduling problem need significant computation efforts. To increase the computation efficiency for the proposed algorithm, we decompose the original process network and generate three sub-networks as shown in Figure 6.12. The decomposition is performed by identifying that S2 and S4 are two connecting intermediates which separate the whole network into three major parts. Thus that the solution complexity of the scheduling problem for the sub-networks can be decreased comparing to the original scheduling model for the whole network. Detailed model statistics can be found in Table 6.7.

Table 6.7 Model statistics

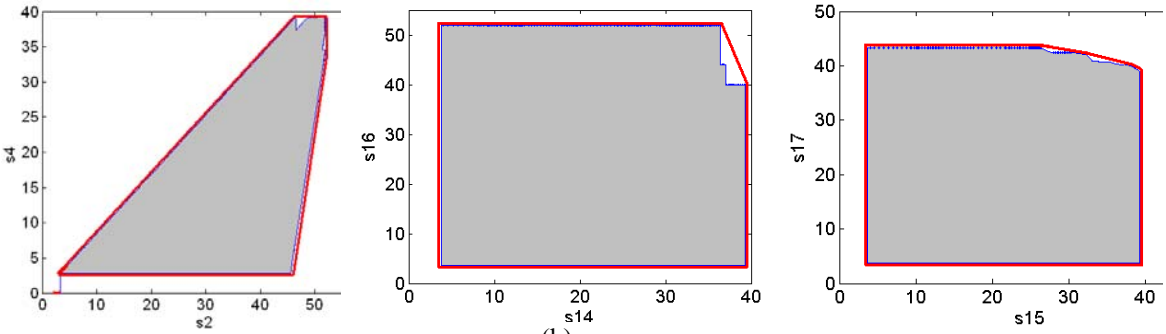
	Constraints	Continuous variables	Binary variables
Whole network	5361	2685	1683
Sub-network 1	559	254	99
Sub-network 2	1488	738	396
Sub-network 3	2439	1057	528

To decrease the computation complexity of solving large scale scheduling problem, the following process network decomposition idea can be applied:



(a) (b) (c)

Figure 6.12 Process network decomposition for example 2



(a) (b) (c)

Figure 6.13 Production capacity information

Base on the decomposed process networks, several sets of production capacity information can be derived. For example, Figure 6.13(a) represents the production capacity between S2 and S4 which is derived from first sub-network; the production capacity information between S14 and S16 is shown in Figure 6.13(b); the production capacity information between S15 and S17 in the third network is shown in Figure 6.13(c). The capacity information between S15 and S18, S17 and S18 are not listed considering the length of the chapter.

Notice that to apply the above production capacity information into the rolling horizon planning model. We need to incorporate the following constraints:

$$f_1(P_{S2}, P_{S4}) \le 0$$
$$f_2(P_{S14}, P_{S16}) \le 0$$
$$f_3^a(P_{S15}, P_{S17}) \le 0, f_3^b(P_{S15}, P_{S18}) \le 0, f_3^c(P_{S17}, P_{S18}) \le 0$$
$$P_{S14} + P_{S16} \le P_{S2}$$
$$P_{S15} + P_{S17} + P_{S18} \le P_{S4}$$

where f_2, f_3 represent the convex hull constraints for the final products derived from sub-networks, respectively. The constraints f_1 and the last two constraints are used to describe the capacity information between the product groups in the second and the third sub-networks which are connected through the intermediate products S2 and S4.

Table 6.8 Solution of the example 2

	Without capacity model	With convex model of the capacity constraints
Production cost	68,352	69,255
Inventory cost	2,654	3,549
Backorder cost	122,693	6,640
Total cost	**193,699**	**79,444**
CPU time (sec)	1353	1429

Table 6.8 shows the results of the rolling horizon method on this example for two cases with and without capacity model. As shown the production capacity model derived from the proposed strategy greatly increased the quality of the solution with no big difference on the computation time for the rolling horizon method. It should be also noticed here that since all of the parametric solutions can be computed off-line and the resulted capacity information, the proposed method provides an efficient way to improve the solution quality of the integrated planning and scheduling optimization in the rolling horizon framework.

6.5 Summary

To ensure the consistency between production planning and scheduling, an integrated decision making is necessary. Among the various types of aggregation, relaxation and decomposition methods, rolling horizon method has received wide study and also realistic applications in the industry. In this chapter, we study the effect of production capacity information on the final solution's quality. It has been shown that the solution

quality can be greatly improved by incorporating the production capacity model representing the production capacity of the short term scheduling problem into the planning model. The total cost can be greatly reduced mainly due to decreased backorder amount since the planning model can “predict” the future possible backorder based on production capacity estimation. To derive accurate production capacity information, we proposed a parametric programming based method, which generates the exact production feasibility information based on the short-term scheduling formulation. To reduce the computation complexity for complex processes with a number of products, a heuristic network decomposition strategy is proposed. Cases studies prove that the method can further improve the quality of the final solution comparing to the simple production capacity constraints derived by production recipe and mass balance. Finally, it is worth to point out the proposed solution framework can be also applied to address the long-term and mid-term scheduling problem within a rolling horizon framework.

