PLANNING AND SCHEDULING IN PHARMACEUTICAL SUPPLY CHAINS

ARUL SUNDARAMOORTHY
(B. Tech. REGIONAL ENGINEERING COLLEGE, TIRUCHIRAPPALLI)

A THESIS SUBMITTED FOR THE DEGREE OF

MASTER OF ENGINEERING

DEPARTMENT OF CHEMICAL & BIOMOLECULAR ENGINEERING

NATIONAL UNIVERSITY OF SINGAPORE

2004

ACKNOWLEDGEMENTS

It gives me an immense pleasure to thank my supervisor A/P I.A.Karimi for his priceless guidance, motivation and suggestions throughout my research period at National University of Singapore, Singapore. Admiring his ingenuity, I assert that his ideas and recommendations on the project have played a substantial role in completing this work successfully. I also thank him for the full freedom of thoughts that he has given me in my research work. Further, I extend my sincere and deepest gratitude for his advice and moral support throughout my stay here in Singapore.

I would like to thank all my lab-mates, friends and colleagues who provided the consistent moral support to carry out the research work successfully. I am very thankful to my mother and brothers who have driven me through out my research work with their love. Needless to say, I am thankful to God and my father for their love and blessings.

I would like to thank Prof. G.V.Reklaitis, Purdue University, USA for his invaluable comments and suggestions in some parts of my research work. Finally, I thank the National University of Singapore for providing me the research scholarship to complete this project.

TABLE OF CONTENTS

SUMMARY

In this work, we address two major problems in pharmaceutical supply chains. One is the planning problem that involves outsourcing and new product introductions. The other is the scheduling problem of operating multipurpose plants.

A pharmaceutical plant repeatedly needs to resolve whether it can or should undertake to produce a new intermediate or product, or should outsource some tasks to enable it to do so. We present a multi-period, continuous-time, mixed-integer linear program (MILP) model that addresses this important problem for a pharmaceutical plant using multiple parallel production lines in campaign mode, and producing products with multiple intermediates. Given a set of due dates, demands of products at these due dates, several operational, and cleaning requirements, the aim is to decide the optimal production levels of various intermediates (new and old) or the optimal outsourcing policy to maximize the overall gross profit for the plant, while considering in detail the sequencing and timing of campaigns and material inventories. The effects of new product introductions on plant production plans, the benefits of outsourcing, and the ability to react to sudden plant/demand changes are illustrated using few examples.

Scheduling of multipurpose batch plants like pharmaceutical plants is a challenging problem for which several formulations exist in the literature. In this work, we present a new, simpler, more efficient, and potentially tighter, MILP formulation using a continuous-time representation with synchronous slots and a novel idea of several balances (time, mass, resource, etc.). The model uses no big-M constraints, and is equally effective for both maximizing profit and minimizing makespan. Using extensive, rigorous numerical evaluations on a variety of test problems, we show that in contrast to the best model in the literature, our model does not decouple tasks and units, but still has fewer binary variables, constraints, and nonzeros, and is faster. In addition, we propose some minimal criteria for any model comparison exercise.

Finally, we conclude and propose some recommendations for future work.

NOMENCLATURE

ABBREVIATIONS

AI/API	Active Ingredients/Active Pharmaceutical Ingredients
G&G	Giannelos and Georgiadis (2002)
GCP	Good Clinical Practice
GLP	Good Laboratory Practice
GMP	Good Manufacturing Practice
GRD	Generalized Recipe Diagram
M&G	Maravelias and Grossmann (2003a)
MILP	Mixed Integer Linear Program
PFD	Process Flow Diagram
POMA	Pharmaceutical Outsourcing Management Association
RD	Recipe Diagram
R&D	Research and Development
RMILP	Relaxed Mixed Integer Linear Program
STN	State Task Network

SYMBOLS

Chapters 3-4

Indices

c	customer
i	task
k	slot

l	line
m	material
t	time period

Superscripts

E	end
L	lower bound
S	start
U	upper bound

Sets

$\boldsymbol{C}$	Set of customers
$\boldsymbol{I}$	Tasks
$\boldsymbol{I}_l$	Tasks that line l can perform
$\boldsymbol{L}$	Production lines
$\boldsymbol{L}_i$	Lines that can perform task i
$\boldsymbol{M}$	Materials
$\boldsymbol{M}_i$	Materials that are either produced or consumed by task i

Parameters

a_{mt}	Penalty for dipping below target level of m in t
CC_{il}	Setup or changeover cost to begin i on l
CL_{ikl0}	Initial campaign length of i in k of l
CT_{il}	Changeover time for i on l
$d_{ii'}$	Delay time if i precedes i' in RD
D_{mct}	Demand for m by c in t
DD_t	Due date defining the end of period t
g_{mc}	Revenue per unit of m when sold to c

H	Planning horizon
H_{lt}	Time available for production on l during t
hc_{mt}	Holding cost per unit of m during t
I_{m0}	Initial inventory level of m
I^{-}_{mc0}	Initial backlog of m to c
I^{*}_{mt}	Target level for m in t
MCL_{il}	Minimum campaign length of i on l
NK_{l}	Number of slots in each period on l
NT	Number of periods in the planning horizon
P_{m}	Cost (purchase, transport and insurance) per unit of m
pc_{ml}	Production cost per unit of m on l
QT_{il}	Validation time of new i on l
R^{L}_{il}	Minimum production rate of i on l
R^{U}_{il}	Maximum production rate of i on l
ST_{m}	Available maximum storage capacity for m
YS_{il0}	Initial value of spill over binary for i on l
μ_{i}	Primary material that task i produces
σ_{mi}	Bill of materials coefficient

Binary Variables

Y_{iklt}	1 if line l performs task i in slot k in period t

0-1 Continuous Variables

YS_{ilt}	1 if task i of line l spills over from period t to period t+1
Z_{iklt}	1 if task i is performed first time in slot k of line l in period t

Continuous Variables

CL_{iklt} Campaign length of i in k of l in t

CL_{klt} Length of slot k on l in t

DQ_{ilt} Differential quantity of material produced or consumed by i on l in t

I^{-}_{mct} Shortage of m to c in t

I_{mt} Inventory level of m at the end of t

I^{Δ}_{mt} Margin by which m is short of its target level in t

NC_{ilt} Number of campaigns of i on l in t

OQ_{mt} Quantity of m outsourced in t

PQ_{mlt} Quantity of material m produced or consumed by i on l in t

PT_{ilt} Actual time for which i produces on l in t

SQ_{mct} Quantity of m supplied to c in t

T^{S}_{iklt} Starting time of i in k of l in t

T^{S}_{klt} Starting time of k on l in t

T^{E}_{klt} Ending time of k on l in t

Chapters 5-6

Indices

i task

j unit

k/n slot/event point

m material

Superscripts

L lower bound

U	upper bound

Sets

I	Tasks
$\boldsymbol{I}_j$	Tasks that unit *j* can perform
$\boldsymbol{II}_m$	Tasks that consume material *m*
J	Units
$\boldsymbol{J}_i$	Units that can perform task *i*
K /N	Slots/event points
M	Materials
$\boldsymbol{M}_i$	Materials that are either produced or consumed by task *i*
$\boldsymbol{OI}_m$	Tasks that produce material *m*

Parameters

B_{ij}^{L}	Minimum batch size of *i* on *j*
B_{ij}^{U}	Maximum batch size of *i* on *j*
D_m	Fixed demand of material *m*
g_m	Net revenue or profit per unit (kg or mu) of *m*
H	Scheduling Horizon
I_m^U	Available maximum storage capacity for *m*
I_{m0}	Initial inventory level of *m*
α_{ij}	Fixed processing time of task *i* on *j*
β_i	Variable processing time of task *i* on *j*
μ_i	Primary material that task *i* produces
σ_{mi}	Bill of materials coefficient
τ_{ij}	Constant processing time of task *i* on *j*

Binary Variables

Y_{ijk} 1 if unit j begins task i at time T_k

0-1 Continuous Variables

y_{ijk} 1 if unit j is engaged with task i at time T_k

YE_{ijk} 1 if unit j discharges material(s) of task i at time T_k

Z_{jk} 1 if unit j begins a task at time T_k

Continuous Variables

B_{ijk} Batch size of task i that unit j begins at T_k

b_{ijk} Amount of batch that exists in unit j just before a new task begins at T_k

BE_{ijk} Size of batch discharged by task i at its completion at T_k

I_{mk} Inventory level of m at T_k

SL_k Length of slot k that spans from $T_{(k-1)}$ to T_k

T_k Time at which slot k ends

t_{jk} Time remaining at T_k to complete the task in progress in slot k on unit j

LIST OF FIGURES

LIST OF TABLES

Chapter 1

INTRODUCTION

Global competition requires every pharmaceutical company to enhance its economic performance. These companies are undergoing major retrofits in their business practice in order to survive the new challenges of the modern economy. The globalization of the business, the variety and complexity of new drugs, and the shortening patent protections are some of the factors driving these changes. Usually, pharmaceutical companies produce several high-profit, low-volume products. Of these, only flagship/dominant products under patent protections are the major contributors to the growth of these companies. Hence, high product turnover is crucial to the continued economic survival and growth of a pharmaceutical company.

Pharmaceutical companies often have several facilities, which are geographically distributed. These companies tend to have their Research and Development (R&D) in some location and production facilities in some other locations. Such distributions of facilities are based on several global factors like market demands, economies of scale, logistics and so on. Mostly, the business activities in different locations are not sufficiently integrated to achieve the best possible solutions.

The pharmaceutical industry is distinctive from many other industries in the amount of attention paid to it by the regulatory authorities. Since these industries produce health care products, stringent work practices like Good Manufacturing Practice (GMP), Good Laboratory Practice (GLP), and Good Clinical Practice (GCP) and so on are followed in the production sites. Most of the operations in pharmaceutical production are batch, and hence quality check must be performed by keeping track of each batch. In addition, thorough cleaning must be performed whenever product changeover occurs. This is mainly to avoid cross contamination of products during the changeovers. All these work practices, which are inevitable, lessen the overall productivity. Hence, these companies are under great pressure to utilize their production resources efficiently.

Pharmaceutical companies aspire to introduce new products in order to revive their business with the early profits. The time to market and the quick reap of the profits from the new products before their shortened life cycles are the keys to the success of these companies. Hence, a lot of money and time are invested on the research and development of new products.

1.1 Life Cycle of a Pharmaceutical Product

A pharmaceutical product has four different phases in its life cycle as shown in Figure 1.1. In the *Birth* phase, an active molecule with a curative effect on a target disease group is discovered. Then, several studies are performed to enhance its efficacy. As a result, the most active molecule is structured, which is then tested for toxicological results in rats or mice. If no worrisome toxic endpoints are observed, then this molecule becomes a candidate for further development.

In the *Development* phase, the candidate undergoes a series of processes such as sampling, testing, patenting etc. Enormous amounts of money and resources are invested in these tedious processes. In addition, process costs and durations, their success probabilities, and their potential revenues are not known with confidence in the initial stage of this phase. If a process fails, all work on that candidate is halted, and the investment in previous processes may be futile. Hence, risk levels are high in these development processes. If the candidate does succeed out of these complex processes, then it is approved for the commercial production under a patent coverage.

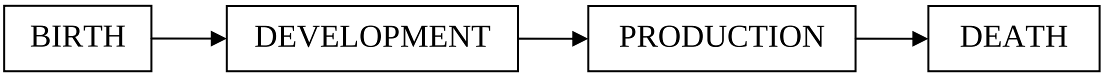


Figure 1.1: Life cycle of a typical pharmaceutical product.

In the *Production/Launch* phase, promising markets are identified for a successful launch of the new product. The launch strategy could be either forecast-driven or response-based. In either case, if the new product succeeds technically as well as financially, then it may survive actively in the market, until its patent expires.

The product is no longer new, when it reaches the *Death* phase. The patent has expired, and the target markets are now open to generics. Hence, the demand of the product either stagnates or declines. If the product is no more fruitful, then its production is stopped.

Of the four phases of the product's life cycle, Development (conceptualization, design, promotion, and pricing) and Production/Launch (physical positioning in the market via commercial production) are the major ones. The product launch phase consumes a significant amount of costs, often exceeding the combined expenditures in all previous development stages (Beard and Easingwood, 1996). Launch phase includes identifying the right place to market, right production site to produce, and optimizing the planning and scheduling of the production of new products. Mistakes, miscalculations, and oversights in any of these product launch activities can become fatal obstacles to new product success. Hence, the optimal planning of new product introductions into the appropriate production sites so as to target the right market is of paramount importance to any pharmaceutical company.

1.2 Pharmaceutical Supply Chain

Most pharmaceutical products undergo two levels of production (Bennett and Cole, 2003): primary and secondary. While the primary production involves making the basic molecules called the active ingredients (AI) or active pharmaceutical ingredients (API), the secondary production involves formulating them into final drugs and supplying them to various customers. Figure 1.2 shows the different layers in a typical pharmaceutical supply chain.

The first layer comprises suppliers that provide raw materials and/or intermediates to the primary and/or secondary production sites. It also includes third party contractors who may supply some intermediates or even APIs.

The second layer includes the primary production facilities that perform various chemical synthesis steps and downstream separations in the case of traditional pharmaceuticals, and fermentation, product recovery, and purification in the case of biopharmaceuticals. Production of an API typically requires complex chemistry involving multiple stages or intermediates. The stringent requirements for cleaning and the need for avoiding cross contamination result in long transition times during product changeovers, which necessitate long campaigns for effective utilization of plant equipment. If the existing production facilities cannot meet all the demands, a company may even outsource some intermediates from third party contractors.

A primary production site is driven mainly by the medium- and long-term forecasts and is less responsive to the changes in the demands of end/finished products. It holds inventories of AIs to ensure good service levels and to maintain smooth operation at the downstream production sites. Thus, anticipatory logistics (or “push” process) dominates the primary production, and primary production is often the rate-limiting step in pharmaceutical supply chains.

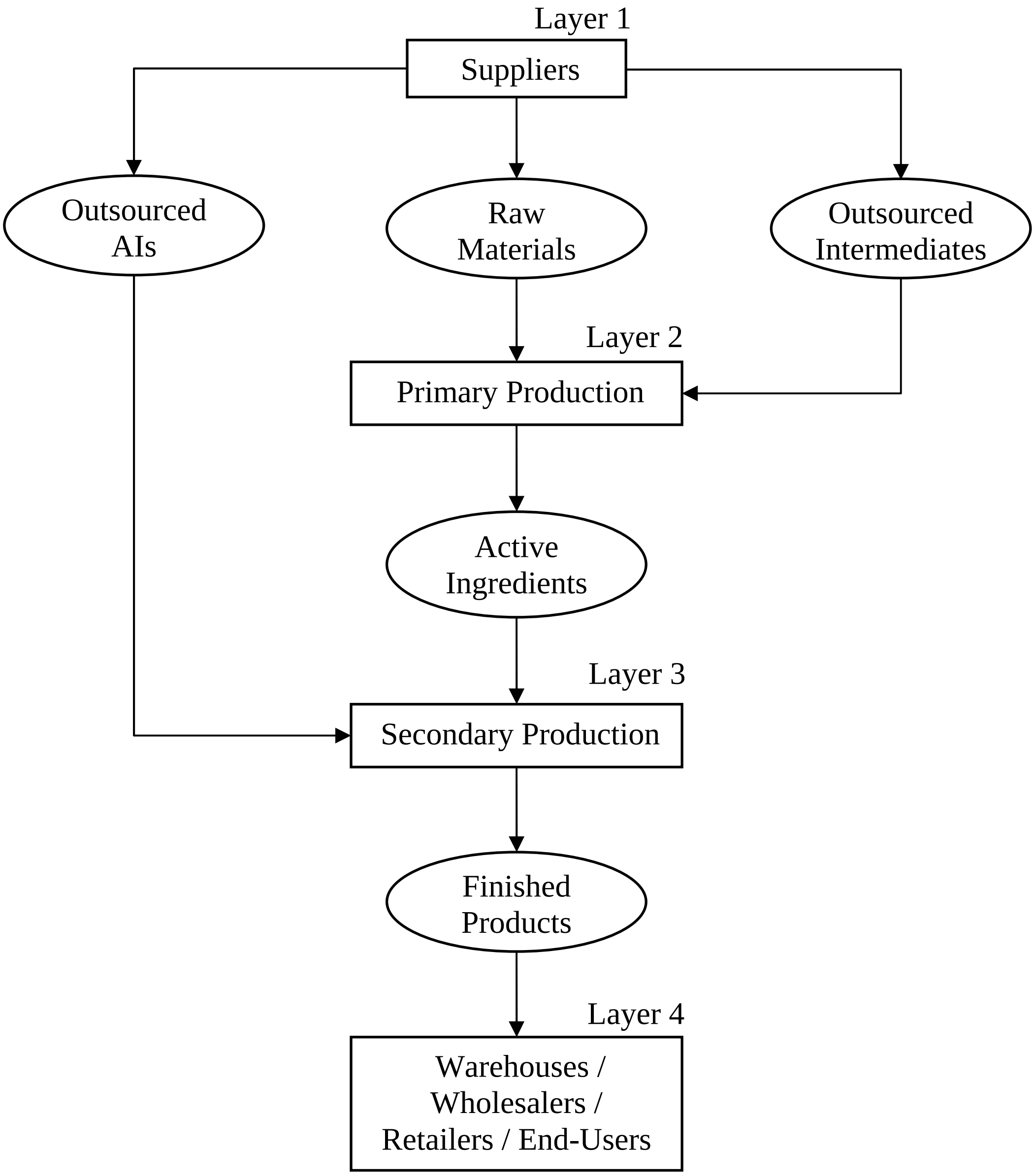


Figure 1.2: A typical pharmaceutical supply chain.

The third layer includes the secondary production sites that add inert materials such as fillers, coloring agents, sweeteners, etc. to the AIs, and formulate and package them to produce finished products such as tablets, capsules, syrups etc. Their processing steps include milling, granulation, compression (to form pills), coating, packaging etc. Relatively short campaigns or batches of huge size are common in the secondary production sites. Formalized cleaning is also a requirement, and outsourcing of AIs from external contractors is common. Most often secondary production sites outnumber the primary ones, are geographically separate from the latter ones, and are closer to the markets. A response-based logistics (or "pull" process) based on customer orders dominates the secondary production, and this layer is more responsive to the market fluctuations.

The fourth layer includes the various customer nodes such as distribution warehouses, wholesalers, retailers, and end-users. These nodes are normally geographically distributed, and separate from the production sites.

Pharmaceutical companies have long been looked as the laggards of supply chain practice. Given their huge profits from proprietary blockbuster drugs, these companies have always made product availability a greater priority than supply chain efficiency. In the past, pharmaceutical companies have neglected supply chain management because its costs are insignificant compared to sales and marketing or R&D. But now, a number of factors like (a) increased competition from generics, (b) shorter patent life cycle, (c) increased pressure to reduce health care costs, (d) consolidation of industries and proliferation of products and so on are putting pressure on pharmaceutical companies to change their traditional ways of doing business.

1.3 Planning and Scheduling

Both planning and scheduling aim at the optimal performance of an industry. However, they do differ mainly in terms of the time frames involved and the level of decisions taken. Planning normally has longer time horizon (order of months/years) and includes higher level management objectives, policies, etc. besides immediate production requirements. It represents aggregated objectives and usually does not include more details. Accordingly, the models used are either abstract or take simplifying assumptions making them more conceptual. If the assumptions overestimate the facility performance giving very little allowances, the resultant plan can become unrealistic. On the other hand, if assumptions underestimate the plant's efficiency, the plan thus obtained might lead to under-utilized production capacities. Therefore, for planning operations, one has to include the key detailed constraints and their interdependencies in order to get an optimal plan and hence a sound basis for undertaking further scheduling. On the other hand, scheduling is the link between the production and the customer. The issues addressed by the scheduling vary with the characteristics of the production process and the nature of market served. Hence, scheduling can be formally defined as the specification of what each stage of production is supposed to do over short scheduling horizon ranging from several shifts to weeks. The objective of scheduling is to implement the plan, subject to the variability that occurs in the real world. This variability can be in raw materials supplies, product quality, production process, customer requirements or logistics.

Planning and Scheduling play a vital role in the pharmaceutical supply chain. Optimal plan is required in both primary and secondary production sites of the pharmaceutical plants. Mostly, primary and secondary sites exercise own production plan for the reasons discussed in the previous section. Hence, a plan that does not address the key issues of the plants may often lead to suboptimal or infeasible schedule. In addition, planning is very important while introducing new products in a plant. One has to consider several global issues before launching new products for commercial production. Moreover, the development stages of the new product candidates also require the scheduling of various testing tasks that involve high levels of uncertainty.

The objective of the planning is mostly based on economic criteria like maximizing the net profit or revenue, minimizing the cost and so on. Some of the factors that drive planning in pharmaceutical plants are:

1. Meeting the forecasted demand fully.
2. Optimal introduction of new products in the production facilities.
3. Keeping low tie-up of Working capital (minimal inventory).
4. Meeting demands even during planned shutdowns of the plants.

The objective of scheduling is often based on operational criteria like minimizing the makespan, maximizing the production, minimizing the tardiness/earliness and so on. The factors that drive scheduling in pharmaceutical plants are:

1. Meeting the demands in the face of high volatility.
2. Reacting to the uncertainties in plants.
3. Better utilization of resources (production units, utilities, manpower and so on) in plants.
4. Maintaining the safe inventory levels.

1.4 Research Objective

It is clear from the above discussion that planning and scheduling play an important role in pharmaceutical plants. The optimal plan and schedule of production activities can tremendously improve the economic performance of these plants. Planning and scheduling can find their application in R&D, facility expansion, production and so on. However, the objective of this work is to present the optimization models for the optimal planning and scheduling of production in pharmaceutical plants.

The planning model should provide decision support for the plant management in selecting which existing products to produce in what quantities so that new products, if any, can be produced in the plant. As the pharmaceutical industry focuses more on the discovery and development activities, outsourcing of some of its testing or production tasks to the external contractors is becoming growingly important. However, the decision to outsource requires several considerations. A production facility may consider outsourcing an intermediate, when it is unable to meet the demands of its products with the existing facility. It may also consider outsourcing, when it is more profitable to use the facility to produce a new product rather than a nearly off-patent product. Hence, the planning model should also address the above issues in outsourcing.

The scheduling model is expected to resolve the problems that could arise due to dynamic demands of products in pharmaceutical plants. Since the available units in a plant are limited, the optimal scheduling is required to better utilize these units in order to meet the demands of several products. Scheduling model considers many real-life operational and supply constraints.

1.5 Outline of the Thesis

This thesis consists of two major sections. The first (Chapters 3-4) and second (Chapters 5-6) sections respectively deal with the planning and scheduling of production activities in pharmaceutical plants. In Chapter 3, we develop a mathematical model for the planning of production in pharmaceutical plants. The planning model also includes scheduling aspects to make the plan realistic. Though we develop the model for planning primary production, we discuss its flexibility to address the planning in secondary production as well. In Chapter 4, we evaluate the performance of the proposed planning model using few examples. Here, we study various business practices like outsourcing, new product introduction and so on using our model.

In Chapter 5, we present a novel mathematical formulation for scheduling in pharmaceutical plants. In Chapter 6, we assess the performance of our scheduling model using several scenarios of three examples. We also compare the performance of our model with those of two other scheduling models existing in the literature. Here, we present some required minimal criteria for the comparison works.

In Chapter 7, we summarize the conclusions of our work, and then provide some recommendations for its potential extensions.

Chapter 2

LITERATURE SURVEY

The industries producing specialty chemicals such as pharmaceuticals, cosmetics, polymers, food products, and electronic materials produce several products, and often introduce new products. However, the pharmaceutical industry has the longest product development period of all. A lot of money and time is invested in the development of pharmaceutical products. If a new product fails at any stage of its development, then all the remaining work on that product is halted and the investment in the previous tests is wasted. Hence, the scheduling of these highly uncertain development activities is increasingly receiving attention.

2.1 New Product Development

Pharmaceutical plants routinely introduce new products in order to revive their business with the early profits. The need for introducing new products early to the market and the uncertainties inherent with the development processes necessitated much research to focus on the portfolio selection a priori to development and scheduling of product development tasks. Schmidt and Grossmann (1996) address the problem of scheduling testing tasks in new product development. They assume that unlimited resources are available for the testing tasks. In reality, these testing tasks often tend to be resource-constrained, and may involve outsourcing of some tests. Hence, Jain and Grossmann (1999) extend the above work and develop a MILP model that performs the sequencing and scheduling of testing tasks for new product development under resource constraints. Blau et al. (2000) use probabilistic network models to capture all the testing activities and their uncertainties involved in the development of new products. They address the issue of managing risk in the selection of new product candidates. Following this work, Bose and Blau (2000) use graphtheoretic techniques to translate a probabilistic network model of a sequence of process activities into a spreadsheet model.

Subramanian et al. (2000) present an integrated simulation-optimization framework, sim-opt, that combines mathematical programming and discrete event system simulation to evaluate the uncertainty and control the risk present in the R&D pipeline. Mockus et al. (2000) propose a two-level approach to address the problem of planning and scheduling in a pharmaceutical pilot plant. They decompose the above problem into long-term planning of resources and short-term scheduling of operations. Mockus et al. (2002) extend the previous work (Mockus et al., 2000) and explore the techniques for combining the production plan and daily operation schedule in a single model. Maravelias and Grossmann (2001) propose an MILP model that integrates the scheduling of testing tasks with the design and production planning decisions. A common assumption in all the above works is that the resources are constantly available throughout the testing period. In practice, the existing resources may not be sufficient to launch the new products in a timely fashion. Hence, the company may often prefer outsourcing of tests at a high cost. To address these issues, Maravelias and Grossmann (2003) present an MILP model that optimizes the overall costs.

2.2 Planning in Pharmaceutical Supply Chains

Chemical manufacturing processes can be broadly classified into two types based on their modes of operation: batch and continuous. A continuous process or unit is the one which produces the product incessantly, whereas a batch unit or process is the one that produces in discrete batches. A semicontinuous unit is a continuous unit that runs intermittently with starts and stops. Continuous process, in most cases, is dedicated to produce a fixed product with little or no flexibility to produce another. In contrast, batch processes are flexible to produce multiple products and are best suited for producing low-volume, high value products requiring similar processing paths and/or complex synthesis procedures as in the case of specialty chemicals such as pharmaceuticals, cosmetics, polymers, food products, electronic materials etc. The latter is also referred as multipurpose batch processes in the literature.

Batch plants operate in either batch or campaign mode. Many pharmaceutical plants producing large amounts of active pharmaceutical ingredients (APIs) employ either multiproduct or multiplant structure (use production lines) and operate in campaign mode. The APIs serve as feeds to the downstream or secondary processing facilities producing final drugs. As long as a plant employs long, single-product campaigns of identical batches, one can model its operation in a manner similar to that of a semicontinuous plant producing products such as polymers, papers, etc., in large quantities. Therefore, the research on batch plants as well as semicontinuous plants is of interest. In the campaign mode of operation, timings and durations of campaigns and their allocation to various production lines over a relatively long period (several months) are the major operational decisions, so the operation management problem falls in the category of planning rather than scheduling.

The early work on campaign planning in general has addressed the production planning of a single facility with one or more noncontinuous production lines or multiple distributed facilities. While Sahinidis and Grossmann (1991) assumed cyclic campaigns in an infinite horizon, the most recent works (McDonald and Karimi, 1997; Karimi and McDonald, 1997; Ierapetritou and Floudas, 1999; Gupta and Maranas, 1999; Gupta and Maranas, 2000; Oh and Karimi, 2001a; Oh and Karimi, 2001b; Lamba and Karimi, 2002a; Lamba and Karimi, 2002b; Lim and Karimi, 2003b; Jackson and Grossmann, 2003) have focused on acyclic campaigns in a finite horizon. The latter works are more realistic from a practical viewpoint and more suitable for time-varying demand scenarios. Furthermore, they subsume the extremely unlikely scenario that the optimal solution involves cyclic campaigns.

McDonald and Karimi (1997) presented a realistic mid-term planning model for parallel semicontinuous processors. Although they incorporated minimum campaign length constraints in their formulation, they did not consider the detailed timings of campaigns. However, in their second paper, Karimi and McDonald (1997) presented two novel multi-period continuous-time formulations for the detailed timings of campaigns using time slots. In both works, the product demands were due at the end of each period. Ierapetritou and Floudas (1999) applied their event-point based formulation on this problem, but the recent works (Sivanandam, 2004; Balla, 2004) reveal that several issues in their comparison are unpersuasive. Gupta and Maranas (1999) develop an efficient decomposition procedure for solving the same problem based on Lagrangean relaxation. In their subsequent work (Gupta and Maranas, 2000), they proposed a two-stage stochastic programming approach to incorporate demand uncertainty.

Recently, Oh and Karimi (2001a, 2001b) addressed the production planning of a single processor with sequence-dependent setups and given finite horizon. Later, Lamba and Karimi (2002a, 2002b) and Lim and Karimi (2003b) addressed the campaign planning of multiple parallel processors with shared resource constraints. While Lamba and Karimi (2002a, 2002b) used synchronous time slots to satisfy shared resource constraints, Lim and Karimi (2003b) showed improvement by using asynchronous slots.

Some recent work has also addressed some supply chain management issues related to the production planning and product distribution of multiple facilities. Jackson and Grossmann (2003) proposed spatial and temporal decomposition methods to solve the multi-period nonlinear programming model. Singhvi et al. (2003) used pinch analysis for aggregate planning in supply chains.

It is clear from the above review that the optimal planning of campaigns in general has received some attention in the literature. However, the same is not true, when it comes to the pharmaceutical industry. As Shah (2003) remarked, the research that directly addresses the issues faced by the pharmaceutical sector is scant and the optimal planning of campaigns within the context of pharmaceutical supply chain has not received due attention. The majority of the work (see section 2.1) has attended to the product pipeline management problem arising in the new product development phase of the life cycle (see Figure 1.1) of pharmaceutical products. However, there is a paucity of research that addresses the planning issues related to new product introductions. Gjendrum et al. (2001) presented a simulation approach to foresee the supply chain dynamics in pharmaceutical plants after the introduction of new products. Papageorgiou et al. (2001) applied mathematical programming techniques to facilitate the strategic supply chain decision-making process in the pharmaceutical industry. They presented an approach to allocate investment and facilities to new products. They used an aggregate approach for decisions such as which products to develop, where to introduce them, and so on. However, their work did not consider outsourcing as an option, and did not account for the impact of new product introductions on the existing products at a facility in detail. In particular, the disruption in the existing production plan resulting from the introduction of a new product at a facility, and the effect on the customer service and production levels of existing products remained unaddressed. Whether it is feasible or even profitable to introduce a new product at a given facility is a very important issue facing many pharmaceutical plants, and this has received little attention so far in the literature.

2.3 Scheduling in General Multipurpose Plants

Scheduling of multipurpose batch plants has received considerable attention in the last decade. Early attempts (Kondili et al., 1993, Shah et al., 1993) used MILP formulations based on the uniform discrete-time representation. However, as the advantages of alternate representations such as non-uniform discrete-time (Mockus and Reklaitis, 1994; Lee et al., 2001) and continuous-time became clear, the recent trend (Ierapetritou and Floudas, 1998; Castro et al., 2001; Giannelos and Georgiadis, 2002; Maravelias and Grossmann, 2003a) has favored continuous-time representations.

The research efforts using continuous-time representation in batch process scheduling have opted to tag themselves with two flavors. The so called slot-based formulations (Karimi & McDonald, 1997) represent time in terms of ordered blocks of unknown variable lengths. The so called event-based formulations (Ierapetritou and Floudas, 1998; Giannelos and Georgiadis, 2002) use unknown points in time at which events such as starts of tasks may occur. Maravelias and Grossmann (2003a) recently attempted to rationalize the different types of time representation.

Both slot-based and event-based representations can be further classified into two types: synchronous (or common) and asynchronous (or uncommon). In the synchronous representation (Lamba & Karimi, 2002a; Lamba & Karimi, 2002b; Giannelos and Georgiadis, 2002; Maravelias and Grossmann, 2003a), slots (or event points) are synchronized or identical or common across all units (or sometimes resource) in a plant, while in the asynchronous (full or partial) representation (Karimi and McDonald, 1997; Ierapetritou and Floudas, 1998; Lim and Karimi, 2003b), they differ from one unit (or resource) to another. Although both representations can in principle handle shared resources such as materials, it is more natural and easier for the former. As shown by Giannelos and Georgiadis (2002) and Maravelias and Grossmann

(2003a), some asynchronous representations may possess errors in (for example) mass balances. Karimi and McDonald (1997) and Lamba and Karimi (2002a, 2002b) had long recognized this pitfall of asynchronous slots for handling shared resources, however Lim and Karimi (2003b) showed that they can still be used successfully and can sometimes be advantageous. To avoid the discrepancy of mass balance in the asynchronous event-based formulation of Ierapetritou and Floudas (1998), Giannelos and Georgiadis (2002) used synchronous event points with some extra timing/sequencing constraints and a concept of buffer time. However, their approach leads to suboptimal solutions, as it seems to hinder the optimal timings of tasks. Recently, Maravelias and Grossmann (2003a) used synchronous time points in their formulation for multipurpose batch plant scheduling and avoided the extra timing/sequencing constraints that Ierapetritou and Floudas (1998, 1999) and Giannelos and Georgiadis (2002) used explicitly.

It is obvious from the above review on scheduling in multipurpose batch plants that there is a need for an efficient model that can generate schedules for the production in multipurpose batch plants like pharmaceutical plants.

2.4 Research Focus

As seen from the above survey, none of the works address the planning problems involving both new product introductions and outsourcing practices in pharmaceutical plants. In addition, no existing scheduling model can efficiently solve the scheduling problems in these plants. In this work, we focus on these two major problems in pharmaceutical plants.

Whether it is profitable or even feasible to introduce a new product at a given facility is a routine but crucial decision for a pharmaceutical company. To address this, we consider pharmaceutical plants operating in campaign mode. We develop a planning model to evaluate in detail the operational and financial effects of new product introductions at such plants. We also address how outsourcing of existing products can lessen these effects and thus make the introduction of high-margin new products more attractive. In other words, we specifically address the supply chain dynamics at the plant level as they relate to the new product introductions in a pharmaceutical plant, and optimize the production, inventory, and outsourcing decisions to maximize gross profit. Here, we focus on the planning of one primary multiplant production site that consumes raw materials, produces and/or outsources intermediates and active ingredients (AIs), maintains necessary inventories, and supplies AIs to secondary production sites. Given a set of due dates, demands of products at these due dates, several operational and cleaning requirements, the aim is to decide the optimal production levels of various intermediates (new and old) and the optimal outsourcing policy to maximize the overall gross profit for the plant, while considering in detail the sequencing and timing of campaigns and material inventories.

For scheduling in pharmaceutical plants, we present a new, simpler and more efficient continuous-time MILP formulation using synchronous slots. We divide the scheduling horizon into a number of variable length slots. To handle the sharing of production units easily and ensure the material balance at any point in horizon, we synchronize the slots on all units.

Chapter 3

PLANNING IN PHARMACEUTICAL PLANTS

In this chapter, we address some important aspects of planning in pharmaceutical supply chains. As discussed in the introduction, pharmaceutical plants are often situated at different geographical locations. These plants face dynamic demands of several end products. Which product to produce in which plant, how much of each product to produce in each plant, which plant should produce the new product(s) and in what quantity are some of the major challenging decisions faced by the pharmaceutical industry. In addition, since the production of primary manufacturing plants is in campaign mode, much of working capital is usually tied up in the inventories of active ingredients. Hence, a proper planning model that can account for several, if possible all, of the above issues is in great demand. Furthermore, Pharmaceutical Outsourcing Management Association (POMA) suggests that the practice of outsourcing can greatly enhance the performance of pharmaceutical supply chains. It is clear from the above discussion that a planning model that can handle the production, inventory, distribution, and outsourcing issues effectively would be a significant contribution to the economic performance of pharmaceutical industry. In what follows, we describe the scope of the planning problem, then develop the formulation and finally present some remarks on the planning model.

3.1 Problem Description

We focus on a primary multiplant production site F that consumes raw materials, produces and/or outsources intermediates and active ingredients, maintains necessary inventories, and supplies AIs to the secondary production sites. We use recipe diagrams to describe the manufacturing processes in F. A recipe diagram (RD) is simply a directed graph in which nodes represent the recipe tasks, arcs represent the various materials with unique properties, and arc directions represent the task precedence. Here, the term material refers to a unique material-state combination. In other words, a chemical A at 60 C is a different material from A at 90 C. Similarly, a task performed on two different materials also means two different tasks. For instance, heating A from 60 C to 90 C is one task, and heating B from 60 C to 90 C is another, although the plant may perform both in the same unit but at two different times. By using different types of arcs to denote different resources, and defining equipment, labor, material, utility, etc. all as resources of various types, we can easily generalize RD (Generalized Recipe Diagram or GRD) to depict resource (like utilities, manpower etc.) utilization too.

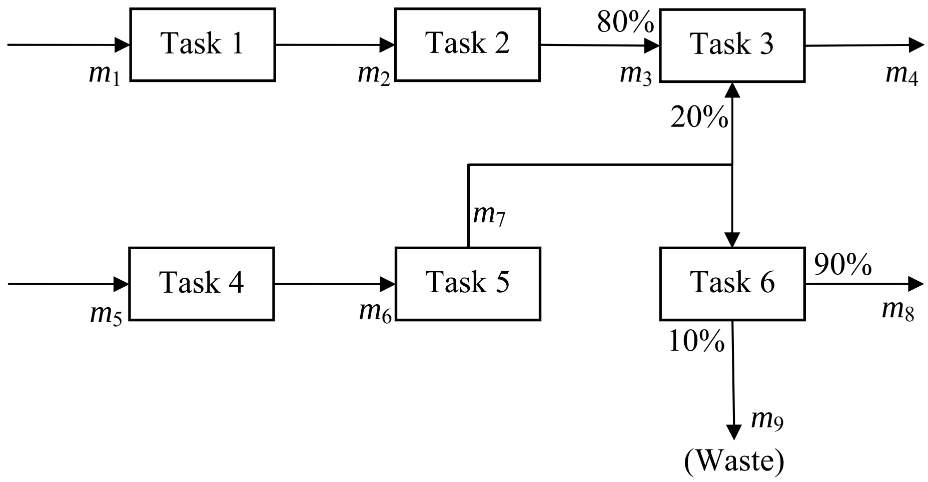


Figure 3.1: Recipe diagram for an example facility producing two products.

Figure 3.1 shows the recipe diagram for an example facility producing two AIs through six tasks. In this example, m_1 (same as $m = 1$) and m_5 are the raw materials; m_2, m_3, m_6, and m_7 are intermediates; m_4 and m_8 are the products (AIs); and m_9 is a waste material associated with the production of m_8. Tasks 3 and 6 share one intermediate m_7. Hence, the production of m_4 requires three intermediates and m_8 requires two intermediates. As we can see, the recipe diagram in Figure 3.1 does provide an unambiguous representation of the recipe without the need to use separate nodes for states. Alternate forms and further discussion of the RDs can be seen in Chapter 5.

The facility F employs $\boldsymbol{L}$ production lines (l = 1, 2, …, $\boldsymbol{L}$). Each production line l comprises multiple stages of noncontinuous equipment and can perform a set $\boldsymbol{I}_l$ of tasks in the recipe diagram using long, single-product campaigns. We use i for tasks and m for materials. Each task i consumes or produces some materials. Let $\boldsymbol{M}_i$ denotes the set of materials ($m \in \boldsymbol{M}_i$) that task i consumes or produces. Note that $\boldsymbol{M}_i$ includes all the different states of raw materials, intermediates, final products, and even waste materials associated with task i. For each task i, we write the mass balance as,

$$\sum_{m \in \boldsymbol{M}_i} \sigma_{mi} (\text{Material } m) = 0$$

where, σ_{mi} is analogous to the stoichiometric coefficient of a species in a chemical reaction except that it is in kg/kg units instead of mol/mol. Thus, $\sigma_{mi} < 0$, if task i consumes material $m \in \boldsymbol{M}_i$ and $\sigma_{mi} > 0$, if it produces $m \in \boldsymbol{M}_i$. Furthermore, for each task i, we designate a primary material μ_i, with respect to which we define the productivity of a line for task i.

Quality controls are highly stringent in pharmaceutical plants. Due to contamination concerns, thorough flushing/cleaning of production lines during the transition from one intermediate to another is mandatory. Moreover, hardware/process changes may also be required between the campaigns of different tasks. Thus, every change of campaign on l may require a considerable changeover time.

Given the above details, our goal is to determine the tasks that each production line should perform, the start/end timings of each task, and the inventory profiles of materials over the planning horizon. The planning objective is to maximize the gross profit of F. To this end, we assume the following.

1. Each single-product campaign is sufficiently long with a stipulated minimum campaign length MCL_{il} for task *i* on line *l*. Therefore, we can treat each production line as semicontinuous with a variable production rate (kg or mu/day, where mu stands for mass unit). Let R_{il}^L and R_{il}^U respectively denote the lower and upper limits on the rate at which line *l* produces (> 0) or consumes (< 0) the primary material μ_i of task *i*.
2. All intermediate materials are stable.
3. Inventory costs for raw materials are negligible, as the plant procures them as and when needed. F has limited capacity for storing the intermediates and final products.
4. All material demands are prespecified point demands. The planning horizon *H* has *NT* discrete, distributed due dates (DD_t, t = 1, 2, …, *NT*) as shown in Figure 3.2 with $DD_0 = 0$ and $DD_{NT} = H$. In other words, although the production can occur at any time between due dates, the product shipments occur only at the due dates.
5. Campaign changeover times are sequence-independent, but task-dependent and line-dependent. Thus, we use CT_{il} to denote the time required to begin a campaign of task *i* on line *l*.

We now develop a continuous-time MILP formulation for the above planning problem.

3.2 Formulation

We view the planning horizon *H* to consist of *NT* periods, where we define the interval [$DD_{(t-1)}$, DD_t] as period *t*. We use a separate local time-axis for each period, so $DD_{(t-1)}$ in real time corresponds to time zero for period *t*, while DD_t to time DD_t–$DD_{(t-1)}$. Let H_{lt} denote the total available production time on line *l* during period *t*. We break this production time in each period on line *l* into $NK_l = |\boldsymbol{I}_l|$ slots of variable lengths, where

$|\boldsymbol{I}_l|$ is the cardinality of $\boldsymbol{I}_l$. For instance, line 1 can perform four different tasks in Figure 3.2, so $NK_1 = 4$. Similarly, $NK_2 = 2$ and $NK_3 = 3$. Thus, each period has NK_l slots on line l, and a line can have at most one campaign for a task during a period. This is mainly to minimize the time wasted during campaign transitions, but it may also result in higher inventory costs. The profit boost due to the former may outweigh the loss due to the latter. We number the slots in each period as $k = 1, 2, \ldots, NK_l$ as shown in Figure 3.2 and define T_{klt}^S and T_{klt}^E respectively as the start and end times of slot k during period t on line l. Note that the slots within a period are not identical across production lines.

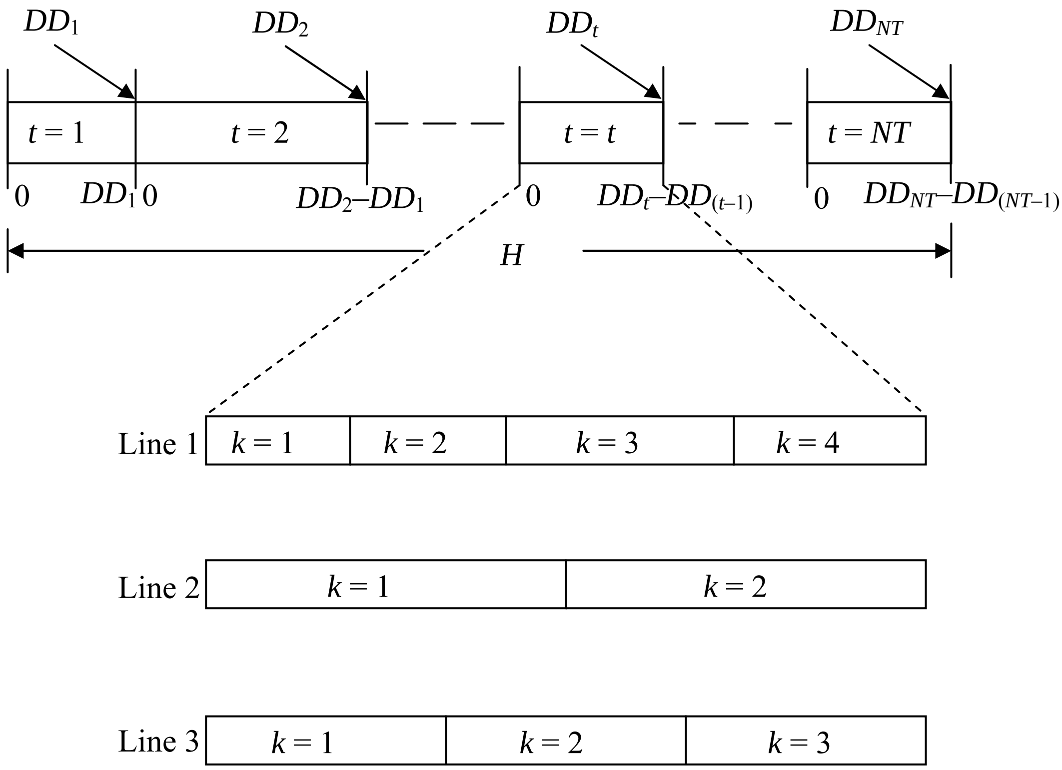


Figure 3.2: Schematic of time periods and slots within a time period.