Bibliography

[1] Acevedo, J., & Pistikopoulos, E. N. (1997). A multiparametric programming approach for linear process engineering problems under uncertainty. Industrial and Engineering Chemistry Research, 36, 717-728.

[2] Acevedo, J., & Salgueiro, M. (2003). An efficient algorithm for convex multiparametric nonlinear programming problems. Industrial and Engineering Chemistry Research, 42, 5883 -5890.

[3] Andreani, R., Birgin, E. G., Martínez, J. M., & Schuverdt, M. L. (2008). On Augmented Lagrangian methods with general lower-level constraints. SIAM Journal on Optimization, 18, 1286-1309.

[4] Balasubramanian, J., & Grossmann, I. E. (2002). A novel branch and bound algorithm for scheduling flowshop plants with uncertain processing times. Computers and Chemical Engineering, 26, 41-57.

[5] Balasubramanian, J., & Grossmann, I. E. (2003). Scheduling optimization under uncertainty-an alternative approach. Computers and Chemical Engineering, 27, 469-490.

[6] Bassett, M. H., Dave, P., Doyle, F. J., Kudva, G. K., Pekny, J. F., Reklaitis, G. V., Subrahmanyam, S., Miller, D. L., & Zentner, M. G. (1996). Perspectives on model based integration of process operations. Computers and Chemical Engineering, 20, 821-844.

[7] Bassett, M. H., Pekny, J. F., & Reklaitis, G. V. (1996). Decomposition techniques for the solution of large-scale scheduling problems. AICHE Journal, 42, 3373-3387.

[8] Bemporad, A., Morari, M., Dua, V., & Pistikopoulos, E. N. (2002). The explicit linear quadratic regulator for constrained systems. Automatica, 38, 3-20.

[9] Ben-Tal, A., & Nemirovski, A. (1999). Robust solutions to uncertain programs. Operations Research Letters, 25, 1-13.

[10] Bertsekas, D. P. (2003). Nonlinear Programming (2nd ed. ed.). Belmont, Massachusetts: Athena Scientific.425

[11] Bertsekas, D. P., & Tsitsiklis, J. N. (1989). Parallel and Distributed Computation. Englewood Cliffs, New Jersey: Prentice-Hall.422

[12] Bertsimas, D., & Sim, M. (2003). Robust Discrete optimization and Network Flows. Math. Prog., 98, 49-71.

[13] Birgin, E. G., Floudas, C. A., & Martínez, J. M. (2009). Global minimization using an Augmented Lagrangian method with variable lower-level constraints. Math. Program., Ser. A, DOI 10.1007/s10107-10009-10264-y.

[14] Bonfill, A., Bagajewicz, M., Espuna, A., & Puigjaner, L. (2004). Risk Management in the Scheduling of Batch Plants under Uncertain Market Demand. Industrial and Engineering Chemistry Research, 43, 741-750.

[15] Bonfill, A., Espuna, A., & Puigjaner, L. (2005). Addressing Robustness in Scheduling Batch Processes with Uncertain Operation Times. Industrial and Engineering Chemistry Research, 44, 1524-1534.

[16] Borrelli, E., Bemporad, A., & Morari, M. (2003). A geometric algorithm for multi-parametric linear programming. Journal of Optimization Theory and Applications, 118, 515-540.

[17] Carpentier, P., Cohen, G., Culioli, J. C., & Renaud, A. (1996). Stochastic optimization of unit commitment: A new decomposition framework. IEEE Transactions on Power Systems, 11, 1067-1073.

[18] Castro, P., Barbosa-Povoa, A., & Matos, H. (2003). Optimal Periodic Scheduling of Batch Plants Using RTN-based Discrete and Continuous-time Formulations: A Case Study Approach. Ind. Eng. Chem. Res., 42, 3346-3360.

[19] Chandrasekaran, R., Kabadi, S. N., & Sridhar, R. (1998). Integer Solution for Linear Complementarity Problem. Mathematics of Operations Research, 23, 390-402.

[20] Cott, B. J., & Macchietto, S. (1989). Minimizing the effects of batch process variability using online schedule modification. Computers and Chemical Engineering, 13, 105-113.

[21] Dimitriadis, A. D., Shah, N., & Pantelides, C. C. (1997). RTN-based rolling horizon algorithms for medium term scheduling of multipurpose plants. Computers & Chemical Engineering, 21, S1061-S1066.

[22] Dua, V., Bozinis, N. A., & Pistikopoulos, E. N. (2002). A multiparametric programming approach for mixed-integer quadratic engineering problems. Computers and Chemical Engineering, 26, 715-733.