The supply chain planning model features two classes of constraints: intra-facility and inter-facility. The former deal with the assignment and sequencing of tasks on lines and their timings, while the latter deal with the flow of materials in and out of the facility, material stocks, and demand fulfillment. Unless otherwise stated, we write each constraint for all the valid values of its defining indexes.

3.2.1 Intra-Facility Constraints

The first block of constraints in this class assigns tasks to slots.

3.2.1.1 Task Assignments and Campaign Lengths. To assign tasks to slots, we use the following binary variable:

$$Y_{iklt} = \begin{cases} 1 & \text{if line } l \text{ performs task } i \text{ in slot } k \text{ of period } t \\ 0 & \text{otherwise} \end{cases}$$

A slot cannot have more than one task, so,

$$\sum_{i \in \boldsymbol{I}_l} Y_{iklt} \leq 1 \tag{3.1}$$

Defining NC_{ilt} as the number of campaigns of task i on line l in period t, we write,

$$NC_{ilt} = \sum_{k=1}^{NK_l} Y_{iklt} \qquad i \in \boldsymbol{I}_l \tag{3.2}$$

As we allow at most one campaign per task in a period, we need $NC_{ilt} \leq 1$.

Let CL_{klt} denote the length of slot k on line l in period t, and CL_{iklt} denote the time that we devote to task i within that slot length. Since the sum of such times must equal the slot length, we have,

$$CL_{klt} = \sum_{i \in \boldsymbol{I}_l} CL_{iklt} \tag{3.3}$$

Similarly, if a task i does not occur in a slot k of period t on line l, then we set its start time T^S_{iklt} and time usage CL_{iklt} in that slot to zero. Therefore,

$$T^S_{iklt} + CL_{iklt} \leq (DD_t - DD_{t-1}) Y_{iklt} \qquad i \in \boldsymbol{I}_l \tag{3.4}$$

Slot length represents the actual time for which a task uses a line, so the sum of slot lengths must not exceed the total available production time on each line. In other words,

$$\sum_{k=1}^{NK_l} CL_{klt} \leq H_{lt} \tag{3.5}$$

As discussed earlier, long campaign lengths are desirable in practice to avoid lengthy and costly changeovers in pharmaceutical plants. When a period has several campaigns, a campaign may have continued from the previous period, or a campaign may continue into the next. To model the continuation of the last campaign in a period into the next period, we define the following continuous 0-1 variable:

$$YS_{ilt} = \begin{cases} 1 & \text{if task } i \text{ on line } l \text{ continues across } DD_t \\ 0 & \text{otherwise} \end{cases}$$

A campaign can spill over a DD_t, only if it is the last in period t and the first in (t+1). In other words,

$$YS_{ilt} \leq Y_{iklt} \qquad i \in \boldsymbol{I}_l,\ k = NK_l \tag{3.6}$$

$$YS_{ilt} \leq Y_{i1l(t+1)} \qquad i \in \boldsymbol{I}_l \tag{3.7}$$

However, at most one task per line may spill over at any period. Hence,

$$\sum_{i \in \boldsymbol{I}_l} YS_{ilt} \leq 1 \tag{3.8}$$

Now, to ensure minimum campaign lengths, we demand,

$$CL_{iklt} \geq MCL_{il}[Y_{iklt} - YS_{il(t-1)} - YS_{ilt}] \qquad i \in \boldsymbol{I}_l \tag{3.9}$$

Note that the above constraint relaxes as desired, if the task campaign continues from the previous period or into the next period. If this does happen, then we force the sum of the two consecutive campaign lengths for that task to exceed the minimum length as follows:

$$CL_{ikl(t-1)} + CL_{i1lt} \geq MCL_{il} YS_{il(t-1)} \qquad i \in \boldsymbol{I}_l,\ k = NK_l \tag{3.10}$$

We assume that if a campaign spans more than two periods, then its length would automatically exceed the minimum and we do not impose an extension of eq. 3.9 to more than two campaigns.

3.2.1.2 Timing and Precedence. A production plan devoid of timing considerations could be unrealistic. Hence, we must impose constraints on the exact timings (T_{klt}^{S} and T_{klt}^{E}) of the campaigns as well. Firstly, we relate the start time of a slot *k* of period *t* on line *l* to the start times of individual tasks by,

$$T_{klt}^{S} = \sum_{i \in \boldsymbol{I}_l} T_{iklt}^{S} \tag{3.11}$$

If a task *i* does not take place in slot *k*, then eq. 3.4 will make T_{iklt}^{S} zero, so eq. 3.11 will pick up only the nonzero start time. Secondly, we relate the start and end times of campaigns by,

$$T_{klt}^{E} = T_{klt}^{S} + CL_{klt} \tag{3.12}$$

Thirdly, a campaign cannot start on a line, until the preceding one has ended, so,

$$T_{(k+1)lt}^{S} \geq T_{klt}^{E} \qquad k < NK_l \tag{3.13}$$

An important timing consideration involves the precedence relationships among various tasks in the recipe diagram. If a task *i* precedes another task *i′* in the recipe diagram, then the latter cannot start until the former has produced sufficient amounts of intermediates that the latter needs. When the latter occurs in a period later than the former's period, we ensure this by means of inventory constraints discussed later. However, if the two occur in the same period, then we need constraints to ensure that the latter starts and ends after the former. In order to impose these task orderings, we demand that the start (end) time of task *i* on any line *l* precede the start (end) time of *i′* on the same or any other line *l′* by some amount $d_{ii'}$. Therefore,

$$d_{ii'}(NC_{ilt} + NC_{i'l't} - 1) + \sum_{k=1}^{NK_l} T_{iklt}^{S} \leq \sum_{k=1}^{NK_l} T_{i'kl't}^{S} \qquad i \in \boldsymbol{I}_l,\ i' \in \boldsymbol{I}_{l'} \tag{3.14}$$

$$d_{ii'}(NC_{ilt} + NC_{i'l't} - 1) + \sum_{k=1}^{NK_l}(T^S_{iklt} + CL_{iklt}) \le \sum_{k=1}^{NK_l}(T^S_{i'kl't} + CL_{i'kl't}) \quad i \in \boldsymbol{I}_l,\, i' \in \boldsymbol{I}_{l'} \tag{3.15}$$

3.2.1.3 Production Amounts. The campaign lengths CL_{iklt} include the changeover times, which are usually significant in pharmaceutical plants. To compute the useful production time of task *i*, we must subtract the changeover times from the campaign lengths. To this end, we define PT_{ilt} as the actual time for which task *i* produces on line *l* during period *t*:

$$PT_{ilt} = \sum_{k=1}^{NK_l} CL_{iklt} - CT_{il}\,(NC_{ilt} - YS_{il(t-1)}) \qquad i \in \boldsymbol{I}_l \tag{3.16}$$

where CT_{il} denotes the changeover time required for task *i* on line *l*. Note that the assumption of at most one campaign per task is pivotal to the above constraint. Furthermore, the constraint assumes that changeovers cannot split between periods. Although we can remedy this assumption, we would need to double the binary variables. For a planning problem, this seems excessive detail, as the scheduling problem will fix the actual campaign timings precisely. Now, the amount PQ_{mlt} of material *m* produced or consumed by task *i* on line *l* during period *t* is:

$$PQ_{mlt} = \sum_{i \ni m \in \boldsymbol{M}_i,\, i \in \boldsymbol{I}_l} \frac{\sigma_{mi}}{\sigma_{\mu_i i}}[R^L_{il} PT_{ilt} + DQ_{ilt}] \tag{3.17}$$

$$DQ_{ilt} \le (R^U_{il} - R^L_{il})\,PT_{ilt} \qquad i \in \boldsymbol{I}_l \tag{3.18}$$

3.2.1.4 Validation Time. In practice, when a pharmaceutical plant introduces a new product, it must undergo a scale-up phase of 2 to 3 months before it can begin commercial production of that product. During this period, the plant personnel tune the tasks of the new product to achieve the desired quality targets. Once this scale-up is completed, the plant must produce 4 to 5 batches of the new product to get a regulatory approval for commercial production. The material produced during this validation period cannot be sold to the customers. Hence, this time resembles a changeover time and we must subtract it from the campaign length to get the actual production time. Let QT_{il} be this validation time required for task i on line l. To account for this time in computing the actual production time, we introduce a continuous 0-1 variable as follows:

$$Z_{iklt} = \begin{cases} 1 & \text{if line } l \text{ performes task } i \text{ for the first time in slot } k \text{ of period } t \\ 0 & \text{otherwise} \end{cases}$$

Now, eq. 3.16 becomes,

$$PT_{ilt} = \sum_{k=1}^{NK_l} CL_{iklt} - CT_{il}\,(NC_{ilt} - YS_{il(t-1)}) - QT_{il} \sum_{k=1}^{NK_l} Z_{iklt} \qquad i \in \boldsymbol{I}_l \qquad (3.16a)$$

The validation run for a task i can occur at most once on each line, so

$$\sum_t \sum_{k=1}^{NK_l} Z_{iklt} \le 1 \qquad i \in \boldsymbol{I}_l \qquad (3.19)$$

Furthermore, it should occur in the very first period that task i occurs and never again. Therefore,

$$Z_{iklt} \ge Y_{iklt} - \sum_{t'<t} \sum_{k'=1}^{NK_l} Z_{ik'lt'} \qquad i \in \boldsymbol{I}_l \qquad (3.20)$$

This completes the intra-facility constraints in our model. We now discuss the inter-facility constraints.

3.2.2 Inter-facility Constraints

If I_{mt} denotes the inventory of material m at the end of period t, then

$$I_{mt} = I_{m(t-1)} + OQ_{mt} + \sum_l PQ_{mlt} - \sum_c SQ_{mct} \qquad (3.21)$$

where, OQ_{mt} is the amount of m outsourced during t and SQ_{mct} is the amount of m supplied to customer c during t. Note that the inventory position is after the shipment of material.

If the plant cannot meet the demand of a material m in any period t, then it carries this shortage over to the next period, i.e.,

$$I_{mct}^{-} \geq I_{mc(t-1)}^{-} + D_{mct} - SQ_{mct} \tag{3.22}$$

where, D_{mct} is the demand of material *m* by customer *c* during period *t* and I_{mct}^{-} is the shortfall in supply of *m* to customer *c* during *t*. Furthermore, supply from a current period can fulfill the demand from previous periods, but the cumulative supply up to a period should not exceed the total demand until that period plus the initial backlog. Hence,

$$\sum_{t' \leq t} SQ_{mct'} \leq \sum_{t' \leq t} D_{mct'} + I_{mc0}^{-} \tag{3.23}$$

Plants usually keep some safety stocks for most materials, especially the raw materials and final products, as a buffer against unforeseen circumstances. Let I_{mt}^{*} denotes the safety stock for material *m* during period *t*. In order to maintain the safety stock target, we penalize the deviations of inventory below the safety stock target. We obtain these deviations by using,

$$I_{mt}^{\Delta} \geq I_{mt}^{*} - I_{mt} \tag{3.24}$$

In the development so far, eqs. 3.1-3.20 deal with production tasks, whereas eqs. 3.21-3.24 deal with materials. The inventory balance of eq. 3.21 links these two sets of constraints. We have bounds on several variables such as inventory, backlog, supply, and slot start/end times. These are as follows:

$$I_{mt} \leq ST_{m} \tag{3.25}$$

$$I_{mct}^{-} \leq \sum_{t' \leq t} D_{mct'} + I_{mc0}^{-} \tag{3.26}$$

$$SQ_{mct} \leq \sum_{t' \leq t} D_{mct'} + I_{mc0}^{-} \tag{3.27}$$

$$T_{klt}^{S} \leq DD_{t} - DD_{(t-1)} \tag{3.28}$$

$$T_{klt}^{E} \leq DD_{t} - DD_{(t-1)} \tag{3.29}$$

3.2.3 Planning Objective

Maximization of gross profit (revenue – costs) is normally the preferred objective of plant management, as it lets the management produce the most profitable products. The facility F derives its revenue by selling the APIs or intermediates, while its costs arise from changeovers, purchase of materials, production, inventory, material transportation, and penalties for supply and target level shortages.

Let CC_{il} denotes the changeover cost for starting a campaign of task *i* on line *l*. We can estimate this from the time, labor, and materials required to clean/flush the lines and units. In addition, it would also include the cost of waste disposal. For the purchase of materials, we assume a fixed price p_m for material *m*. This includes the purchase, transport, and insurance costs. Let pc_{ml} denotes the cost of producing a unit of material *m* on line *l*. We do not use exact time-averaged inventory, as it results in a nonlinear objective function. Instead, we approximate the inventory costs by a linear function. Let hc_{mt} denotes the cost of holding a unit amount of material *m* for the entire length of period *t*. We compute the inventory cost at the end of a period *t* based on the amount present at the period's start, and that produced during the period. We assess the penalty for the inventory shortfall from target levels at the end of period *t* as $a_{mt} I_{mt}^{\Delta}$. In addition, we take the supply shortage cost of a product as the revenue of that product g_{mc}. With this, the objective of our planning model is:

$$\text{Maximize } GP = \sum_{m,c,t} g_{mc}\, SQ_{mct} - \text{Cost} \tag{3.30}$$

$$\text{Cost} = \sum_{i \in \boldsymbol{I}_l, l, t} CC_{il}\,(NC_{ilt} - YS_{il(t-1)}) + \sum_{m,l,t} (p_m + pc_{ml})\, PQ_{mlt} + \sum_{m,t} p_m\, OQ_{mt}$$
$$+ \sum_{m,c,t} g_{mc}\, I_{mct}^{-} + \sum_{m,l,t} PQ_{mlt} hc_{mt} / 2 + \sum_{m,t} hc_{m(t+1)} I_{mt} + \sum_{m,t} a_{mt}\, I_{mt}^{\Delta}$$

This completes our formulation for the supply chain planning in pharmaceutical plants. It comprises the objective function (eq. 3.30), constraints (eqs. 3.1-3.24), and bounds (eqs. 3.25-3.29).

3.3. Remarks

Though we assume that our model is deterministic in nature, it can handle unexpected events during a planning horizon by means of a rolling horizon strategy. These events may include line failures, unexpected requests for testing/launching of new products, extreme changes in the demand forecasts, and so on. A new product is quite likely to face a situation when its demand may go up or down suddenly. In either extremity, the current production plan would need a revision. Such revisions could minimize the costs arising from a failure or high backlog of a new product. In order to revise the production plan, we redefine periods from the current time onwards and simply update the model with the current status of the facility. This includes initial inventory levels (I_{m0}), initial backlogs (I^{-}_{mc0}) to various customers, campaign lengths (CL_{ikl0}) of the tasks that currently spill over to next periods, and initial values of spill-over binaries (YS_{il0}). This updating ensures the continuity of production activities in the new plan.

With slight modifications, we can apply our model to several extensions. So far, we focused on primary production alone. Secondary production involves mostly semicontinuous operations such as coating, granulation, packaging etc. As discussed earlier, the production is largely order-driven, so transitions are more frequent, but easier. We can assume the campaign mode of operation in the secondary production as well except that the period and campaign lengths will generally be shorter than those in the primary production will. The minimum campaign lengths may also be shorter. However, with appropriate data, our model can easily address the secondary production as well.

We also assumed that the scale-up procedures are complete before a new product enters facility F for commercial production. Hence, we incorporated only the validation times explicitly in our model. However, the scale-up of a new product may also accompany its introduction. Our model can easily include such scale-up tasks as well. In this case, the lines would perform the new tasks to meet the scale-up targets instead of customer needs.

Chapter 4

PLANNING - MODEL EVALUATION

In this chapter, we present three examples to evaluate the performance and illustrate the application of our planning model that we developed in the previous chapter. Example 1 demonstrates the impact of a new product introduction on the operation of a facility. Example 2 highlights the benefits of outsourcing an intermediate, when a facility is overloaded. Lastly, Example 3 shows the results for the re-run model that accounts for the current state of the facility.

4.1 Examples

The following are common for all three examples:

(i) Minimum production rate R_{il}^{L} of task *i* on line *l* is R_{il}^{U} /4.

(ii) Penalty a_{mt} for violating target inventory is twice the holding cost hc_{mt}.

(iii) Only one secondary production site exists, so all the active ingredients are shipped to only one site. Hence, index *c* is redundant.

(iv) Raw materials are available as and when required, so the inventory cost for storing the raw materials is not important.

(v) Gross profits reported are those for the entire horizon

We display the production plans in the examples via Gantt charts with the following format. The rectangles in the Gantt charts represent time slots. The label within each rectangle denotes the material produced in that slot. The slot start/end times are shown underneath each rectangle. The start/end times are with respect to the start of period as time zero. If a task on a line continues across DD_t, then the last slot in period *t* and the first slot in (*t*+1) on that line are merged to show the continuity of production across DD_t.

We used GAMS (Brooke et al., 1998)/CPLEX 7.5 on a Dell PWS650 workstation with Windows 2000 to solve the three examples. Table 4.4 lists the model and solution statistics. All solutions are optimal solutions with 0% gap. All solution times are within 50 s. The runs that involve new product introductions take substantially more CPU times than those without the introductions.

Table 4.1: Maximum production rates, minimum campaign lengths, changeover times, changeover costs, and qualification times of tasks on various lines for the examples.

Task i	Line l	R^U_{il} (h)	MCL_{il} (h)	QT_{il} (h)	CT_{il} (h)	CC_{il} (h)
Example 1						
1	1	20	100	-	2.0	40
2	2	15	110	-	2.0	50
3	3	25	140	-	3.0	70
4	1	15	110	-	3.0	50
5	2	10	120	-	3.0	60
6	3	20	130	-	2.0	65
7	2	20	100	15	2.0	55
8	3	25	125	10	3.0	80
Example 2						
1	1	20	-	-	-	-
2	2	8	-	-	2.0	50
	3	7	-	-	1.5	45
3	4	15	-	-	3.0	70
4	2	5	-	-	2.0	40
	3	5	-	-	2.0	40
5	4	20	-	-	3.0	80
Example 3						
1	1	20	-	-	3.0	45
2	2	16	-	-	2.0	50
	3	7	-	-	1.5	45
3	4	15	-	-	3.0	70
4	2	10	-	-	2.0	40
	3	5	-	-	2.0	40
5	4	20	-	-	3.0	80
6	1	20	-	10	3.0	50
7	2	16	-	12	2.0	45
	3	7	-	12	1.5	50
8	4	15	-	15	3.0	90

Table 4.2: Available production times in periods, and demands, revenues and safety stock targets for products in the examples.

Material m	Demand (mu) in period t: 1	2	3	4	Revenue ($/mu)	Safety Stock Target (mu)
Example 1						
4	2000	1500	6000	5000	1.2	1500
8	1000	1500	6000	2500	0.8	1000
12	1000	1000	1200	1500	2.3	1000
Example 2						
4	1000	1500	6000	2500	1.2	1500
6	2000	1500	6000	5000	0.8	1000
Example 3						
4	1500	6000	2500	-	1.2	1500
6	1500	6000	5000	-	0.8	1000
11	1000	1500	2000	-	2.3	1000

Available production times in periods are H_{l1} = 360 h, H_{l2} = 360 h, H_{l3} = 720 h, and H_{l4} = 720 h for all lines except that H_{13} = 600 h for Example 3.

Table 4.3: Holding costs, storage capacities, and initial inventories of materials in the examples.

	Example 1		Example 2		Example 3		
Material m	Storage Capacity (mu)	Holding Cost (k$/mu/day)	Storage Capacity (mu)	Holding Cost (k$/mu/day)	Initial Inventory (mu)	Storage Capacity (mu)	Holding Cost (k$/mu/day)
1	AA	0.00	AA	0.00	AA	AA	0.00
2	3500	1.30	3500	1.30	1276.4	3500	1.30
3	3000	1.23	3000	1.23	233.5	3000	1.23
4	UL	1.76	UL	1.76	0.0	UL	1.76
5	AA	0.00	4000	1.36	0.0	4000	1.36
6	4000	1.60	UL	1.60	352.0	UL	1.60
7	5000	1.40	3000	1.40	261.0	3000	1.40
8	UL	1.82	-	-	AA	AA	0.00
9	3000	1.50	-	-	0.0	3500	1.35
10	AA	0.00	-	-	0.0	4000	1.28
11	3000	1.28	-	-	0.0	UL	1.80
12	UL	1.90	-	-	-	-	-

UL = Unlimited, AA = Available as and when required. Delays required between task pairs in Examples 2 and 3 are $d_{12} = d_{14}$ = 4 h, d_{23} = 2 h, $d_{43} = d_{45}$ = 3 h, d_{67} = 4 h, and d_{78} = 3 h.

4.1.1 Example 1

This example illustrates the impact of a new product introduction on the production of an existing facility. We consider the facility in Figure 3.1 (see previous chapter) that produces two active ingredients using six production tasks. F has three production lines, and cannot treat more than a certain amount of the waste material m_9 in each period. The planning horizon is one quarter (2160 h) and comprises four periods. The first two periods are 15 days (360 h) each, and the last two are 30 days (720 h) each. Thus, DD_1 = 360 h, DD_2 = 720 h, DD_3 = 1440 h, and DD_4 = 2160 h. Tables 4.1-4.3 list the required data. In this example, we take minimum campaign lengths to be line-dependent and assume $d_{ii'}$ = 0. In addition, we assume that the initial inventories of all materials other than the raw materials (m = 1 and 5) are zero.