[23] Dua, V., & Pistikopoulos, E. N. (1999). Algorithms for the solution of multiparametric mixed-integer nonlinear optimization problems. Industrial and Engineering Chemistry Research, 38, 3976 - 3987.

[24] Dua, V., & Pistikopoulos, E. N. (2000). An algorithm for the solution of multiparametric mixed integer linear programming problems. Ann. Oper. Res., 99, 123-139.

[25] El-Ghaoui, L., Oustry, F., & Lebret, H. (1998). Robust solutions to uncertain semidefinite programs. SIAM J. Optim., 9, 33-52.

[26] Elliott, M. (2000). Advanced planning and scheduling software. IIE Solutions, 32, 48-56.

[27] Equi, L., Gallo, G., Marziale, S., & Weintraub, A. (1997). A combined transportation and scheduling problem. European Journal of Operational Research, 97, 94-104.

[28] Erdirik-Dogan, M., & Grossmann, I. E. (2006). A Decomposition Method for the Simultaneous Planning and Scheduling of Single-Stage Continuous Multiproduct Plants. Ind. Eng. Chem. Res., 45, 299-315.

[29] Floudas, C. A., & Lin, X. (2004). Continuous-time versus discrete-time approaches for scheduling of chemical processes: A review. Computers and Chemical Engineering, 28, 2109-2129.

[30] Grossmann, I. E., Van den Heever, S. A., & Harjunkoski, I. (2002). Discrete optimization methods and their role in the integration of planning and scheduling. AIChE Symp. Ser., 98, 150.

[31] Grossmann, I. E., & Westerberg, A. W. (2000). Research challenges in process system engineering. AICHE Journal, 46, 1700-1703.

[32] Gupta, A., & Maranas, C. D. (1999). A Hierarchical Lagrangean Relaxation Procedure for Solving Midterm Planning Problem. Ind. Eng. Chem. Res., 38, 1937.

[33] Hamdi, A., Mahey, P., & Dussault, J. P. (1997). A new decomposition method in nonconvex programming via a separable augmented Lagrangian. In Recent advances in optimization (Vol. 452, pp. 90-104).412

[34] Honkomp, S. J., Mockus, L., & Reklaitis, G. V. (1997). Robust scheduling with processing time uncertainty Computers and Chemical Engineering, 21, S1055-S1060

[35] Huercio, A., Espuña, A., & Puigjaner, L. (1995). Incorporating on-line scheduling strategies in integrated batch production control. Computers and Chemical Engineering, 19, S609-S615.

[36] Hugo, A., & Pistikopoulos, S. (2005). Long-range process planning under uncertainty via parametric programming. European Symposium on Computer-Aided Process Engineering, 15, 127-132.

[37] Ierapetritou, M. G., & Floudas, C. A. (1998). Effective continuous-time formulation for short-term scheduling. 1. Multipurpose batch processes. Industrial & engineering chemistry research, 37, 4341-4359.

[38] Ierapetritou, M. G., & Pistikopoulos, E. N. (1996). Global optimization for stochastic planning, scheduling and design problems. In I. E. Grossmann (Ed.), Global optimization in engineering design (pp. 231-287). Dordrecht: Kluwer Academic Publishers.9

[39] Janak, S. L., & Floudas, C. A. (2006). Production Scheduling of a Large-Scale Industrial Batch Plant. II. Reactive Scheduling. Industrial and Engineering Chemistry Research, 45, 8253-8269.

[40] Janak, S. L., Floudas, C. A., Kallrath, J., & Vormbrock, N. (2006). Production scheduling of a large-scale industrial batch plant. I. Short-term and medium-term scheduling. Industrial and Engineering Chemistry Research, 45, 8234-8252.

[41] Janak, S. L., Lin, X., & Floudas, C. A. (2007). A new robust optimization approach for scheduling under uncertainty: II. Uncertainty with known probability distribution. Computers and Chemical Engineering, 31, 171-195.

[42] Jia, Z., & Ierapetritou, M. G. (2004). Short-Term Scheduling under Uncertainty Using MILP Sensitivity Analysis. Industrial and Engineering Chemistry Research, 43, 3782-3791.

[43] Jia, Z., & Ierapetritou, M. G. (2007). Generate Pareto optimal solutions of scheduling problems using normal boundary intersection technique. Computers and Chemical Engineering, 31, 266-280.

[44] Johansen, T. A. (2002). On multi-parametric nonlinear programming and explicit nonlinear model predictive control. In 41th IEEE Conference on Decision and Control (pp. 2768-2273). Las Vegas, Nevada, USA.

[45] Jones, C. N., & Morrari, M. (2006). Multiparametric Linear Complementarity Problems. 45th IEEE Conference on Decision and Control, 5687-5692.

[46] Kallrath, J. (2002). Planning and scheduling in the process industry. OR Spectrum, 24, 219-250.

[47] Kanakamedala, K. B., Reklaitis, G. V., & Venkatasubramanian, V. (1994). Reactive schedule modification in multipurpose batch chemical plants. Industrial and Engineering Chemistry Research, 30, 77-90.

[48] Kondili, E. (1987). Optimal scheduling of Batch Processes. Imperial College London, London, U.K.

[49] Kondili, E., Pantelides, C. C., & Sargent, R. W. H. (1993). A general algorithm for short-term scheduling of batch operations. I. MILP formulation. Computers and Chemical Engineering, 17, 211-227.

[50] Kouvelis, P., Daniels, R. L., & Vairaktarakis, G. (2000). Robust scheduling of a two-machine flow shop with uncertain processing times. IIE. Trans., 32, 421.

[51] Kreipl, S., & Pinedo, M. (2004). Planning and Scheduling in Supply Chains: An Overview of Issues in Practice. Production and Operations Management, 13, 77-92.

[52] Li, Y., Lu, Z., & Michalek, J. J. (2008). Diagonal quadratic approximation for parallelization of analytical target cascading. ASME Journal of Mechanical Design, 130, 051402.

[53] Li, Z., & Ierapetritou, M. G. (2007a). A New Methodology for the General Multiparametric Mixed-Integer Linear Programming (MILP) Problems. Industrial & engineering chemistry research, 46, 5141-5151.

[54] Li, Z., & Ierapetritou, M. G. (2007b). Process scheduling under uncertainty using multiparametric programming. AICHE Journal, 53, 3183-3203.

[55] Li, Z., & Ierapetritou, M. G. (2009). Integrated production planning and scheduling using a decomposition framework. Chemical Engineering Science, 64, 3585-3597.

[56] Lin, X., Floudas, C. A., Modi, S., & Juhasz, N. M. (2002). Continuous-time optimization approach for medium-range production scheduling of a multiproduct batch plant. Industrial & engineering chemistry research, 41, 3884-3906.

[57] Lin, X., Janak, S. L., & Floudas, C. A. (2004). A new robust optimization approach for scheduling under uncertainty: I. bounded uncertainty. Computers and Chemical Engineering, 28, 1069-1085.

[58] Maravelias, C. T., & Grossmann, I. E. (2006). On the relation of continuous- and discrete-time state-task network formulations. AICHE Journal, 52, 843-849.

[59] Maravelias, C. T., & Sung, C. (2008). Integration of production planning and scheduling: overview, challenges and opportunities. Proceedings Foundations of Computer-Aided Process Operations (FOCAPO 2008), 13-22.

[60] Méndez, C. A., & Cerdá, J. (2003). Dynamic scheduling in multiproduct batch plants. Computers and Chemical Engineering, 27, 1247-1259.

[61] Méndez, C. A., & Cerdá, J. (2004). An MILP framework for batch reactive scheduling with limited discrete resources. Computers and Chemical Engineering, 28, 1059-1068.

[62] Méndez, C. A., Cerdá, J., Grossmann, I. E., Harjunkoski, I., & Fahl, M. (2006). State-of-the-art review of optimization methods for short-term scheduling of batch processes. Computers and Chemical Engineering, 30, 913-946.

[63] Mulvey, J., Vanderbei, R., & Zenios, S. (1995). Robust optimization of large scale systems. Operations Research, 43, 264.

[64] Munawar, S. A., & Gudi, R. D. (2005). A Multilevel, Control-Theoretic Framework for Integration of Planning, Scheduling, and Rescheduling. Ind. Eng. Chem. Res., 44, 4001-4021.

[65] Orçun, S., Altinel, K., & Hortaçsu, Ö. (1996). Scheduling of batch processes with operational uncertainties. Computers and Chemical Engineering, 20, S1191-S1196.

[66] Padhy, N. P. (2004). Unit commitment-a bibliographical survey. Power Systems, IEEE Transactions on, 19, 1196-1205.

[67] Papageorgiou, L. G., & Pantelides, C. C. (1996). Optimal campaign planning scheduling of multipurpose batch semicontinuous plants. 2. A mathematical decomposition approach. Industrial & engineering chemistry research, 35, 510-529.

[68] Pardalos, P. M., & Nagurney, A. (1990). The Integer Linear Complementarity Problem. International Journal of Computer Mathematics, 31, 205-214.

[69] Petkov, S. B., & Maranas, C. D. (1997). Multiperiod Planning and Scheduling of Multiproduct Batch Plants under Demand Uncertainty. Industrial and Engineering Chemistry Research, 36, 4864-4881.

[70] Petrovic, D., & Duenas, A. (2006). A fuzzy logic based production scheduling/rescheduling in the presence of uncertain disruptions. Fuzzy Sets and Systems, 157, 2273-2285.

[71] Pistikopoulos, E. N. (1995). Uncertainty in process design and operations. Computers and Chemical Engineering, 19, S553-S563.