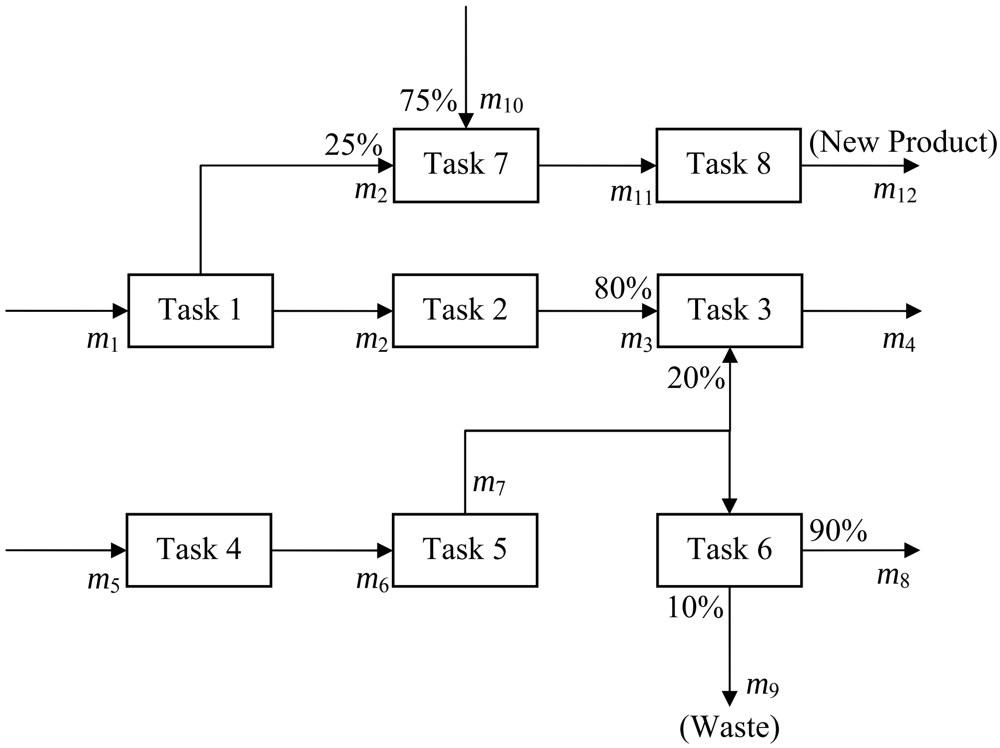
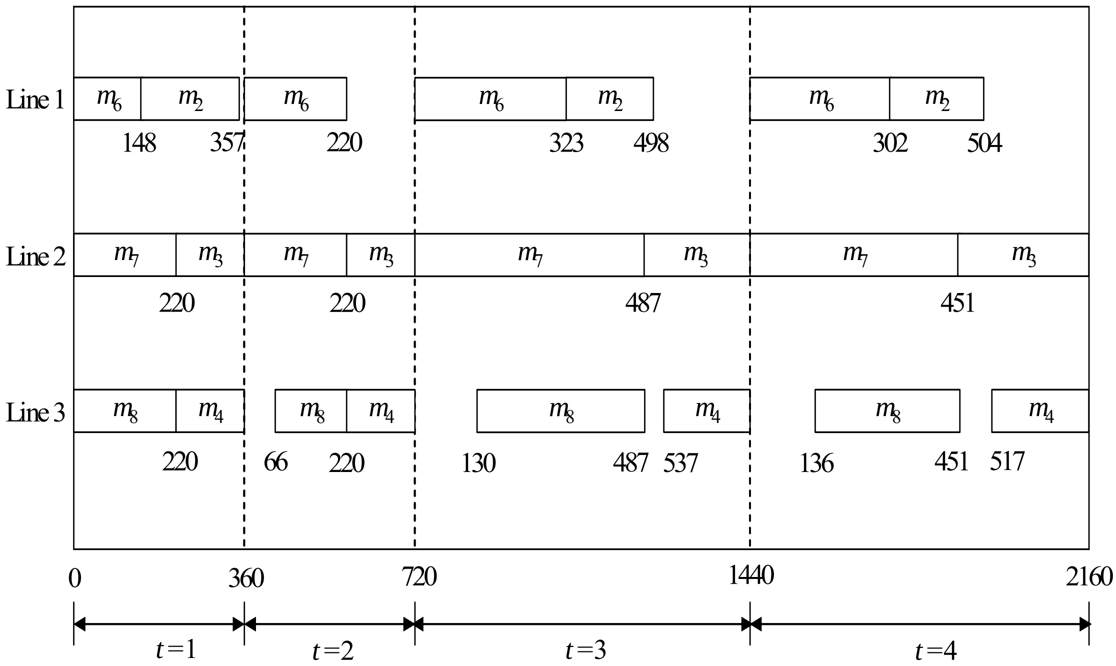


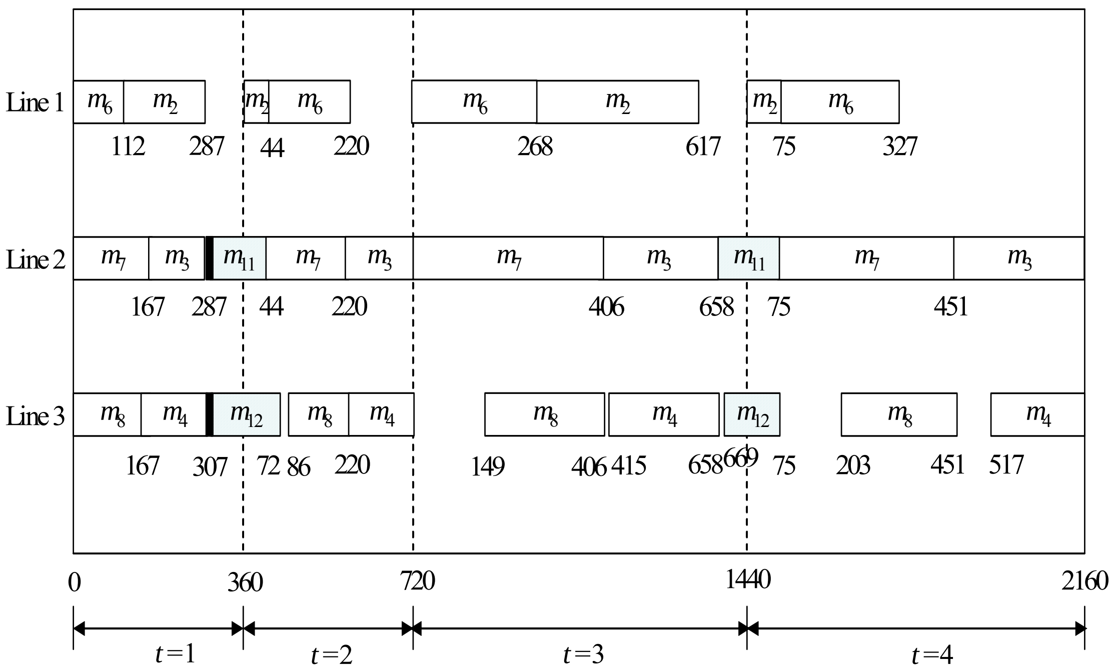
Figure 4.1: Recipe diagram for Example 1 with new product m_{12}.

Figure 4.2(a) shows the current production plan for F with a gross profit of \$20108.1 (Example 1a in Table 4.4). A new product m_{12} is under consideration for production at F. Figure 4.1 shows the recipe diagram for F including m_{12}. The production of m_{12} requires two new tasks i = 7 and 8. Note that line 2 is fully utilized in Figure 4.2(a), so line 2 has no room to produce m_{11}. Therefore, it is clear that F must sacrifice the production of some existing products to accommodate m_{12}. Therefore, the plant management faces several questions: (1) Is it profitable to produce m_{12}, and if so how much to produce? (2) How should the production levels of other products reduce to accommodate m_{12}? (3) What will be the new production plan? Thus, the goal of our model is to decide the optimal combination of products to produce, which will naturally determine if F should or should not produce m_{12}.

In practice, the safety stock level usually depends on the importance of a product and the volatility of its demand. For the flagship/dominant products with well-established markets and high service levels (required), these would be high. However, the same is not true for new products. A new product is highly susceptible to technical/financial failures in spite of the warm welcome assured by the market intelligence. Thus, in spite of their importance to any organization, one must consider the high cost of inventory reclamation in case of new product failures, and cannot risk maintaining high inventories during the initial launch periods. Bowersox et al. (1999) studied the so-called lean-launch strategies based on response-based logistics. They suggested that a lean-launch could cut losses in launch failures by reducing the inventory exposure. Because primary production is the slowest responsive part of the entire pharmaceutical supply chain, it cannot exercise response-based logistics. Hence, a proper combination of the response-based inventory and that required for demand volatility is a better inventory scheme for new products.



(a) Before the introduction of m_{12}



(b) After the introduction of m_{12}

Figure 4.2: Production plans before and after the introduction of m_{12} in Example 1.

To effect a lean-launch, we set the inventory target for m_{12} as the minimum of all target inventories as discussed later in the remarks section. Figure 4.2(b) shows the revised production plan for F after the introduction of m_{12}. The shaded slots represent the new tasks (i = 7 and 8) for m_{12}. Because F performs the new tasks for the first time in period 1, it has to validate them, but in period 1 only. The black color in Figure 4.2(b) highlights these validation times. The gross profit for the new production plan is $25068.9 (Example 1b in Table 4.4), thus the introduction of m_{12} at F can boost profit by 25% ($4960.8). Expectedly, the productions of m_4 and m_8 suffer. Their backlogs increase and inventories decrease (see Table 4.5). m_{12} is so lucrative that the new plan meets its demands in each period at the expense of m_4 and m_8. Clearly, this represents a compelling reason for the management to accommodate m_{12} at F.

Table 4.4: Model and solution statistics for the examples.

Example	CPU Time (s)	Nodes	Iterations	RMILP ($)	MILP ($)	Binary Variables	Continuous Variables	Constraints	Nonzeros
1a	0.5	129	2745	21066.6	20108.1	48	872	1047	2163
1b	28.9	4149	199545	26739.0	25068.9	88	919	1061	3987
2a	1.2	206	7287	17986.0	13688.8	52	605	761	2613
2b	1.5	153	8057	21158.9	16573.4	52	605	761	2617
2c	1.6	213	9388	21864.6	20729.3	52	605	761	2617
2d	1.6	140	8679	21864.6	20729.3	52	605	761	2621
2e	2.0	192	12709	21841.2	20473.2	52	605	761	2621
3	47.7	4816	280687	30400.1	28646.8	93	872	1047	4131

4.1.2 Example 2

This example highlights the value of outsourcing under suitable conditions. We consider a facility similar to that in Example 1, but with some changes in the production recipe as shown in Figure 4.3. The facility in this case produces two products using five tasks. It employs four production lines. Tables 4.1-4.3 also list the data for this example. As in Example 1, we use DD_1 = 360 h, DD_2 = 720 h, DD_3 = 1440 h, DD_4 = 2160 h, identical minimum campaign lengths for all tasks, and zero initial inventories for all but the raw materials. However, we do consider nonzero delays $d_{ii'}$ (see Table 4.3) for a task i preceding i' in the RD in a given period.

Table 4.5: Inventory and backlog details for the examples.

Material	Inventory (mu)			Backlog (mu)					
m	Period	Before	After	Period	Before	After	Period	Before	After
Example 1a,b (before & after introdution of m_{12})									
4	1	587.5	217.5	-	-	-	-	-	-
-	2	1500.0	0.0	-	-	-	-	-	-
8	1	487.3	76.85	2	0.0	99.4	4	1310.0	3596.0
-	2	506.0	0.0	3	1945.0	3556.0	-	-	-
Example 2a,b (before and after outsourcing m_3 only)									
4	2	1546.4	3112.7	1	-	278.2	-	-	-
6	1	352.0	569	2	1148.0	931.0	4	3548.3	1335.0
-	-	-	-	3	2970.3	757.0	-	-	-
Example 2c,d (before and after outsourcing m_5 only or both m_3 and m_5)									
4	1	0.0	305.7	-	-	-	-	-	-
-	2	1546.4	1886	-	-	-	-	-	-
-	3	0.0	727	-	-	-	-	-	-
6	1	352.0	2500	2	1148.0	0.0	4	3548.3	0.0
-	2	0.0	1000	3	2970.3	0.0	-	-	-
-	3	0.0	11	-	-	-	-	-	-
Example 2e (before and after outsourcing unlimited m_3 and limited m_5)									
4	2	1546.4	1771.7	-	-	-	-	-	-
6	1	352.0	1252	2	1148.0	0.0	4	3548.3	0.0
-	-	-	-	3	2970.3	0.0	-	-	-
Example 3 (Inventory and backlog profiles)									
4	1	480.0	-	2	1764.8	-	3	2030.2	-
6	-	-	-	1	181.4	-	3	2235.6	-
-	-	-	-	2	1715.0	-	-	-	-

Figure 4.4(a) shows the current production plan with no outsourcing. The plan has a gross profit of $13688.8 (Example 2a in Table 4.4). In the current plan, lines 2

and 3 are the bottlenecks and make it impossible to meet the demand of m_6 in periods 2, 3 and 4 (see Table 4.5). By outsourcing m_3 or m_5 or both, it should be possible to increase the production of m_6, for which line 4 has unused capacity. The plant management faces the decision of which intermediates to outsource and in what amounts. Given the choices of outsourcing, our model can identify the optimal decision including which intermediates should be outsourced, when and in what amounts. However, to study the impact of outsourcing each intermediate or all of them at a time, we consider three scenarios. In the first scenario, F can outsource only m_3, in the second, it can outsource only m_5, and in the last, it can outsource both. In all three cases, the optimal outsourcing quantity is the one that gives the maximum gross profit. Hence, we impose no upper limit on the outsourcing quantity. For these studies, we use a price of \$0.5 per mu (mass unit) for both m_3 and m_5.

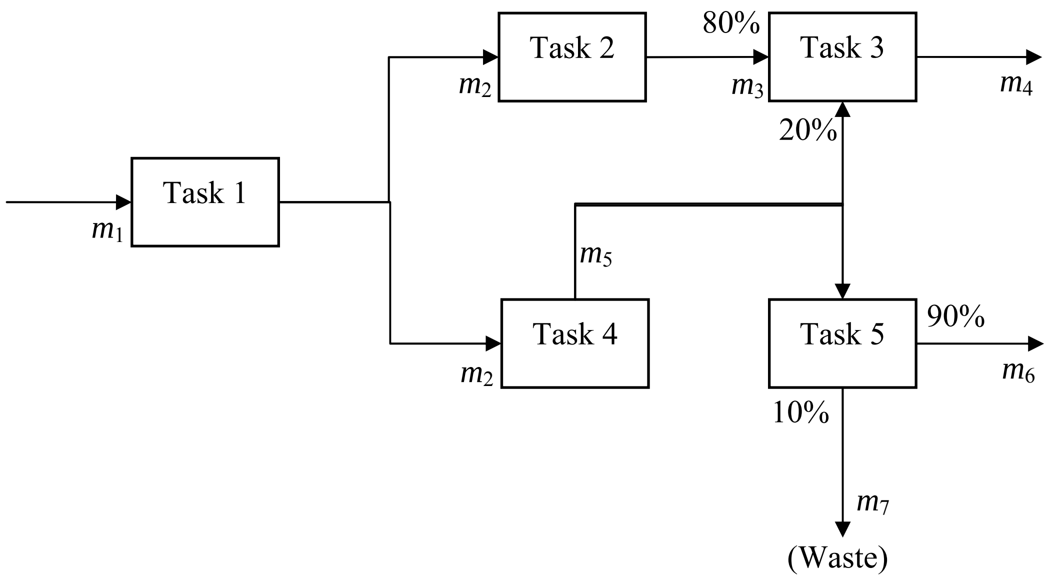
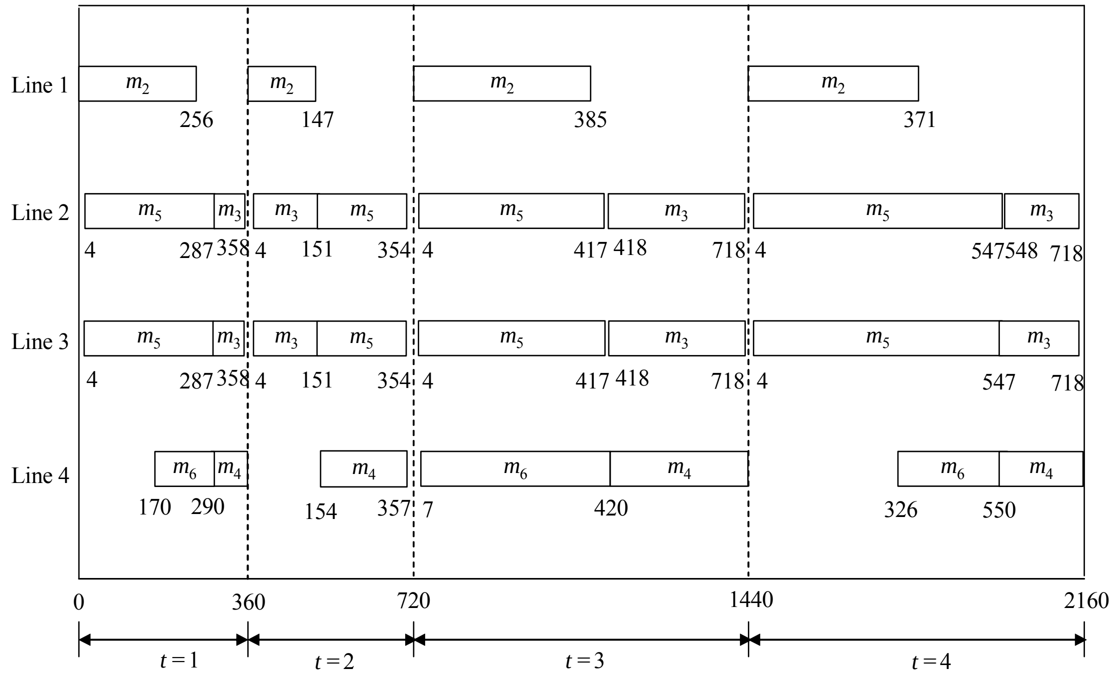
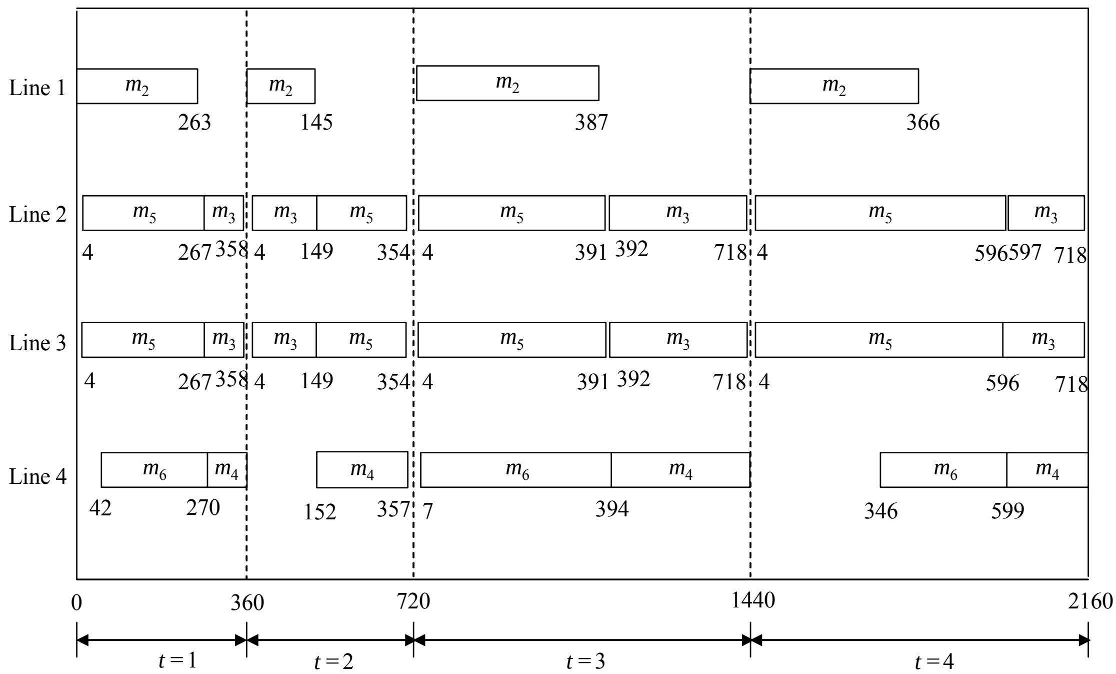


Figure 4.3: Recipe diagram for Example 2.



(a) Before outsourcing m_5



(b) After outsourcing m_5

Figure 4.4: Production plans before and after outsourcing m_5 for Example 2.

In the first case, where F can outsource m_3 only, backlogs (see Table 4.5) of m_6 in periods 2, 3, and 4 reduce to 931, 757, and 1335 mu respectively. However, period 1 still shows a backlog of 278.2 mu for m_4. Although outsourcing m_3 does not eliminate the backlogs fully, it does reduce them. The quantities of m_3 outsourced in periods 1 and 2 are 389 and 3825 mu respectively, while the gross profit for this scenario increases by roughly 21% to $16573.4 (Example 2b in Table 4.4).

In the second case, where F can outsource m_5 only, the backlogs (see Table 4.5) of both m_4 and m_6 disappear in all periods. The outsourced quantities of m_5 are 2652 and 1246 mu in periods 1 and 3 respectively. The gross profit of F increases by roughly 50% to $20729.3 (Example 2c in Table 4.4). This is roughly 25% higher than the first case, so outsourcing m_5 is more profitable than outsourcing m_3. Figure 4.4(b) shows the resulting production plan after outsourcing m_5. In the optimal plan, the campaign of m_6 on line 4 starts immediately after that of m_4 in period 2. However, because the campaign length of m_6 is just 3 h in period 2, this portion of the campaign is invisible in period 2. However, the same campaign continues with a discontinuity due to the delay time required between m_5 and m_6 in period 3. This delay is inconsequential as far as the actual production schedule is concerned and one can suitably eliminate this while doing detailed production scheduling.

In the third case, F can outsource any or both m_3 and m_5, but the optimal outsourcing solution (Example 2d in Table 4.4) is identical to that in the second case. Given the options of outsourcing both m_3 and m_5, the model outsources m_5 only and no m_3. Because both m_4 and m_6 share m_5 (see Figure 4.3), outsourcing m_5 elevates the production levels of both m_4 and m_6.

If F can outsource only a limited amount of m_5 but unlimited amount of m_3, then one may want to know the amounts of backlogs that F can avoid. To study this case, we impose a maximum limit of 1000 mu per period on m_5. In this case, F outsources 1000 mu of m_5 in periods 1, 2, and 3, and 642 mu in period 4. Moreover, it outsources 592 mu of m_3 in period 2. Interestingly, this eliminates the entire backlog of m_6 (see Table 4.5). The gross profit in this case is \$20473.2 (Example 2e in Table 4.4), which although comparable to cases 2 and 3 is still less. We expected this, as the optimal solution is the one in cases 2 and 3.

The decision to outsource requires several considerations. A facility may consider outsourcing an intermediate, when it is unable to meet the demands of its products with the existing equipment. It may also consider outsourcing, when it is more profitable to use the facility to produce a new product rather than a nearly off-patent product. However, at times, it may not be acceptable to outsource some intermediates due to business reasons. In any case, only the intermediates produced by the bottleneck lines are the potential candidates for outsourcing.

4.1.3 Example 3

In this example, we consider several plant changes that force a revision of the production plan. We consider a facility very similar to the one in Example 2. At the end of period 1, we assume that the following changes occur in the facility:

(i) The plant wishes to introduce a new product m_{11} as shown in Figure 4.5.