[72] Pistikopoulos, E. N., & Dua, V. (1998). Planning under uncertainty: A Parametric Programming approach. In J. F. Penky & C. E. Blau (Eds.), 3rd International Conference on Foundations of Computer-Aided Process Operations (pp. 164-169).

[73] Pistikopoulos, E. N., Georgiadis, M. C., & Dua, V. (2007). Process Systems Engineering: Volume 1: Multiparametric Programming: Wiley-VCH.364

[74] Qi, L., & Wei, Z. (2000). On the constant positive linear dependence condition and its application to SQP methods. SIAM Journal on Optimization, 10, 963-981.

[75] Rodrigues, M. T. M., Gimeno, L., Passos, C. A. S., & Campos, M. D. (1996). Reactive scheduling approach for multipurpose chemical batch plants. Computers and Chemical Engineering, 20, S1215-S1220.

[76] Rosen, J. B. (1990). Minimum Norm Solution to the Linear Complementarity Problem. In Functional Analysis, Optimization, and Mathematical Economics: Oxford University Press.333

[77] Roslöf, J., Harjunkoski, I., Björkqvist, J., Karlsson, S., & Westerlund, T. (2001). An MILP-based reordering algorithm for complex industrial scheduling and rescheduling. Computers and Chemical Engineering, 25, 821-828.

[78] Ruiz, D., Cantón, J., Nougués, J. M., Espuña, A., & Puigjaner, L. (2001). On-line fault diagnosis system support for reactive scheduling in multipurpose batch chemical plants. Computers and Chemical Engineering, 25, 829-837.

[79] Ruszczynski, A. (1995). On convergence of an augmented Lagrangian decomposition method for sparse convex optimization. Mathematics of Operations Research, 20, 634-656.

[80] Ryu, J. H., & Pistikopoulos, E. N. (2007). A novel approach to scheduling of zero-wait batch processes under processing time variations. Computers and Chemical Engineering, 31, 101-106.

[81] Sand, G., & Engell, S. (2004). Modeling and solving real-time scheduling problems by stochastic integer programming. Computers and Chemical Engineering, 28, 1087-1103.

[82] Sand, G., Engell, S., Märkert, A., Schultz, R., & Schultz, C. (2000). Approximation of An Ideal Online Scheduler for A Multiproduct Batch Plant. Comput. Chem. Eng., 24, 361-367.

[83] Sanmartí, E., Espuña, A., & Puigjaner, L. (1997). Batch production and preventive maintenance scheduling under equipment failure uncertainty Computers & Chemical Engineering, 21, 1157-1168

[84] Schilling, G., & Pantelides, C. C. (1999). Optimal periodic scheduling of multipurpose plants. Computers & Chemical Engineering, 23, 635-655.

[85] Seron, M., De Dona, J. A., & Goodwin, G. C. (2000). Global analytical model predictive control with input constraints. In 39th IEEE Conference on Decision and Control (pp. 154-159). Sydney, Australia.

[86] Shah, N. (2005). Process industry supply chains: Advances and challenges. Computers & Chemical Engineering, 29, 1225-1235.

[87] Shaik, M. A., Floudas, C. A., Kallrath, J., & Pitz, H.-J. (2007). Production scheduling of a large-scale industrial continuous plant: Short-term and medium-term scheduling. COMPUTER AIDED CHEMICAL ENGINEERING, 24, 613-618.

[88] Shaik, M. A., Floudas, C. A., Kallrath, J., & Pitz, H.-J. (2009). Production scheduling of a large-scale industrial continuous plant: Short-term and medium-term scheduling. Computers & Chemical Engineering, 33, 670-686.

[89] Soyster, A. L. (1973). Convex programming with set-inclusive constraints and applications to inexact linear programming. Operations Research, 21, 1154-1157.

[90] Sung, C., & Maravelias, C. T. (2007). An attainable region approach for effective production planning of multi-product processes. AICHE Journal, 53, 1298-1315.

[91] Tosserams, S., Etman, L. F. P., & Rooda, J. E. (2008). Augmented Lagrangian coordination for distributed optimal design in MDO. Int. J. Numer. Meth. Engng., 73, 1885-1910.

[92] Verderame, P. M., & Floudas, C. A. (2008). Integrated operational planning and medium-term scheduling of a large-scale industrial batch plants. Ind. Eng. Chem. Res, 47, 4845-4860.

[93] Vin, J. P., & Ierapetritou, M. G. (2000). A new approach for efficient rescheduling of multiproduct batch plants. Industrial and Engineering Chemistry Research, 39, 4228-4238.

[94] Vin, J. P., & Ierapetritou, M. G. (2001). Robust short-term scheduling of multiproduct batch plants under demand uncertainty. Industrial and Engineering Chemistry Research, 40, 4543-4554.

[95] Waltz, R. A., & Plantenga, T. D. (2006). KNITRO User's Manual (Version 5.0): Ziena Optimization, Inc.429

[96] Wang, J. (2004). A fuzzy robust scheduling approach for product development projects. European Journal of Operational Research, 152, 180-194.

[97] Wu, D., & Ierapetritou, M. (2004). Cyclic short-term scheduling of multiproduct batch plants using continuous-time representation. Computers & Chemical Engineering, 28, 2271-2286.

[98] Wu, D., & Ierapetritou, M. G. (2003). Decomposition approaches for the efficient solution of short-term scheduling problems. Computers & Chemical Engineering, 27, 1261-1276.

[99] Wu, D., & Ierapetritou, M. G. (2007). Hierarchical approach for production planning and scheduling under uncertainty. Chemical Engineering and Processing, 46, 1129-1140.

[100] Zhu, X. X., & Majozi, T. (2001). Novel continuous-time MILP formulation for multipurpose batch plants. 2. Integrated planning and scheduling. Ind. Eng. Chem. Res., 40, 5621-5634.

Appendix A. Parametric Linear Complementarity Problem

For a given matrix $M \in \mathrm{R}^{n\times n}$ and vector $q \in \mathrm{R}^{n}$, the *Linear Complementarity Problem* LCP(q, M) is to find solution $z \in \mathrm{R}^{n}$, $w \in \mathrm{R}^{n}$, such that it satisfies the following equations or else to determine that no solution exists:

$$z^{\mathrm{T}} w = 0 \tag{A.1a}$$

$$w = q + Mz \tag{A.1b}$$

$$w \geq 0\ ,\ z \geq 0 \tag{A.1c}$$

The *Parametric Linear Complementarity Problem* pLCP$(q(\theta), M)$ is defined as follows: given $M \in \mathrm{R}^{n\times n}$, $q(\theta) = d + F\theta$, $d \in \mathrm{R}^{n}$, $F \in \mathrm{R}^{n\times m}$, find solution map $z \in \mathrm{R}^{n}$, $w \in \mathrm{R}^{n}$ in the range $\theta \in [\theta^{L}, \theta^{U}]$, $\theta^{L} \in \mathrm{R}^{m}$, $\theta^{U} \in \mathrm{R}^{m}$, such that

$$z^{\mathrm{T}} w = 0 \tag{A.2a}$$

$$w = d + F\theta + Mz \tag{A.2b}$$

$$w \geq 0\ ,\ z \geq 0 \tag{A.2c}$$

The problem of finding a solution for the LCP problem (A.1) can be recasted as finding a solution for the following set of mixed integer constraints:

$$y^{\mathrm{T}} w + (e - y)^{\mathrm{T}} z = 0 \tag{A.3a}$$

$$w = q + Mz \tag{A.3b}$$

$$w \geq 0,\ z \geq 0 \tag{A.3c}$$

$$y \in \{0,1\}^{\mathrm{n}\times 1} \tag{A.3d}$$

where $e = [1,1,...,1]^{\mathrm{T}}$. If this problem is infeasible, it means that the original LCP (1) does not have a solution.

Since the LCP can be reduced to a set of mixed integer constraints (A.3), the solution of LCP can be obtained by solving a mixed integer programming problem which is formed with these constraints and an additional objective function. By setting the objective function, we can get the minimum norm solution of the LCP problem (Rosen, 1990). The *minimum norm solution* to LCP(q, M) is a solution z^* such that $\| z^* \| \leq \| z \|$ is satisfied for all solutions z of the LCP problem. For example, the minimum 1-norm solution formulation for LCP (A.1) can be formulated as the following:

$$\min \quad e^{\mathrm{T}} z \tag{A.4a}$$

$$\text{s.t.} \quad y^{\mathrm{T}} w + (e - y)^{\mathrm{T}} z = 0 \tag{A.4b}$$

$$w = q + Mz \tag{A.4c}$$

$$w \geq 0,\ z \geq 0,\ y \in \{0,1\}^{n \times 1} \tag{A.4d}$$

For the pLCP (A.2), the corresponding minimum 1-norm solution formulation is similar to problem (A.4) except that (A.4c) should be changed to $w = Mz + d + F\theta$ as follows:

$$\min \quad e^{\mathrm{T}} z \tag{A.5a}$$

$$\text{s.t.} \quad y^{\mathrm{T}} w + (e - y)^{\mathrm{T}} z = 0 \tag{A.5b}$$

$$w = d + F\theta + Mz \tag{A.5c}$$

$$w \geq 0,\ z \geq 0,\ y \in \{0,1\}^{n \times 1} \tag{A.5d}$$

For a mixed integer linear complementarity problem, some of the variables are required to take integer values. This type of problem does not get too much attention in the literature (Chandrasekaran et al., 1998; Pardalos & Nagurney, 1990) and the parametric mixed integer LCP has not even been studied before. The proposed solution algorithm for pLCP can also address the mixed integer pLCP as follows.