(ii) The capacity of line 2 doubles due to a retrofit project.

(iii) 120 h are required near the end of period 3 on line 1 to test a new product. Hence, the available production time on line 1 in period 3 is 600 h instead of 720 h.

The end of period 1 or t = 360 h now becomes the start of the scheduling horizon, and the state of the facility at this point (see Figure 4.4a) becomes current state. Thus, period 2 becomes period 1, period 3 becomes period 2, and so on. Tables 4.1-4.3 provide the remaining data.

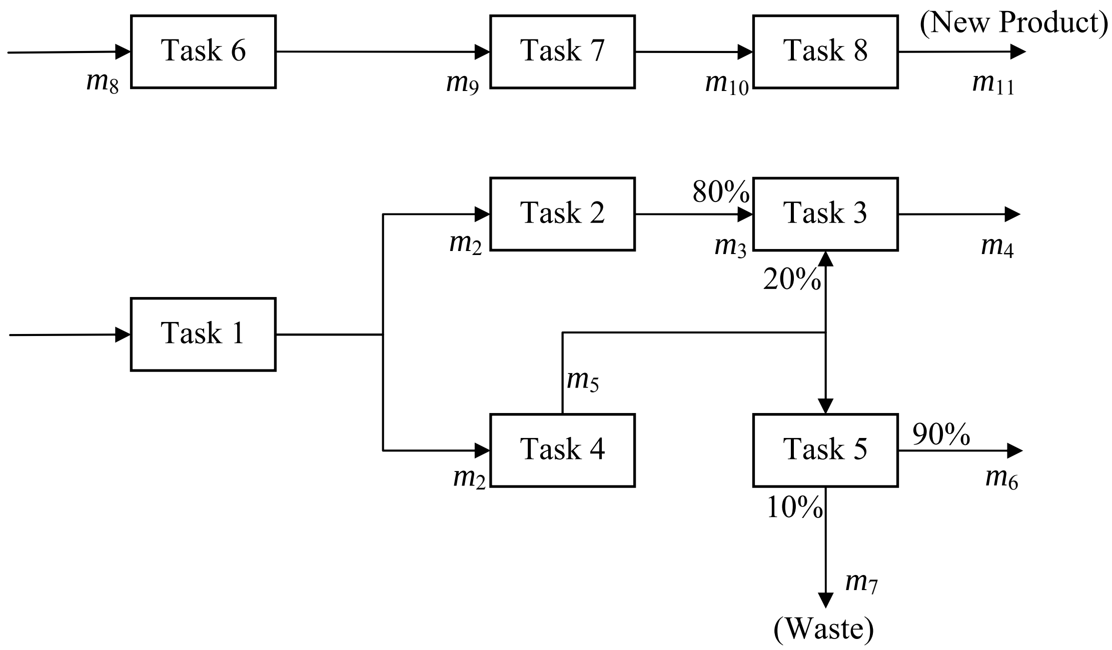


Figure 4.5: Recipe diagram for Example 3 with new product m_{11}.

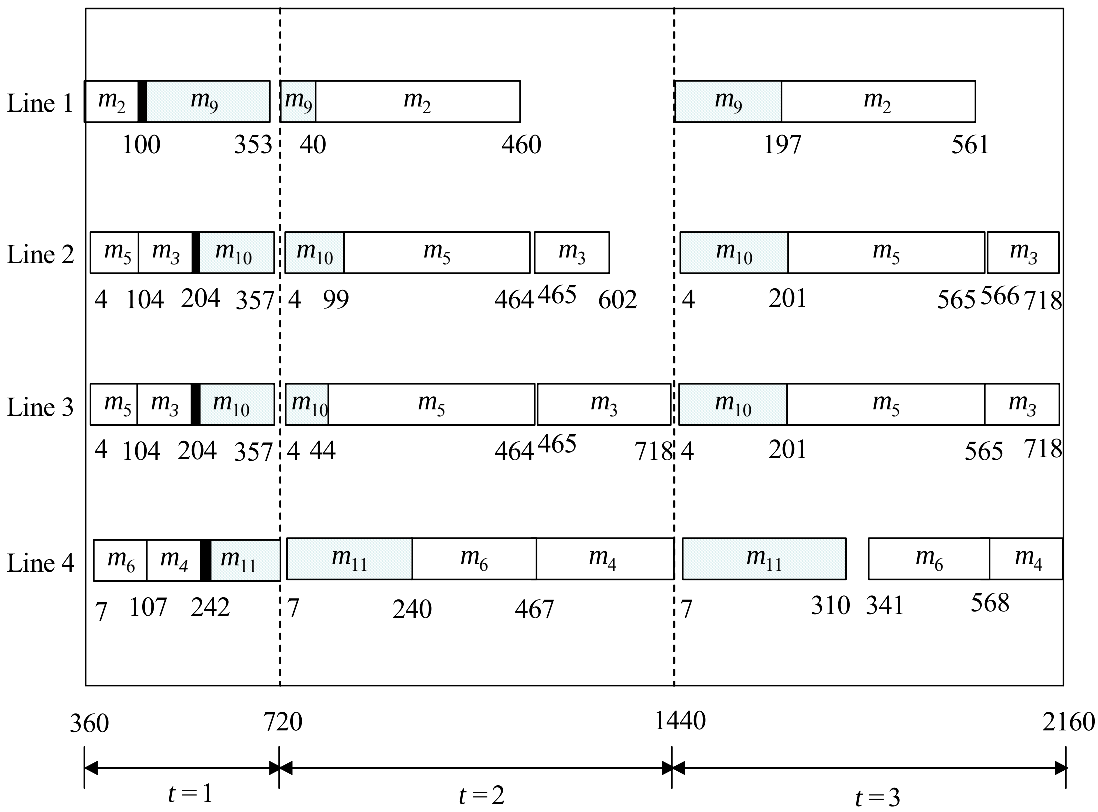


Figure 4.6: Production plan for Example 3.

Figure 4.6 shows the revised production plan that incorporates the above changes in the facility. The slots with slanted line-pattern represent the new tasks related to m_{11}, while those filled with black represent the validation times. The gaps that occur in the performance of new tasks across the DD_2 are due to the delay times ($d_{ii'}$) required between successive tasks in a recipe, when they take place in a given period. Based on the available inventories of the intermediates, the plant operation can remove these gaps easily during scheduling. Again, as in Example 1, the new product is highly lucrative, and its production is at the expense of existing products. The backlogs of m_4 and m_6 occur in all periods as shown in Table 4.5. However, the gross profit still increases to \$28646.8 (Example 3 in Table 4.4) as compared to \$13688.8. Thus, it is profitable for the plant to accommodate m_{11}.

4.2 Conclusion

We have addressed an important and common supply chain planning problem to assess the feasibility as well as profitability of introducing new active ingredients or intermediates in a given pharmaceutical plant. In the previous chapter, we developed a single-plant-centric, multi-period, MILP model that allows complex API-recipes with multiple intermediates, outsourcing of existing intermediates, material movement among different production/supply/demand facilities, validation times for new tasks, minimum campaign lengths, line-dependent cleaning, and so on, and considers explicitly the details of campaign sequencing and timing on individual production lines in a pharmaceutical plant. In this chapter, we tested its efficacy on three examples that feature many of the real-life issues of a pharmaceutical plant. It is clear from the evaluation that our model gives reasonably quick solutions (< 50 s) for three examples (see Table 4.4) involving up to twelve materials (intermediates, products, wastes, etc.), four production lines, and up to three months of planning horizon. Hence, we conclude that our planning model can assist in quick, optimal assessments of new product introduction and outsourcing in a pharmaceutical plant.

Chapter 5

SCHEDULING IN PHARMACEUTICAL PLANTS

It is common in pharmaceutical industry that the demands of products in a particular primary/secondary site vary quite often. In such cases, the operation management may not prefer the campaign mode of operation, instead it would resort to a batch mode of operation as the latter can effectively respond to the variable demands. Hence, the decision level lowers from planning to scheduling, where scheduling is more accurate as it accounts for the realistic constraints in the plants. Normally, the scheduling horizon is about few days to few weeks, whereas planning horizon could go up to few years. Now, the short-term demands of products dictate the operations in a plant, and much competition arises among different products. Since the pharmaceutical plants are multipurpose in nature, the major challenges that the plant management faces are how to utilize the available production units very efficiently, which products to produce from when to when so that the customer demands are met satisfactorily and so on. In this chapter, we describe an important scheduling problem for the pharmaceutical plants, present the motivation behind our scheduling work, develop a novel formulation for addressing this important problem, and present some remarks on the proposed formulation.

5.1 Motivation

Some recent attempts (Ierapetritou & Floudas, 1998, 1999; Giannelos & Georgiadis, 2002; Maravelias & Grossmann, 2003a) at scheduling multipurpose batch plants base their MILP formulations on the idea of decoupling tasks from units, which entails replacing the key 3-index binary assignment variables (task on unit at event point or slot) into two sets of 2-index binary variables involving task to event point/slot or unit to event point/slot assignment decisions. The motivation for this decoupling is that it appears to reduce the total number of binary variables in the formulation which in turn may reduce overall solution time.

Although reducing the number of binary variables in a formulation is generally a desirable modeling objective, it is well known that this does not guarantee improved solution times. The ultimate proof in any specific case still lies in the hard evidence of computational performance. Furthermore, the following analysis shows that this type of decoupling strategy does not actually lead to the binary variable reduction that is intended.

Consider an *arbitrary* (multipurpose or otherwise) batch plant with I tasks (i = 1, 2, …, I) and J units. Whether one uses slots or events in a continuous-time formulation, a key scheduling decision is to assign tasks to units at various slots or events. One approach to model this decision is to directly use the following straightforward 3-index binary variables:

$$y(i,j,n) = \begin{cases} 1 & \text{if task } i \text{ begins on unit } j \text{ at event point } n \\ 0 & \text{otherwise} \end{cases}$$

Under the decoupling strategy, the same decision is modeled using the following two 2-index binary variables:

$$w(i,n) = \begin{cases} 1 & \text{if task } i \text{ begins at event point } n \\ 0 & \text{otherwise} \end{cases}$$

$$v(j,n) = \begin{cases} 1 & \text{if unit } j \text{ begins a task at event point } n \\ 0 & \text{otherwise} \end{cases}$$

Let us refer to the above two approaches as non-decoupling and decoupling approaches respectively. Since the latter allows one to eliminate the v-variables from the formulation, it may be argued that it uses fewer binary variables. The justification is that the w-variables have only two indexes, hence their number is of $O(I)$ per event point. In contrast, the y-variables have three indexes and hence their number is of

$O(I \times J)$ per event point, which is expected to be much higher. However, the key assumption that the decoupling approach uses to decouple tasks from units is that each task is performed by a unique unit. If several units can perform a task, then the decoupling approach replaces that task by several tasks that represent unique task-unit combinations. For instance, if units 1 and 2 can perform a task A, then the decoupling approach must define two tasks A-1 and A-2, which are nothing but A-performed-on-unit-1 and A-performed-on-unit-2 respectively.

Now, let $\boldsymbol{J}_i$ denote the set of units that can perform task i. The decoupling approach would replace each task i by $|\boldsymbol{J}_i|$ new tasks, where $|\boldsymbol{J}_i|$ is the cardinality of set $\boldsymbol{J}_i$. Therefore, the total number of tasks in this approach would not be I, but $|\boldsymbol{J}_1| + |\boldsymbol{J}_2| + \ldots + |\boldsymbol{J}_I|$ and the number of w-variables required would be $|\boldsymbol{J}_1| + |\boldsymbol{J}_2| + \ldots + |\boldsymbol{J}_I|$ per event point. The non-decoupling approach would define one binary per event point for each unit that can perform a task i. So, the non-decoupling approach would require $|\boldsymbol{J}_1| + |\boldsymbol{J}_2| + \ldots + |\boldsymbol{J}_I|$ y-variables per event point. Clearly, the number of binary variables is the same in both approaches.

When we assume a unique unit for each task, the index j becomes fixed as soon as we fix i. Then, $y(i, j, n)$ in fact becomes $y[i, j(i), n]$, which is fully equivalent to the 2-index binary variable $w(i, n)$. As we saw earlier, the assumption of a unique unit for each task increases the number of actual tasks to that of $O(I \times J)$, hence the number of w-variables is still the same as the $O(I \times J)$ of y-variables. Another way of viewing this is that since any given plant operation has some inherent degrees of freedom for assigning tasks to units and event points, any complete formulation cannot reduce this inherent freedom. If a formulation were to do so, it may lead to suboptimal solutions. The only difference between the 3-index y-variables and the 2-index w-variables is that the former display the unit information explicitly in terms of j, while the latter hide the same behind *i*. The number of binary variables per event point must be the same; otherwise the formulation cannot give the optimal solution. The decoupling of tasks from units increases the number of binaries by increasing the number of tasks, and at the same time decreases them by eliminating the *v*-variables, but the net effect of these two actions is zero in terms of the number of binary variables.

The aim of this work is to propose a simple and novel formulation that does not decouple tasks from units and uses a slot-based continuous-time formulation for scheduling multipurpose batch plants. As we show later conclusively, in spite of using 3-index binary assignment variables, our formulation requires fewer variables, uses fewer constraints and nonzeros, solves significantly faster, and has the potential to yield tighter relaxed objective than any other existing model (even-based or otherwise). Furthermore, it should not have any problems in addressing several key features of resource-constrained multipurpose batch plants.

5.2 Problem Description

We consider a pharmaceutical batch plant F that produces multiple products using a number of shared production units that constrain the plant operation. We use recipe diagrams (RDs) to describe the production in F, which we feel are a more straightforward extension of Process Flow Diagram (PFD) concept to a batch process. In chapter 3, we discussed about the representation of RD with an example. Here, we use an alternate representation of RD, which provides unit information in addition to the task sequence. In a RD, nodes represent the tasks, arcs represent the various materials, and arc directions represent the task precedence. Hence, RD uses only one set of nodes that denote tasks and obviates the need for using a second set to represent material states. In this sense, RD is a simpler and unambiguous depiction of recipes using only task nodes. A plant in general may involve one or more disjoint RDs.

Figure 5.1 shows the RD for an example pharmaceutical plant producing one product that requires two intermediates. In the figure, D denotes the final product, B and C intermediates, and A the raw material. Each rectangle represents a task and shows information about the suitable units that can process the task. As mentioned above, we use arrows and their directions to represent the materials and task precedence respectively as shown in the figure. Further discussion on RDs appears in the examples section of the next chapter.

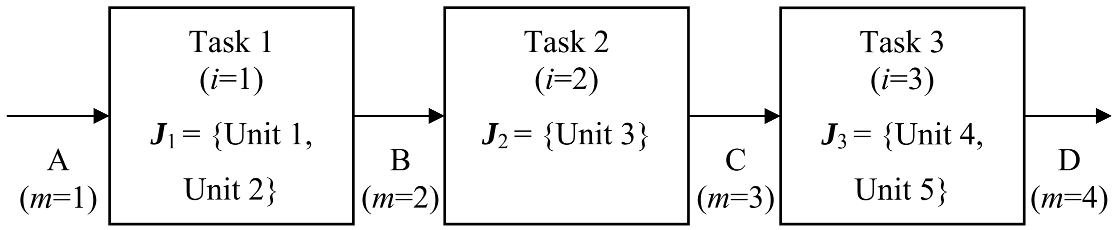


Figure 5.1: Recipe diagram for the motivating example. $\boldsymbol{J}_i$ denotes the set of units that can perform task i.

The facility F houses J (j = 1, 2, …, J) units and performs I (i = 1, 2, …, I) tasks. Each unit j can perform a set $\boldsymbol{I}_j$ of tasks in the RD. Similarly, a set $\boldsymbol{J}_i$ of units can perform a task i. We use index m to represent materials in the RD. Let $\boldsymbol{M}_i$ denote the set of materials ($m \in \boldsymbol{M}_i$) that a task i consumes or produces. $\boldsymbol{M}_i$ includes all the different states of raw materials, intermediates, and final products associated with task i. We use the general mass balance for each task i as,

$$\sum_{m \in \boldsymbol{M}_i} \sigma_{mi}(\text{Material } m) = 0 \tag{5.0}$$

where, σ_{mi} is analogous to the stoichiometric coefficient of a species in a chemical reaction except that it can be in kg/kg units instead of mol/mol. Thus, $\sigma_{mi} < 0$, if task i consumes material $m \in \boldsymbol{M}_i$ and $\sigma_{mi} > 0$, if it produces $m \in \boldsymbol{M}_i$. Furthermore, for each task i, we designate a primary material μ_i, with respect to which we define the extent of task *i*. The batch size of a task *i* is defined as the amount of the primary material μ_i that task *i* consumes or produces in a batch.

For the short-term scheduling of such a multipurpose batch plant, we need to determine:

(i) the optimal sequence and schedule of different tasks on each unit

(ii) the batch size of each batch of each task on each unit at various times

using:

(a) RD for the plant with material and unit requirements

(b) Suitability of units (processing and storage) for tasks, their capacity limits, and batch processing time information

(c) Time horizon *H* for profit maximization or fixed product demands D_m for makespan minimization

(d) Final product revenues, net or otherwise

Although, we consider only two scheduling objectives (maximizing the profit/revenue from the sales of finished products and minimizing the makespan) in this work, other objectives such as minimizing inventory, production, or setup costs and even minimizing the tardiness or earliness can readily be accommodated in the proposed formulation with minor modifications.

We assume the following for the scheduling formulation.

1. Transfer and setup times are lumped into batch processing times of tasks.
2. The batch processing time of task *i* on unit *j* is either a constant (τ_{ij}) or varies linearly with its batch size as α_{ij} + β_{ij}(Batch size), where α_{ij} and β_{ij} respectively are known.
3. Product revenues have accounted for various production costs.

We now develop a continuous-time MILP formulation for the above scheduling problem.

5.3 Formulation

In multipurpose batch plants, it is quite common that multiple tasks share a limited number of production units and resources. Therefore, we need a time representation that can handle the shared resources effectively. Although it is possible to use a representation using asynchronous slots (Lim and Karimi, 2003b) to handle shared resources, we use synchronous slots in this work because they simplify the treatment of shared resources. However, note that the former approach generally requires fewer slots than the latter for a given problem. In this chapter, we present a basic formulation for scheduling multipurpose plants, in which all units are batch units, and no resources other than materials and equipment are required for tasks.

We divide the horizon H into K ($k = 1, 2, \ldots, K$) slots of variable lengths SL_k as in Figure 5.2, with $k = 0$ denoting the slot prior to time zero. The slots are common to or synchronized on all units ($j = 1, 2, \ldots, J$), and we fix K a priori or gradually increase K until we have adequate slots. We denote T_k as the time at which slot k *ends*. $T_0 = 0$ represents the start of the horizon, while $T_K \leq H$ may occur before H. Normally, each T_k corresponds to the start of a task on one or more units, but this need not be so. A task beginning at T_k can end before, at, or after $T_{(k+1)}$ as shown in Figure 5.2. Since a slot k runs from $T_{(k-1)}$ to T_k, we get,

$$T_k = T_{k-1} + SL_k \tag{5.1}$$

Also, the sum of all slot lengths must not exceed the given horizon H. Hence,

$$\sum_{k=1}^{K} SL_k \leq H \tag{5.2}$$

The start of every new task on any unit triggers a slot. However, a new slot may begin, even without a task start. Similarly, several tasks may start at the same time on different units. This type of representation allows us to accommodate extra, redundant slots, when we overestimate *K*. Furthermore, we define a zero task (i = 0) to model the idling of units and to occupy extra, redundant slots. Thus, we have I+1 tasks in our formulation, I real (i = 1, 2, …, I) and one idle (i = 0).

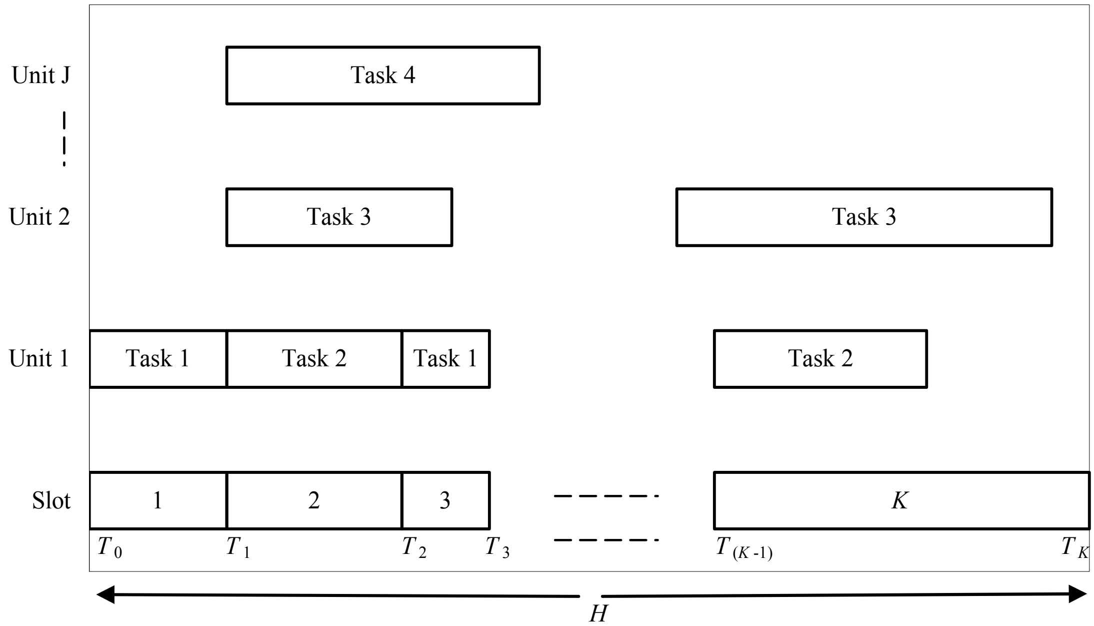


Figure 5.2: Schematic of slots and time points.

With this preamble, we proceed with our primary aim of deciding which tasks to begin/end at what times, on which units, and in how much amounts.

5.3.1 Task Assignments and Slot Lengths

First, we define a binary variable for the start of a task as follows:

$$Y_{ijk} = \begin{cases} 1 & \text{if unit } j \text{ begins task } i \text{ at time } T_k \\ 0 & \text{otherwise} \end{cases} \qquad i \in \mathbf{I}_j,\ 0 \le k < K$$

If a task i merely continues on unit j at T_k, then Y_{ijk} = 0. Based on the binary Y_{ijk}, we define a continuous 0-1 variable Z_{jk} to know if a unit j begins a new task at T_k:

$$Z_{jk} = \begin{cases} 1 & \text{if unit } j \text{ begins a task (including } i = 0\text{) at time } T_k \\ 0 & \text{otherwise} \end{cases} \qquad 0 \le k \le K$$

If $Z_{jk} = 0$, then the current task on unit j at T_k continues. Since at most one task can start on a unit j at any T_k, we can write,

$$Z_{jk} = \sum_{i \in \mathbf{I}_j} Y_{ijk} \qquad 0 \le k < K \qquad (5.3)$$

We assume that all tasks must end at or before T_K, and imagine that a new task will start on all units at T_K, i.e. $Z_{jK} = 1$ for all j in our formulation.

Having modeled the starts of tasks, we now model their sizes and timings. Let B_{ijk} ($i \in \mathbf{I}_j$, $0 \le k < K$) be the batch size of task i that unit j begins at T_k. This refers to the actual amount of primary material μ_i involved in the batch starting at T_k on unit j. If task i does not start at T_k, then $B_{ijk} = 0$, and vice versa. Therefore, we have,

$$B_{ij}^{L}\, Y_{ijk} \le B_{ijk} \le B_{ij}^{U}\, Y_{ijk} \qquad i > 0 \qquad (5.4)$$

where, B_{ij}^{U} and B_{ij}^{L} respectively are the maximum and minimum batch sizes of task i on unit j.

5.3.2 Different Balances

Now, our formulation revolves mainly around four balances involving unit, time, and material inventories in units and storage:

1. Status of a processing unit
2. Processing time of a task in a unit
3. Amount of batch material residing in a unit
4. Inventory in each material storage

We begin with the balance on the use of a processing unit.

5.3.2.1 Balance on Units. We define the following 0-1 continuous variables:

$$y_{ijk} = \begin{cases} 1 & \text{if unit } j \text{ is continuing to perform task } i \text{ at time } T_k \\ 0 & \text{otherwise} \end{cases} \qquad i \in \mathbf{I}_j, 0 \le k \le K$$

$$YE_{ijk} = \begin{cases} 1 & \text{if unit } j \text{ ends task } i \text{ and releases its batch at time } T_k \\ 0 & \text{otherwise} \end{cases} \qquad i \in \boldsymbol{I}_j,\ 0 \le k \le K$$

A balance on the status of unit j simply means,

$$y_{ijk} = y_{ij(k-1)} + Y_{ij(k-1)} - YE_{ijk} \qquad 0 < k < K \qquad (5.5)$$

Note that y_{ijk} is zero, when task i is not precisely under progress in unit j at T_k. It is zero, when a task begins, ends or is not at all taking place. It becomes one, only *after* a task has begun, and becomes zero exactly when the task ends. In our formulation, we set $y_{ijK} = 0$, so no task can continue beyond the last slot K. Similarly, for an empty plant at the start, we set $y_{ij0} = YE_{ij0} = 0$.

It is clear that unit j cannot start a new task, unless it ends the previous task. Since we have idle tasks, we assume that a new task always starts on a unit at the end of each task, that is,

$$Z_{jk} = \sum_{i \in \boldsymbol{I}_j} YE_{ijk} \qquad 0 < k < K \qquad (5.6)$$

Note that since $Z_{jK} = 1$ and $YE_{ij0} = 0$, we do not enforce the above for $k = K$ and $k = 0$. However, if a task i on unit j must discharge its batch at T_K, then YE_{ijK} will be one, as that would favor the objective. Similarly, a unit j can start a new task only if it is not continuing any task, so

$$\sum_{i \in \boldsymbol{I}_j} y_{ijk} + Z_{jk} = 1 \qquad 0 < k < K \qquad (5.7)$$

Again, we do not enforce the above for $k = K$ and $k = 0$, since $Z_{jK} = 1$, and $y_{ijK} = y_{ij0} = 0$. Lastly, a unit may end a task i and start the same at the same time, but it cannot continue and end or continue and start at the same time. We write these as,

$$y_{ijk} + Y_{ijk} \le 1 \qquad 0 < k < K \qquad (5.8a)$$

$$y_{ijk} + YE_{ijk} \le 1 \qquad 0 < k < K \qquad (5.8b)$$

Eqs. 5.3, 5.5, 5.6, 5.7, and 5.8 force y_{ijk} and YE_{ijk} to be 0 or 1 only (even though we treat them as continuous 0-1), as long as Y_{ijk} are binary. In fact, it is easy to see that eqs. 5.3, 5.6, and 5.7 make eqs. 5.8a-b redundant. Also, with eq. 5.5 in effect, one of eqs. 5.3, 5.6, and 5.7 is redundant. We discuss the impact of eqs. 5.6 and 5.7 on model performance in the discussion section of the next chapter.

5.3.2.2 Balance on Processing Times. Here, we keep track of the duration of a task in progress on a unit. Let t_{jk} denote the time remaining at T_k to complete the task that was in progress during slot k on unit j. As we move from T_k to T_{k+1}, this time will either remain constant (task not in progress during slot k) or decrease (task in progress) by an amount equal to the slot length. Thus, a time balance at T_{k+1} gives,

$$t_{j(k+1)} \geq t_{jk} + \sum_{i \in \mathbf{I}_j} (\alpha_{ij} Y_{ijk} + \beta_{ij} B_{ijk}) - SL_{(k+1)} \qquad k < K \qquad (5.9)$$

The inequality allows a task to continue in a unit even after its completion. Unless a task is in progress at time zero, we set $t_{j0} = 0$. Also, if we do not allow a unit to continue processing beyond slot K, we also set $t_{jK} = 0$. Whenever, a task completes its required duration on a unit, t_{jk} must be zero. We could enforce this simply by using $t_{jk} \leq H(1-Z_{jk})$, but this results in a loose formulation. To get a tighter formulation, we need the third balance.

5.3.2.3 Balance on Batch Amounts. Let b_{ijk} be the amount of primary material μ_i (batch size) that resides in unit j just before T_k and BE_{ijk} is the amount that task i discharges at its completion at T_k. A simple mass balance around unit j gives,

$$b_{ijk} = b_{ij(k-1)} + B_{ij(k-1)} - BE_{ijk} \qquad i > 0, k > 0 \qquad (5.10)$$

If unit j is empty at time zero, then $b_{ij0} = BE_{ij0} = 0$. At T_K, we want all units to be empty, so $b_{ijK} = 0$. Similarly, whenever a unit j is not performing a task i at T_k, b_{ijk} must be zero, and vice versa. In other words,

$$B_{ij}^{L} y_{ijk} \leq b_{ijk} \leq B_{ij}^{U} y_{ijk} \qquad i > 0, 0 < k < K \qquad (5.11)$$

Furthermore, a unit can release a batch at T_k only if its task ends, i.e. $YE_{ijk} = 1$, and vice versa. In other words,

$$B_{ij}^{L}\, YE_{ijk} \le BE_{ijk} \le B_{ij}^{U}\, YE_{ijk} \qquad i > 0,\ 0 < k < K \qquad (5.12)$$

Finally, t_{jk} can be nonzero only when both y_{ijk} and b_{ijk} are so. Therefore,

$$t_{jk} \le \sum_{i \in \boldsymbol{I}_j} (\alpha_{ij} y_{ijk} + \beta_{ij} b_{ijk}) \qquad 0 < k < K \qquad (5.13)$$

Note that the above equation does not violate the maximum value possible for t_{jk}, which is the batch processing time of the task in progress on unit j. Furthermore, it ensures that a task batch does not end, until its duration is over. As mentioned earlier, eq. 5.13 affects the tightness of our formulation considerably.

5.3.2.4 Balance on Material Inventory. Execution of a task will consume and produce materials. We assume that (1) all materials are stored in a storage facility (imaginary or real), (2) the time to transfer to or from this storage is negligible or included in the processing time, (3) a task at its start withdraws the required materials from storage, and (4) a task at its end transfers product materials to storage. Then, the inventory balance for a material m at T_k is,

$$I_{mk} = I_{m(k-1)} + \sum_{i \in \boldsymbol{OI}_m, i \ne 0} \sum_{j \in \boldsymbol{J}_i} \frac{\sigma_{mi}}{\sigma_{\mu_i}} BE_{ijk} + \sum_{i \in \boldsymbol{II}_m, i \ne 0} \sum_{j \in \boldsymbol{J}_i} \frac{\sigma_{mi}}{\sigma_{\mu_i}} B_{ijk} \qquad (5.14)$$

where, I_{mk} is the inventory of material m at T_k, $\boldsymbol{OI}_m$ is the set of tasks that produce material m, $\boldsymbol{II}_m$ is the set of tasks that consume material m, and σ_{mi} is the stoichiometric yield coefficient of material m in the mass balance of task i, which is negative for the raw materials of task i and positive for its products. Eq. 5.14 ensures that a task is never performed, unless the required raw materials are present in their respective inventories. Indirectly, it also governs the precedence of tasks on various units.

Finally, imposing good upper and lower bounds on all variables in the formulation reduces the nodes in the branch & bound solution of MILPs. Therefore, we use the following upper bounds on SL_k, t_{jk}, I_{mk}, and others.

$$SL_k \leq \max_j \left[\max_{i \in \mathbf{I}_j} \left(\alpha_{ij} + \beta_{ij} B_{ij}^U \right) \right] \tag{5.15}$$

$$t_{jk} \leq \max_{i \in I_j} \left(\alpha_{ij} + \beta_{ij} B_{ij}^U \right) \tag{5.16}$$

$$I_{mk} \leq I_m^U \tag{5.17}$$

$$B_{ijk}, b_{ijk}, BE_{ijk} \leq B_{ij}^U \tag{5.18}$$

where, I_m^U is the maximum storage capacity for material m. In addition, all continuous variables are nonnegative,

$$Z_{jk}, y_{ijk}, YE_{ijk}, SL_k, t_{jk}, B_{ijk}, b_{ijk}, BE_{ijk}, I_{mk} \geq 0 \tag{5.19}$$

In addition to the bounds (eqs. 5.15-5.19), eqs. 5.2-5.6 and 5.9-5.14 are all the constraints that we need for our scheduling problem. To complete our formulation, we need a suitable scheduling objective.

5.3.3 Scheduling Objective

Existing literature has used two scheduling objectives. One is the maximization of total revenue, net or otherwise, while the other is the minimization of makespan (Maravelias & Grossmann, 2003b; Shah et al., 1993). In most scheduling problems, the latter seems to be the more difficult objective (Lamba & Karimi, 2002a; Lamba & Karimi, 2002b; Maravelias & Grossmann, 2003a).

Assuming that the plant can sell all the products that it produces, the net revenue or profit from selling the final product inventories at the end of the horizon is,

$$P = \sum_m g_m I_{mK} \tag{5.20}$$

where, g_m is the net revenue or profit per kg or mu (mass unit) of product. For this objective, our formulation for maximizing sales or net profit comprises eqs. 5.2-5.6, 5.9-5.14, 5.20, and the bounds (eqs. 5.15-5.19).

For minimizing the makespan, we modify the formulation slightly. Now, H ceases to be a given parameter. Instead, the plant must satisfy some minimum demands of products. If D_m denotes the demand for material m, then we have,

$$I_{mK} \geq D_m \tag{5.21}$$

Furthermore, we do not need eq. 5.2. Instead, we minimize the makespan given by,

$$MS = \sum_{k=1}^{NK} SL_k \tag{5.22}$$

Thus, the complete model for minimizing the makespan comprises eqs. 5.3-5.6, 5.9-5.14, 5.21, 5.22, and the bounds (eqs. 5.15-5.19).

5.4 Remarks

Some features of our above formulation are noteworthy and distinct from the previous work.

First, in contrast to all the existing continuous-time models of Ierapetritou & Floudas (1998), Giannelos & Georgiadis (2002), and Maravelias and Grossmann (2003a), our formulation has absolutely no big-M constraints. We believe that this is significant, because our experience shows that eliminating the big-M constraints generally improves MILP formulations. In general, event-based formulations in the literature have profusely used big-M constraints. Maravelias and Grossmann (2003a) require several big-M constraints in modeling the duration, finish-time, and time-matching constraints. As in their case, the use of disjunctive programming also results in big-M constraints. In contrast, our formulation needs no ideas such as disjunctive programming or convex-hull reformulations. We believe that the lack of big-M constraints gives our model a computational edge, as we show later in performance evaluation.

Second, our model is much simpler than previous models. As we see in the examples of next chapter, it has substantially fewer constraints and nonzeros. It also uses fewer binary variables. For instance, Maravelias & Grossmann (2003a) use binary variables for the start, end, and continuation of tasks, while we use them only for the starts of tasks. All these contribute significantly to the computational superiority of our model.

Chapter 6

SCHEDULING - MODEL ASSESSMENT

In this chapter, we assess the performance of our scheduling model, proposed in the previous chapter, comparing it with two other models (Maravelias and Grossmann, 2003a; Giannelos and Georgiadis, 2002) in the literature. Firstly, we solve several example scenarios for different objectives to make an unambiguous conclusion. Then, we discuss some miscellaneous aspects of our model, and present some basic criteria required for any model comparison task. Finally, we make some concluding remarks.

6.1 Examples

For the sake of a fair comparison, we implemented our model and those of Maravelias & Grossmann (2003a) and Giannelos & Georgiadis (2002) in GAMS (Brooke et al., 1998). We solved several examples using the three models on DELL GX 270 (Pentium IV 2.8 GHz CPU with 1 GB of RAM) running Cplex 8.1.0 in GAMS 21.2. We evaluate the three models for both scheduling objectives (profit maximization and makespan minimization) to get a better idea of which model is fundamentally better. Moreover, we compare them on several scenarios of each example to get a robust comparison. Furthermore, we compare them for the special case of constant batch processing times, as this case has appeared in the literature. While Table 6.1 gives the task and unit information for all the examples, Table 6.2 gives the material information. We begin with the first objective of profit maximization, and then consider the second of makespan minimization. For each case, we discuss each example individually.

Table 6.1: Limits on batch sizes of tasks and coefficients in the expressions for processing times in the examples.

Recipe Task	Label i	Unit	Label j	τ_{ij}	α_{ij}	β_{ij}	B^L_{ij} (mu)	B^U_{ij} (mu)
				Example 1				
Task 1	1	Unit 1	Unit1	2.0	1.3330	0.013330	-	100
		Unit 2	Unit2	2.0	1.3330	0.013330	-	150
Task 2	2	Unit 3	Unit3	1.5	1.0000	0.005000	-	200
Task 3	3	Unit 4	Unit4	1.0	0.6670	0.004450	-	150
		Unit 5	Unit5	1.0	0.6670	0.004450	-	150
				Example 2				
Heating	H	Heater	HR	1.0	0.6670	0.006670	-	100
Reaction-1	R1	Reactor 1	RR1	2.0	1.3340	0.026640	-	50
		Reactor 2	RR2	2.0	1.3340	0.016650	-	80
Reaction-2	R2	Reactor 1	RR1	2.0	1.3340	0.026640	-	50
		Reactor 2	RR2	2.0	1.3340	0.016650	-	80
Reaction-3	R3	Reactor 1	RR1	1.0	0.6670	0.013320	-	50
		Reactor 2	RR2	1.0	0.6670	0.008325		80
Separation	S	Separator	SR	2.0	1.3342	0.006660	-	200
				Example 3				
Heating-1	H1	Heater	HR	1.0	0.6670	0.006670	-	100
Heating-2	H2	Heater	HR	1.5	1.0000	0.010000	-	100
Reaction-1	R1	Reactor 1	RR1	2.0	1.3330	0.013330	-	100
		Reactor 2	RR2	2.0	1.3330	0.008890	-	150
Reaction-2	R2	Reactor 1	RR1	1.0	0.6670	0.006670	-	100
		Reactor 2	RR2	1.0	0.6670	0.004450	-	150
Reaction-3	R3	Reactor 1	RR1	2.0	1.3330	0.013300	-	100
		Reactor 2	RR2	2.0	1.3330	0.008890	-	150
Separation	S	Separator	SR	3.0	2.0000	0.006670	-	300
Mixing	M	Mixer 1	MR1	2.0	1.3330	0.006670	20	200
		Mixer 2	MR2	2.0	1.3330	0.006670	20	200

6.1.1 Profit Maximization

The objective of the scheduling model is to maximize the profit or net revenue of the pharmaceutical plant. Here, the batch processing times are variable with the batch sizes of the tasks. Table 6.3 summarizes the model and solution statistics for various scenarios of all three test problems under profit maximization.

Table 6.2: Storage capacities, initial inventories, and revenues of materials in the examples.

Material *m*	Example 1			Example 2			Example 3		
	Storage Capacity (mu)	Initial Inventory (mu)	Revenue ($/mu)	Storage Capacity (mu)	Initial Inventory (mu)	Revenue ($/mu)	Storage Capacity (mu)	Initial Inventory (mu)	Revenue ($/mu)
1	UL	AA	0	UL	AA	0	UL	AA	0
2	200	0	0	UL	AA	0	UL	AA	0
3	250	0	0	UL	AA	0	100	0	0
4	UL	0	5	100	0	0	100	0	0
5	-	-	-	200	0	0	300	0	0
6	-	-	-	150	0	0	150	50	0
7	-	-	-	200	0	0	150	50	0
8	-	-	-	UL	-	10	UL	AA	0
9	-	-	-	UL	-	10	150	0	0
10	-	-	-	-	-	-	150	0	0
11	-	-	-	-	-	-	UL	AA	0
12	-	-	-	-	-	-	UL	0	5
13	-	-	-	-	-	-	UL	0	5

UL = Unlimited; AA = Available as and when required

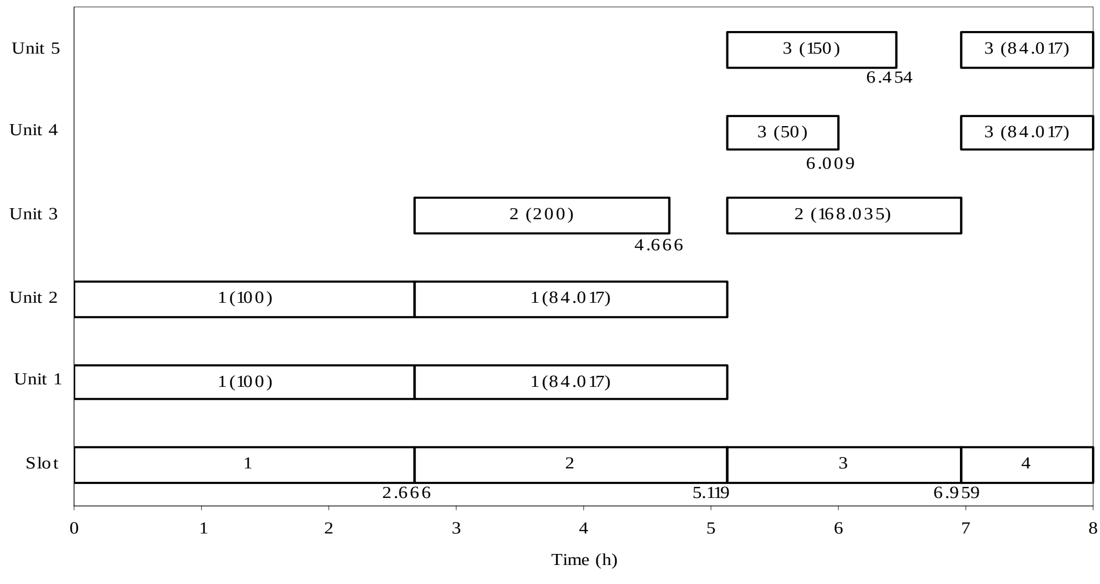


Figure 6.1: Maximum-profit schedule for Example 1 with H = 8 h, K = 4, and variable batch processing times. The numbers within the parentheses denote the batch sizes (mu) of corresponding tasks.

6.1.1.1 Example 1. We used the motivating example in Figure 5.1 (see chapter 5) as the first test example. We solved it for three scenarios. In the first scenario (call it Example 1a), we considered H = 8 h. Increasing K from 4 to 5 did not improve the

MILP objective, so we assumed the solution from $K = 4$ as the optimal. We will do the same for determining the minimum K in all examples. Note that although all the existing formulations in the literature have assumed this for this example, there is no way to generally guarantee that $K = 6$ and beyond will not give a better solution. Figure 6.1 shows the (presumably) optimal schedule from our model. In each figure displaying a schedule in this chapter, we show a separate row of slots that form the basis for each schedule. As mentioned earlier, these slots are common to or synchronized on all units. We use a rectangle to denote each slot. A batch may require more than one slot for its completion. Therefore, we merge these slots into one rectangle that represents the entire task duration. The label within each rectangle denotes the task that the unit performs and the number within the parentheses denotes the batch size (mu) of that task. Moreover, if a task ends before its last slot ends, then we show its exact end underneath the rectangle. Note that the RMILP and MILP objectives (see Table 6.3) are identical for all three models. In addition, the solution time is also almost the same for all three models. This is expected for a simple example such as this. In our opinion, it is foolhardy to claim computational superiority of a model over others based on such trivial examples, which incidentally has occurred commonly in the literature. One should compare models only on difficult problems. Therefore, we cannot conclude from Example 1a that all three models perform equally well.

To make this trivial problem more difficult, we increased H. Thus, in the second scenario (call it Example 1b), we used $H = 12$ h. Our model required $K = 8$, whereas the M&G (Maravelias and Grossmann, 2003a) and G&G (Giannelos and Georgiadis, 2002) models required $N = 9$ and $N = 6$ (event points) respectively. Our model and the M&G model require almost the same order of solution times. However,

Table 6.3: Model and solution statistics for the maximum-profit examples with variable batch processing times.