Without loss of generality, we can always assume that the first k variables of vector z in pLCP (A.2) are restricted to be binary by rearranging the columns of M, i.e., $z_i \in \{0,1\}$, $i \in K = \{1,2,...,k\}$, $k \leq n$. To solve this mixed integer pLCP, we propose the following minimum 1-norm formulation:

$$\min \quad e^{\mathrm{T}} z \tag{A.6a}$$

$$\text{s.t.} \quad y^{\mathrm{T}} w + (e - y)^{\mathrm{T}} z = 0 \tag{A.6b}$$

$$w = d + F\theta + Mz \tag{A.6c}$$

$$z_i = v_i \qquad \forall i \in K \tag{A.6d}$$

$$w \geq 0,\ z \geq 0,\ y \in \{0,1\}^{n\times 1},\ v \in \{0,1\}^{k\times 1} \tag{A.6e}$$

The above parametric MILP formulation (A.5) and (A.6) can be solved with the algorithm presented in Chapter 2, which gives the solution for the pLCP problem.

Example. Parametric Quadratic Programming

Consider the following quadratic programming problem

$$\begin{aligned} &\min_{x\geq 0} 0.5x^T Qx + c^T x \\ &s.t.\ \ Ax \geq b \end{aligned} \tag{A.6}$$

where Q is positive semidefinite, then the Karush-Kuhn-Tucker (KKT) condition guarantees global optimality for the QP problem. According to the KKT theorem, if the vector x is a local minimizer for the QP problem (A.6), there exists a vector λ such that it satisfies the following KKT conditions:

$$Ax \geq b,\ x \geq 0,\ c + Qx - A^T\lambda \geq 0 \tag{A.7a}$$

$$\lambda \geq 0,\ \lambda^T(Ax - b) = 0,\ x^T(c + Qx - A^T\lambda) = 0 \tag{A.7b}$$

with the following definition

$$M = \begin{bmatrix} Q & -A^T \\ A & 0 \end{bmatrix},\ q = \begin{bmatrix} c \\ -b \end{bmatrix},\ z = \begin{bmatrix} x \\ \lambda \end{bmatrix}, \tag{A.8}$$

The KKT conditions (A.7) form the problem $LCP(q, M)$ (Jones & Morrari, 2006). Thus, parametric quadratic programs where the cost c and the right hand side of the constraints b are parameterized can be solved through the solution of the corresponding pLCP .

Appendix B. Convergence Property of Augmented Lagrangian Optimization Algorithm

For the following general problem:

$$\begin{aligned} &\min \ f(x) \\ &\text{s.t. } h(x)=0,\ g(x)\le 0, \\ &\quad x\in\Omega=\{x\mid H(x)=0, G(x)\le 0\} \end{aligned} \quad , \tag{B.1}$$

and its augmented Lagrangian dual problem generated by relaxing the upper level constraints $h(x)=0,\ g(x)\le 0$:

$$\max_{\lambda,\nu\ge 0,\sigma}\ \min_{x\in\Omega} L(x,\lambda,\nu,\sigma)= f(x)+\frac{\sigma}{2}\sum_i\left(h_i(x)+\frac{\lambda_i}{\sigma}\right)^2+\frac{\sigma}{2}\sum_j\left(g_j(x)+\frac{\nu_j}{\sigma}\right)_+^2 , \tag{B.2}$$

where $\left(g_j(x)+\frac{\nu_j}{\sigma}\right)_+=\max\left(g_j(x)+\frac{\nu_j}{\sigma},0\right)$. The corresponding augmented Lagrangian algorithm is as follows:

Step 0. Given bounds value for the multipliers, $\lambda\in[\lambda_{\min},\lambda_{\max}]$, $\nu\in[0,\nu_{\max}]$, $\gamma>1$, $0<\tau<1$, $\{\varepsilon^k\}$ is a sequence of nonnegative numbers such that $\lim_{k\to\infty}\varepsilon^k=0$. Set k=1, select arbitrarily $\lambda_i^1\in[\lambda_{\min},\lambda_{\max}]$, $i=1,...,m$, $\nu_j^1\in[0,\nu_{\max}]$, $j=1,...,n$, $\sigma^1>0$.

Step 1. Compute an approximate solution x^k of the augmented Lagrangian relaxation problem $\min_{x\in\Omega} L(x,\lambda,\nu,\sigma)$, which satisfies following approximate KKT conditions

a) $\left\|\nabla L(x^k,\lambda^k,\nu^k,\sigma^k)+\nu^k\nabla H(x^k)+u^k\nabla G(x^k)\right\|\le\varepsilon^k$

b) $G_j(x^k)\le\varepsilon^k$, $u_j^k\ge 0$, $\forall j$

c) if $G_j(x^k)<-\varepsilon^k$, then $u_j^k=0$, $\forall j$

d) $\left\|H_i(x^k)\right\|\le\varepsilon^k$, $\forall i$

Step 2. Update multipliers' and penalty parameter's value

$$\lambda_i^{k+1} = \min\left\{\max\left(\lambda_{\min}, \lambda_i^k + \sigma^k h_i(x^k)\right), \lambda_{\max}\right\},\ \nu_j^{k+1} = \min\left\{\max\left(0, \nu_j^k + \sigma^k g_j(x^k)\right), \nu_{\max}\right\}$$