Model	K/N	CPU Time (s)	Nodes	RMILP ($)	MILP ($)	Binary Variables	Continuous Variables	Constraints	Nonzeros
Example 1a (H = 8)									
Our	4	0.06	12	2000.0	1840.2	40	216	192	643
M&G	5	0.09	1	2000.0	1840.2	50	221	613	1798
G&G	4	0.08	0	2000.0	1840.2	20	81	141	393
Example 1b (H = 12)									
Our	8	23.50	22850	4481.0	3463.6	80	416	408	1359
M&G	9	34.05	28469	4563.8	3463.6	90	397	1089	3884
G&G	6	0.08	19	3890.0	3301.6	30	119	207	593
Example 1c (H = 16, Suboptimal K/N)									
Our	11	4434.44	2655537	6312.6	5038.1	110	566	570	1896
M&G	12	39746.54	18868920	6332.8	5038.1	120	529	1446	5816
G&G	11	3.61	7339	6236.0	4840.9	55	214	372	1093
Example 2a (H = 8)									
Our	4	0.11	5	1730.9	1498.6	48	291	251	930
M&G	5	0.17	11	1730.9	1498.6	80	421	988	3106
G&G	4	0.09	14	1812.1	1498.6	32	142	271	893
Example 2b (H = 10)									
Our	7	112.86	72406	2690.6	1962.7	84	489	458	1686
M&G	8	381.76	137320	2690.6	1962.7	128	673	1567	5665
G&G	6	1.05	2443	3078.4	1860.7	48	208	399	1349
Example 2c (H = 12, Suboptimal K/N)									
Our	6	1.22	524	3002.5	2610.1	72	423	389	1434
M&G	7	3.31	1232	3002.5	2610.1	112	589	1374	4756
G&G	6	0.36	517	3190.5	2564.6	48	208	399	1349
Example 3a (H = 8)									
Our	5	37.19	49765	2100.0	1283.1	85	502	597	1816
M&G	6	84.94	71626	2100.0	1283.1	132	691	1627	5432
G&G	4	0.05	0	1571.9	1150.0	44	198	376	1183
Example 3a (H = 8)									
Our	6	500.98	318290	2560.6	1583.4	102	606	629	2099
M&G	7	588.34	237565	2560.6	1583.4	154	806	1880	6617
G&G	5	0.38	600	2100.0	1274.5	55	244	465	1488
Example 3a (H = 8)									
Our	7	32974.27	13638920	2712.1	1583.4	119	688	859	2598
M&G	8	67156.93	18515053	2712.1	1583.4	176	921	2159	7909
G&G	6	4.33	9057	2809.4	1274.5	66	290	554	1793
Example 3b (H = 12, Suboptimal K/N)									
Our	7	139.85	42013	3464.0	2867.2	119	688	859	2598
M&G	8	461.77	75195	3464.0	2867.2	176	921	2159	7909
G&G	7	28.56	298927	3465.6	2443.2	77	336	643	2098

M&G = Maravelias & Grossmann (2003a), G&G = Giannelos & Georgiadis (2002) our model gives a better RMILP objective ($4481.0 vs. $4563.8) and requires fewer nodes (22850 vs. 28469) than the latter. This suggests that our model has the potential to be tighter. Although, the G&G model seems much faster than the other two and gives a better RMILP objective, it gives a suboptimal solution ($3301.6 vs. $3463.6). We observed no improvement in the objective from the G&G model even for N = 8. Throughout this numerical comparison, we do such confirmation, whenever the G&G model gives an objective inferior to the other two, except when we use suboptimal N deliberately. As we discuss later, suboptimal solutions are a serious flaw in the G&G model. Figure 6.2 shows the maximum-profit schedule from our model for this scenario. It matches the one obtained by M&G.

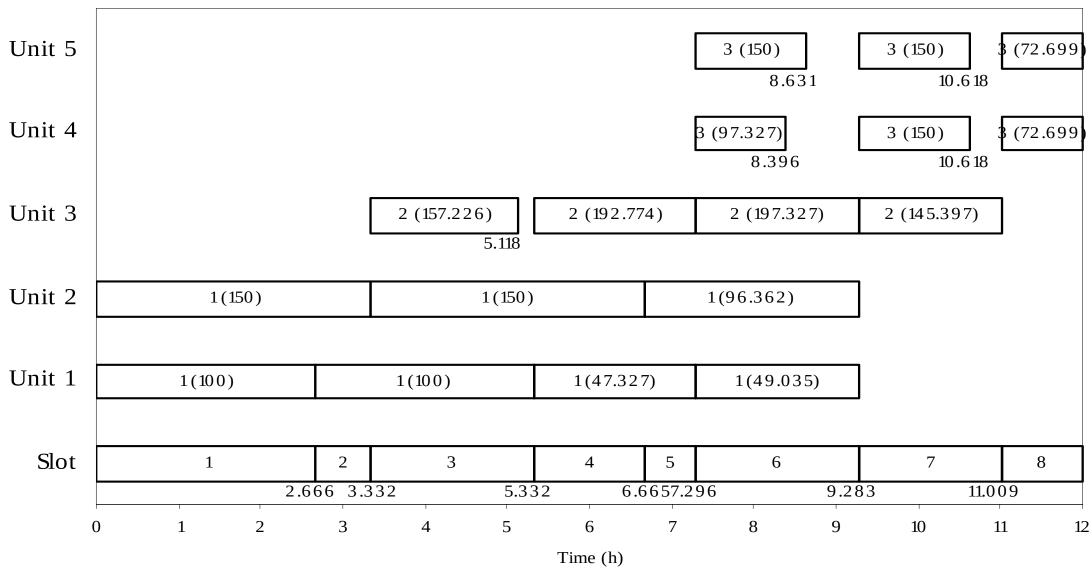


Figure 6.2: Maximum-profit schedule for Example 1 with H = 12 h, K = 8, and variable batch processing times. The numbers within the parentheses denote the batch sizes (mu) of corresponding tasks.

To make the problem even more difficult, we used a third scenario (call it Example 1c) with H = 16 h. However, we did not pursue its solution to the best possible, so we use K = 11 (suboptimal) for our model, N = 12 (suboptimal) for the M&G, and N = 11 (suboptimal) for the G&G. The longer H resulting in increased solution difficulty brings about further resolution between three models, and our model proves clearly faster than the M&G model by almost one order of magnitude (4434 s vs. 39747 s) and uses far fewer nodes (2,655,537 vs. 18,868,920). In addition, as observed earlier, our model gives tighter RMILP objective than the M&G ($6312.6 vs. $6332.8). The G&G model is again faster, but gives a suboptimal schedule ($4840.9 vs. $5038.1).

For all three scenarios (1a, 1b & 1c) of this example, our model uses fewer binary variables (40 vs. 50, 80 vs. 90, 110 vs. 120), and has fewer constraints (192 vs. 613, 408 vs. 1089, 570 vs. 1446) and nonzeros (643 vs. 1798, 1359 vs. 3884, 1896 vs. 5816) than the M&G model. All these result in a faster performance by our model for this example. For reasons discussed at the end of previous chapter, our model requires fewer binary variables than the M&G model. Although this example, as used in the literature, is trivial, we made it difficult enough to conclude reliably that our model outperforms the other two models.

6.1.1.2 Example 2. We now consider the example from Kondili et al. (1993), which has been used extensively in the literature. Figure 6.3 shows the RD for that example unambiguously without resorting to the state nodes as in the STN representation (Kondili et al., 1993). Note that we use a storage task to model the mixing or splitting of the same material streams. However, this storage task does not appear in the formulation, as it has a dedicated unit assigned for the entire horizon, and it has no specified task time. To differentiate this storage task from the normal processing task in the formulation, we use dashed rectangles for the storage tasks. However, we do assume that the transfer times to and from this storage unit are negligible, just as those between any two processing tasks.

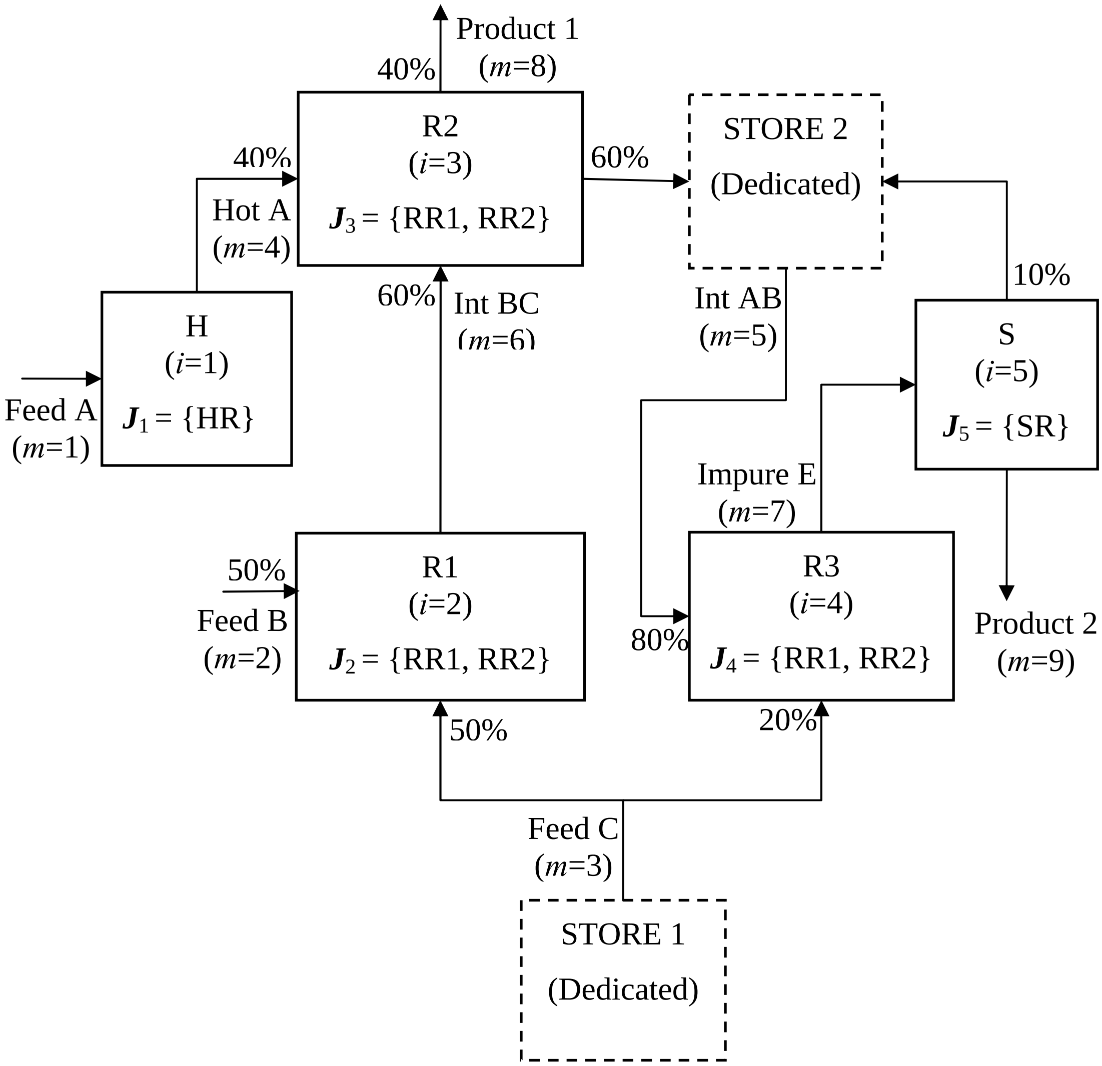


Figure 6.3: Recipe diagram for Example 2. $\boldsymbol{J}_i$ denotes the set of units that can perform task *i*.

Much fuss exists in the literature about the differences in the objective function arising from minor numerical round-offs in problem data with some researchers claiming superiority of their models based on such minute differences. Hence, we used exactly the same parameter values that Maravelias & Grossmann (2003a) used in their paper to make our comparison fair and reliable. As in Example 1, we considered three scenarios for this example. In first scenario (Example 2a), we used H = 8 h and K = 4. Further increase in K did not improve the objective, so we took K = 4 as the minimum required. Figure 6.4 shows the optimal schedule for this scenario. All three models perform equally well for this scenario. In fact, G&G seems to be the best in terms of the RMILP objective. Again, we refrain from drawing any conclusion based on this trivial scenario. The statistics of the M&G and G&G models reported in Table 6.3 are different from those reported by Maravelias and Grossmann (2003a) and Giannelos and Georgiadis (2002). Since we implemented their models, there are some differences in the model and solution statistics. For this scenario, Giannelos and Georgiadis (2002) reported an MILP objective of \$1480.06 (RMILP \$1804.35), whereas we get \$1498.60 (RMILP \$1812.10). This is due to the round-off errors in parameters α_{ij} and β_{ij}. To confirm this, we also used the same values of α_{ij} and β_{ij} as reported by them. Expectedly, we obtained the same MILP and RMILP values (\$1480.06 & \$1804.35 respectively) as reported in Giannelos and Georgiadis (2002).

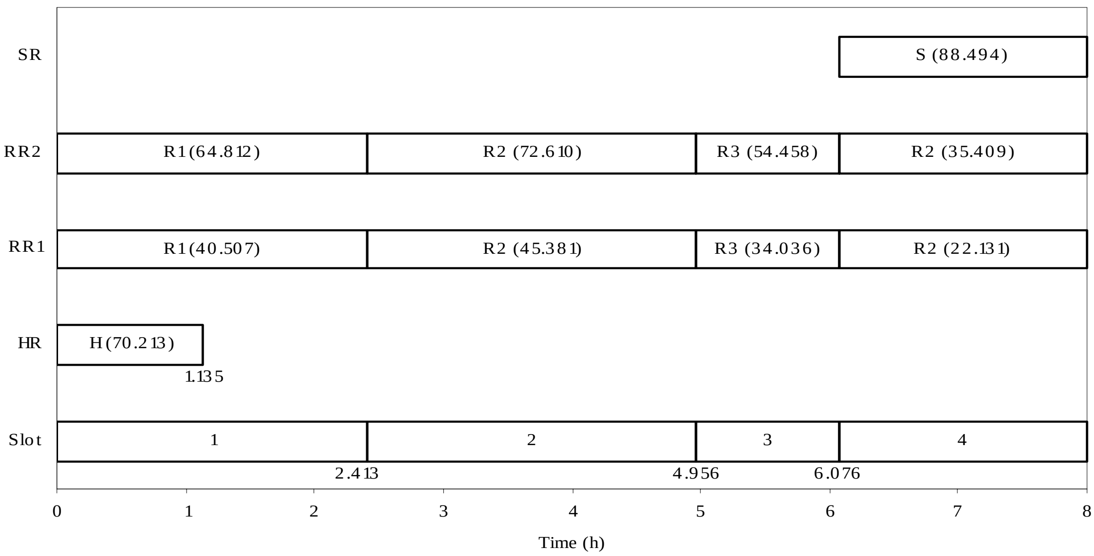


Figure 6.4: Maximum-profit schedule for Example 2 with H = 8 h, K = 4, and variable batch processing times. The numbers within the parentheses denote the batch sizes (mu) of tasks.

For the second scenario (Example 2b), we took H = 10 h. To get an optimal solution, our model needed K = 7 and the M&G needed N = 8, whereas the G&G

required N = 6 only. Our model is faster (113 s vs. 382 s) than the M&G. Again, in spite of being faster, the G&G model gives an inferior solution. In fact, it is reasonable to say that it is faster because it gives a suboptimal solution. Figure 6.5 shows our schedule for this scenario.

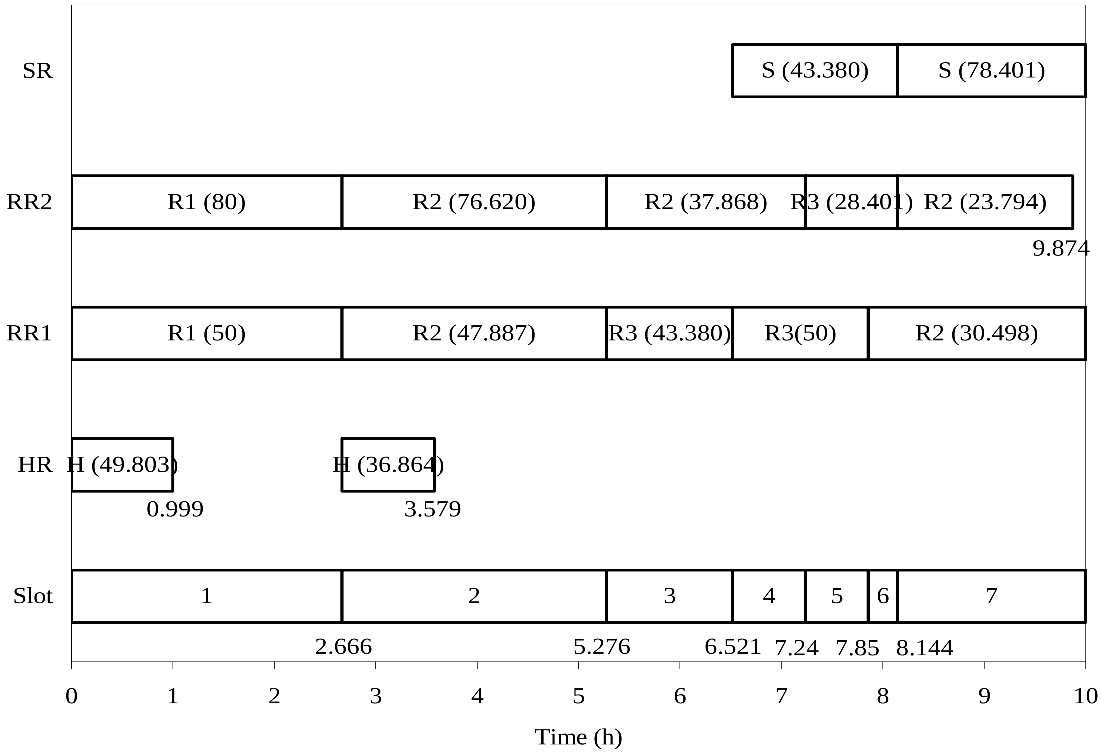


Figure 6.5. Maximum-profit schedule for Example 2 with H = 10 h, K = 7, and variable batch processing times. The numbers within the parentheses denote the batch sizes (mu) of tasks.

For the third scenario (Example 2c), we used H = 12 h, K = 6 for our model, and N = 7 and 6 for the M&G and G&G respectively. Maravelias & Grossmann (2003a) used this scenario to show the impact of their tightening constraints. Note that our formulation needed no additional tightening constraints. For this scenario, our model performs better than the M&G in terms of both solution time (1.22 s vs. 3.31 s) and model statistics (72 vs. 112 binary variables, 389 vs. 1374 constraints, and 1434 vs. 4756 nonzeros). Again, the G&G model gives inferior RMILP and MILP objectives. Note that the statistics reported in Table 6.3 for this scenario differ from those reported by Maravelias and Grossmann (2003a) again due to our own implementation of their model.

From the above three scenarios of this widely studied example, we have demonstrated that our model is superior to the best existing model. However, to reinforce our claim, we take another more complex example (Figure 6.6).

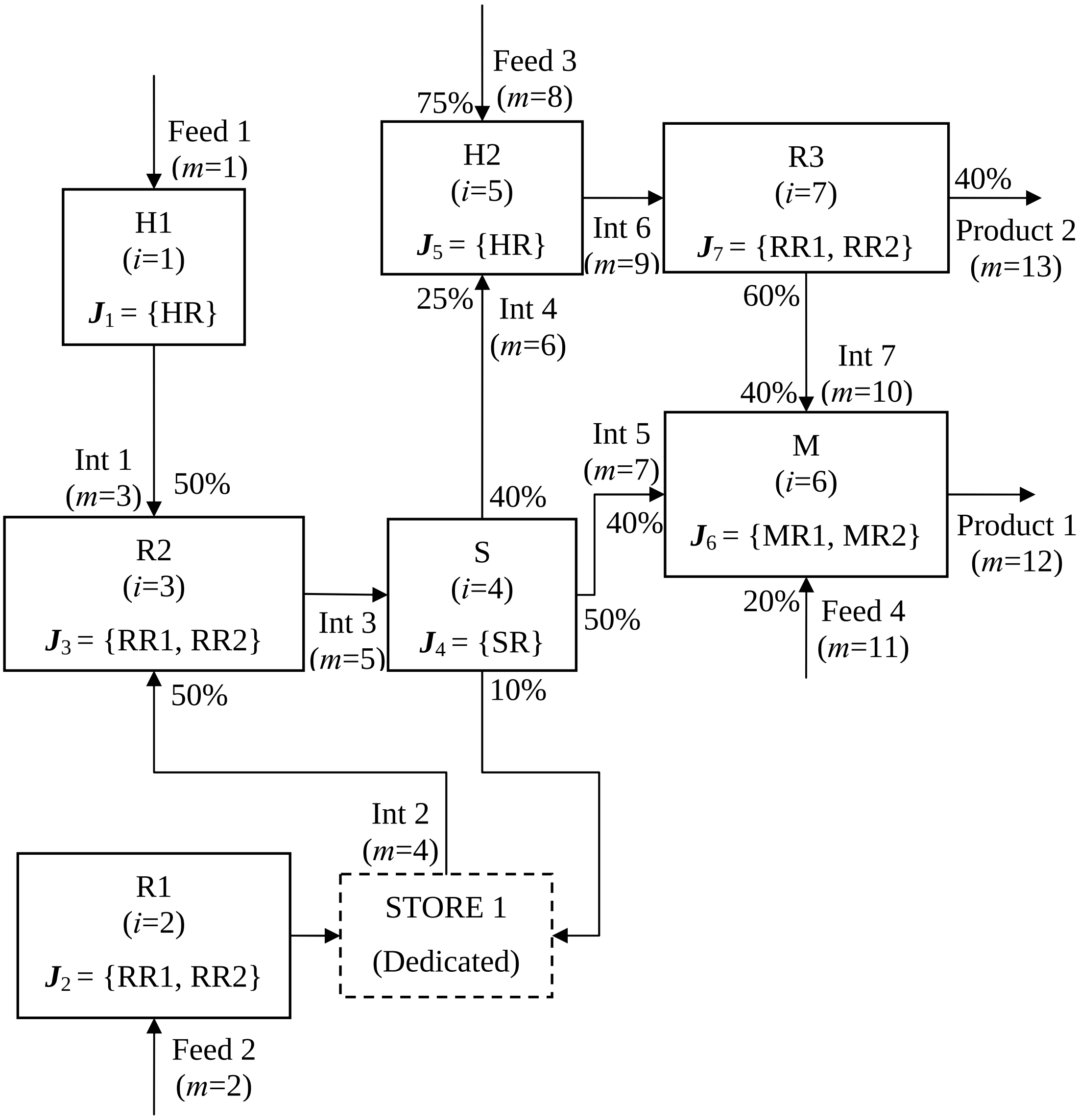


Figure 6.6: Recipe diagram for Example 3. $\boldsymbol{J}_i$ denotes the set of units that can perform task i.

6.1.1.3 Example 3. This example is more complex and comprehensive than that of Kondili et al. (1993), because it involves more units (J = 6), tasks (I = 7), and materials (M = 13). It considers the most common characteristics of a multipurpose plant, namely (1) a unit can perform multiple tasks (2) a task can be performed in multiple units (3) several tasks suitable for a set of units and (4) only one task suitable for a unit. In addition, we assume nonzero initial inventories for some intermediates with $I_{60} = I_{70}$ = 50 mu. It also requires a storage task to imitate the mixing of material Int 2 recycled from task S and produced by task R1. As discussed in Example 2, we assume the transfer times to and from this storage task to be negligible.