Compute $V_j^k = \max\left(g_j(x^k), -\frac{\nu_j^k}{\sigma^k}\right)$, if $\max\left(\| h(x^k) \|_\infty, \| V^k \|_\infty\right) \le \tau \max\left(\| h(x^{k-1}) \|_\infty, \| V^{k-1} \|_\infty\right)$, set

$\sigma^{k+1} = \sigma^k$, otherwise set $\sigma^{k+1} = \gamma\sigma^k$. Set k=k+1, go to next iteration.

The convergence of the proposed approach is shown using the following theorems. The details of the proofs are not shown but the reader is referred to the literature where proofs are given (Andreani et al., 2008), (Birgin et al., 2009).

Theorem1. The solution sequence $\{x^k\}$ generated by the proposed augmented Lagrangian approach admits a limit point x^*.

It is proved in (Andreani et al., 2008) that even if the augmented Lagrangian relaxation problem is not solved to optimality at every iteration of the augmented Lagrangian optimization algorithm, Constant Positive Linear Dependence (CPLD) based convergence property are ensured under the conditions that f, g, h admit continuous first derivatives and $\Omega = \{x \mid H(x) = 0, G(x) \le 0\}$ is a closed set. Furthermore, it is also proved in (Andreani et al., 2008) that at least a limit point x^* of the sequence $\{x^k\}$ generated by the augmented Lagrangian optimization algorithm exists under the sufficient condition that there exists $\varepsilon > 0$ such that the set $\{x \| \| H(x) \| < \varepsilon, G(x) < \varepsilon\}$ is bounded.

Theorem 2. The limit point x^* is a feasible and local optimal solution of the original problem (B.1).

Regarding the feasibility of the solution, the work of (Andreani et al., 2008) proved that if the sequence of penalty parameters $\{\sigma^k\}$ is bounded (i.e., from some iteration on, the penalty parameters are not updated, or there exists k_0 such that $\sigma^k = \sigma^{k_0}$ for all $k \ge k_0$), the limit point x^* is a feasible solution of problem (B.1). Furthermore, if the limit point x^* satisfies the Constant Positive Linear Dependence (CPLD) constraint

qualification condition (Qi & Wei, 2000) with respect to the lower-level constraints $x \in \Omega$, then x^* is a KKT (stationary) point of the original problem (B.1).

Theorem 3. The algorithm converges to a ε^{BB}-global optimal solution of original problem.

The above theorem is proved by (Birgin et al., 2009), which pointed out that for problem (B.1), if in each outer iteration, an ε_k-global minimization of the relaxation problems is found, where $\varepsilon_k \to \varepsilon^{BB}$, then the convergence to ε^{BB}-global minimum of the original problem is ensured for the augmented Lagrangian method.

Curriculum Vitae

ZUKUI LI

EDUCATION

Rutgers - The State University of New Jersey, Piscataway, NJ, USA

Ph.D.	in Chemical and Biochemical Engineering	2010

University of Science and Technology of China, Hefei, China

M.E.	in Control Theory and Engineering	2005
B.E.	in Automatic Control	2002

PUBLICATIONS

·**Z. Li**, M.G. Ierapetritou. Production planning and scheduling integration through augmented Lagrangian optimization. *Computers & Chemical Engineering,* doi:10.1016/j.compchemeng. 2009.11.016, 2009.

·**Z. Li**, M.G. Ierapetritou. Integrated planning and scheduling in a decomposition framework. *Chemical Engineering Science,* 2009, 64, 3585.

·M.G. Ierapetritou, **Z. Li**. Modeling and managing uncertainty in process planning and scheduling. *Optimization and Logistics Challenges in the Enterprise*, W. Chaovalitwongse et al.(eds.), Springer Optimization and Its Applications 30, 2009, 97.

·**Z. Li**, M.G. Ierapetritou. Reactive scheduling using parametric programming. *AIChE Journal,* 2008, 54, 2610.

·**Z. Li**, M.G. Ierapetritou. Robust optimization for process scheduling under uncertainty. *Industrial & Engineering Chemistry Research*, 2008, 47, 4148.

·**Z. Li**, M.G. Ierapetritou. Process scheduling under uncertainty: review and challenges. *Computers & Chemical Engineering*, 2008, 32, 715.

·**Z. Li**, M.G. Ierapetritou. Process scheduling under uncertainty using multiparametric programming. *AIChE Journal,* 2007, 53, 3183.

·**Z. Li**, M.G. Ierapetritou. A new methodology for the general mpMILP problems. *Industrial & Engineering Chemistry Research*, 2007, 46, 5141.