For this example, we consider two scenarios. As this is a new example, we present the model and solution statistics for various values of K for the first scenario (Example 3a). We used H = 8 h for this scenario. We solved our model for K = 5, K = 6, and K = 7, and observed that the objective did not improve from K = 6 to K = 7. However, as observed by Castro et al. (2001), the same objective value for two successive Ks does not mean optimality. Hence, we tried K = 8 and K = 9 and observed that the objective value did not improve. Thus, we can safely take the schedule in Figure 6.7 for K = 6 as the maximum-profit schedule. The solution statistics is almost the same for both the M&G and our model for K = 6. But, note that when we increase K to 7, our model is almost twice as fast as the M&G (32974 s vs. 67157 s). The G&G model again gives a suboptimal solution.

In the second scenario (Example 3b), we solved the three models for H = 12 h, and used K = 7 in our model and N = 8 and 7 for the M&G and G&G models respectively. Our model is almost three times faster (140 s vs. 462 s) than the M&G model. Again, the G&G model gives poor RMILP and MILP values. This example also demonstrates that our model handles the more complex problems better than the other literature models.

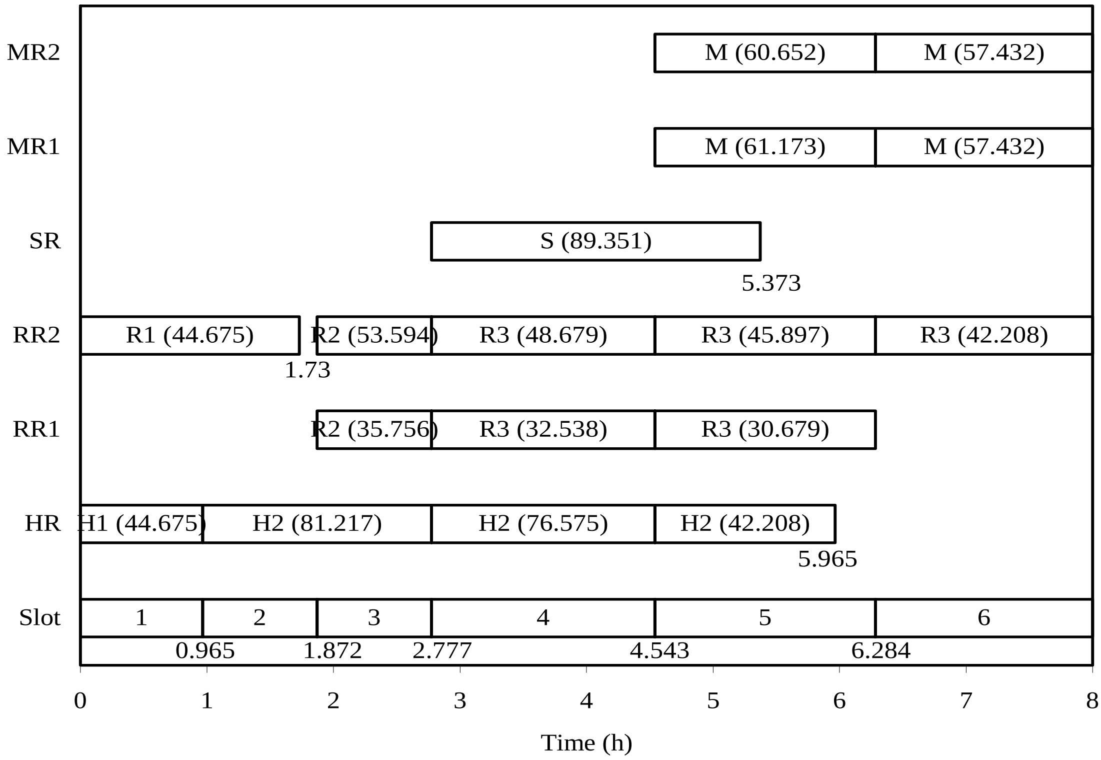


Figure 6.7: Maximum-profit schedule for Example 3 with H = 8 h, K = 6, and variable batch processing times. The numbers within the parentheses denote the batch sizes (mu) of tasks.

Having seen the performance of our model on the maximum-profit problems, we now consider the minimum-makespan problems.

6.1.2 Makespan Minimization

Maravelias and Grossmann (2003a) noted that the solution efficiency of their formulation deteriorated significantly while minimizing the makespan as compared to maximizing the profit. Recently, they (Maravelias and Grossmann, 2003b) addressed separately the minimization of makespan for multipurpose batch plants assuming constant processing times. They modified the discrete-time formulation of Shah et al. (1993), and used assignment binary variables without decoupling, i.e. 3-index binary variables. They proposed an algorithm that can solve problems of medium size and complexity. As we see now, our model seems to maintain its efficiency even for the makespan problems with variable processing times, and can solve moderate-size problems much faster than the model of Maravelias & Grossmann (2003a).

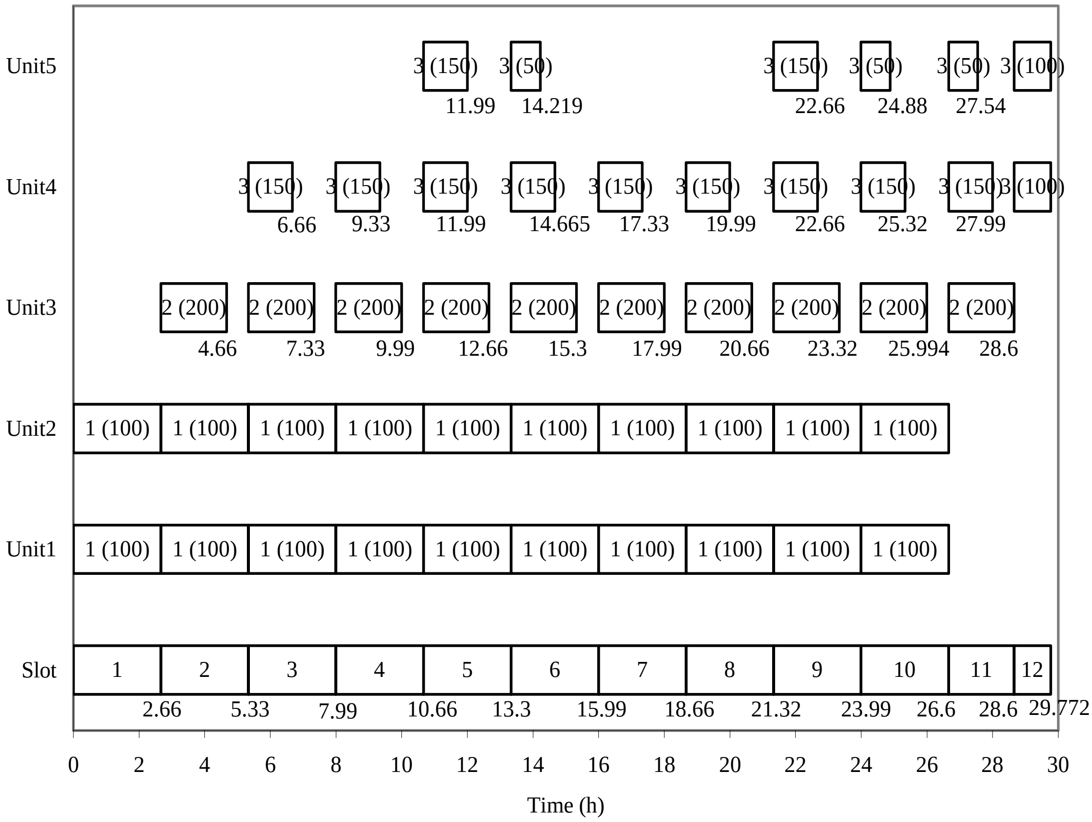


Figure 6.8: Minimum-makespan schedule for Example 1 with *K* = 12. The numbers within the parentheses denote the batch sizes (mu) of tasks.

To evaluate our model for minimizing the makespan, we compared it with those of Maravelias and Grossmann (2003a) and Giannelos and Georgiadis (2002) using the same three examples discussed earlier. All example data are same, except that fixed demands are now imposed, and *H* is no longer a parameter for our model. We solved each example for two demand scenarios. Table 6.4 summarizes the results of all the scenarios for all three models, while Figures 6.8-6.10 show the minimum-makespan schedules of our model for the first scenario of each example. The statistics for the second scenario of each example shown in Table 6.4 belong to the first feasible solution.

Table 6.4: Model and solution statistics for the minimum-makespan examples with variable batch processing times.

Model	K/N	H	CPU Time (s)	Nodes	RMILP (h)	MILP (h)	Binary Variables	Continuous Variables	Constraints	Nonzeros
Example 1a (D_4 = 2000 mu)										
Our	12	-	1.00	217	27.13	29.77	120	616	627	2078
M&G	13	50	4.94	1237	27.13	29.77	130	574	1569	6641
G&G	12	50	0.08	0	27.13	29.77	60	234	409	1202
Example 1b (D_4 = 4000 mu, First feasible K/N)										
Our	22	-	94.16	22648	51.36	56.43	220	1116	1167	3868
M&G	23	100	16034.27	2000283	51.36	56.43	230	1014	2759	15786
G&G	22	100	0.17	0	51.36	56.43	110	424	739	2202
Example 2a ($D_8 = D_9$ = 200 mu)										
Our	8	-	14.20	5593	18.69	19.79	96	555	535	1945
M&G	9	50	23.08	5204	18.69	19.79	144	758	1769	6735
G&G	8	50	1.97	2982	12.56	19.79	64	275	536	1821
Example 2b (D_8 = 500 mu, D_9 = 400 mu, First feasible K/N)[1]										
Our	22	-	136.13	4060	48.78	50.13	264	1479	1501	5473
M&G	23	100	798.90	6860	48.78	50.25	368	1934	4471	26279
G&G	22	100	5000.00	2555997	26.38	49.72	176	737	1432	5013
Example 3a (D_{12} = 100 mu, D_{13} = 200 mu)										
Our	7	-	1.09	439	12.40	14.37	119	688	871	2609
M&G	8	50	4.02	1203	12.40	14.37	176	922	2172	8053
G&G	7	50	1.47	2095	11.07	14.70	77	337	656	2121
Example 3b ($D_{12} = D_{13}$ = 250 mu, First feasible K/N)[2]										
Our	10	-	3042.17	337408	15.21	17.71	170	967	1264	3782
M&G	11	100	5000.00	316800	15.21	17.79	242	1267	2970	12427
G&G	10	100	27.19	30428	12.87	19.84	110	475	923	3036

M&G = Maravelias & Grossmann (2003a), G&G = Giannelos & Georgiadis (2002)

[1]Relative gaps: Our = 2.69%, M&G = 2.92%, G&G = 24.16%

[2]Relative gaps: Our = 2.99%, M&G = 3.75%, G&G = 2.99%

6.1.2.1 Example 1. In the motivating example, we first assume D_4 = 2000 mu. For this trivial case, all three models perform equally well (see Table 6.4). However, when we increase the demand to 4000 mu, the M&G model performs very poorly (16034 s), whereas our model requires only 94.2 s for an optimal solution. The G&G model again seems to be the fastest, but cannot guarantee an optimal solution, as we will again see in Example 3.

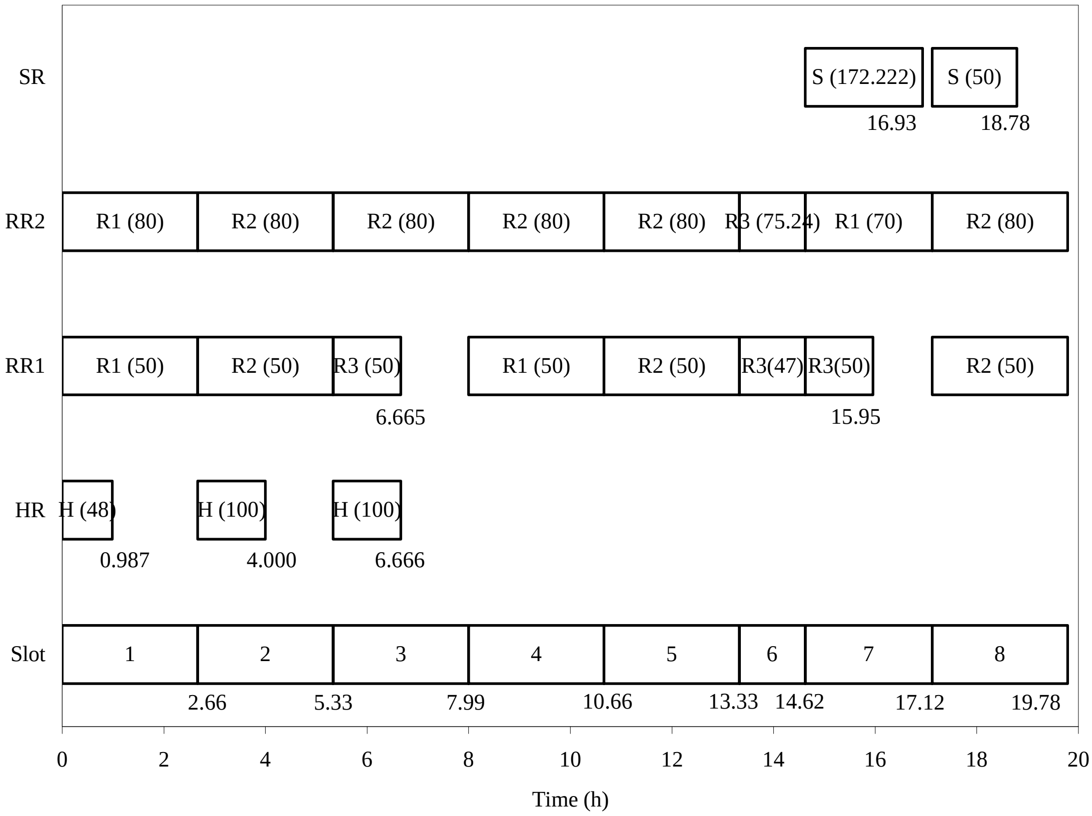


Figure 6.9: Minimum-makespan schedule for Example 2 with K = 8. The numbers within the parentheses denote the batch sizes (mu) of tasks.

6.1.2.2 Example 2. In the first scenario, we set $D_8 = D_9 = 200$ mu. Again, the M&G model is inferior to the other two in solution time (23.1 s vs. 14.2 s & 1.97 s). Note that the G&G model gives a poor RMILP objective for this case. In the second scenario, we take $D_8 = 500$ mu and $D_9 = 400$ mu, as Maravelias and Grossmann (2003b) did. However, they solved this scenario for constant processing times. Instead, we use variable processing times to evaluate our model for this more difficult scenario. We set the termination criteria as 3% gap and 5000 s for all three models uniformly. Interestingly, our model performs much faster (136 s for 2.69% gap) than the other two models. The M&G model shows a gap of 2.92% after 799 s, while the G&G model performs very poorly (5000 s for a gap of 24.16%). For the latter, even RMILP objective is inferior to those of the other two models. Again, our model outperforms the other two models convincingly.

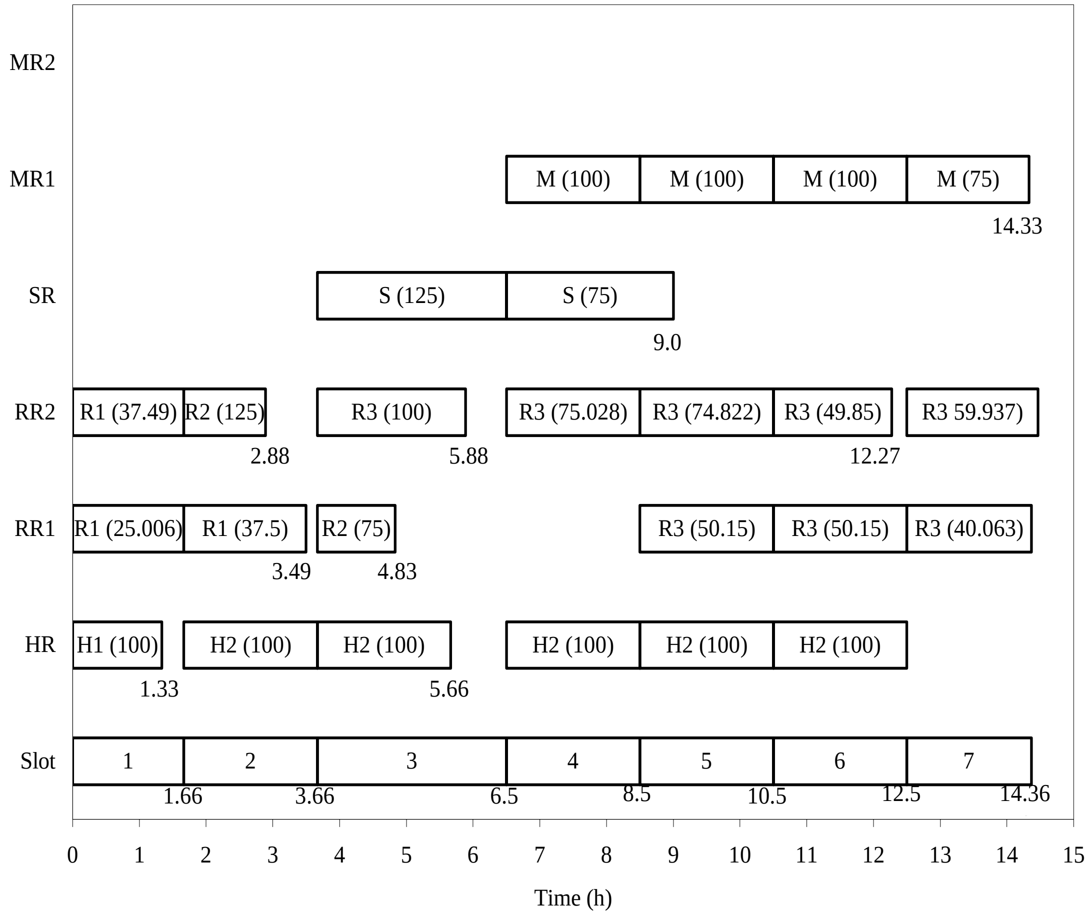


Figure 6.10: Minimum-makespan schedule for Example 3 with K = 7. The numbers within the parentheses denote the batch sizes (mu) of tasks.

6.1.2.3 Example 3. For the first scenario of D_{12} = 100 mu and D_{13} = 200 mu, the solution statistics of all three models are similar. However, the G&G model fails to give an optimal solution. For the second scenario of D_{12} = D_{13} = 250 mu, we set the same termination criteria as in the second scenario of Example 2. The same story repeats even for this scenario. This time, the G&G model even gives an inferior RMILP objective. Our model is again faster than the M&G model. Our model shows a gap of 2.99% after 3042 s, while the M&G model requires 5000 s for a gap of 3.75%.

The G&G model requires only 27.2 s for a gap of 2.99%, but the objective is again inferior.

From the above examples, it is clear that our model solves the makespan problems with variable processing times equally efficiently and optimally. Its performance does not seem to deteriorate in comparison to the profit objective. Unlike any other existing formulation, our model solves moderate-size problems quite effectively. Moreover, our model has no big-M constraints, so is not susceptible to the influences of parameter M on solution times. The other two models use big-M constraints and require some numerical value of the same. Hence, we conclude that our model is superior to the other two models. We now consider the case of constant processing times, as this has received attention in the literature.

6.1.3 Constant Batch Processing Times

The assumption of constant processing times should make the problems easier to solve. However, we do consider larger horizons to keep raise the difficulty. Table 6.1 gives the constant processing times (τ_{ij}) for all three examples. Table 6.5 summarizes the model and solution statistics, and Figures 6.11-6.13 show the maximum-profit schedules from our model for the first scenario of each example. For the first scenario (H = 12) of Example 1, all three models perform equally well as it is a trivial problem. We doubled the horizon for the second scenario (H = 24). Our model is an order of magnitude faster than the M&G model (440 s vs. 7572 s) whereas, the G&G results in an inferior objective though being attractive in terms of the statistics. For both scenarios (H = 12 and H = 16) of Example 2, our model is faster than the M&G model (0.38 s vs. 1.36 s and 81.3 s vs. 837 s) and the G&G model gives poor RMILP and MIP objectives. Finally, we solved Example 3 for H = 12. For this scenario, all models perform equally well. However, the G&G model fails to give an optimal solution.

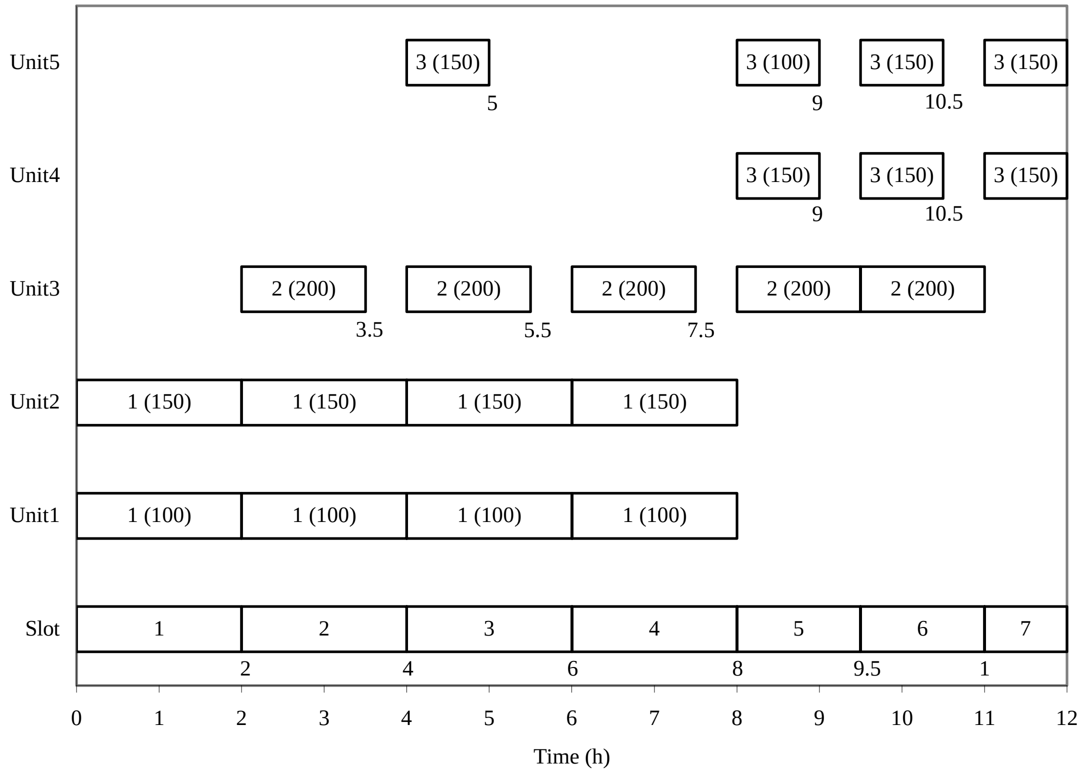


Figure 6.11: Maximum-profit schedule for Example 1 with H = 12 h, K = 7, and constant batch processing times. The numbers within the parentheses denote the batch sizes (mu) of tasks.

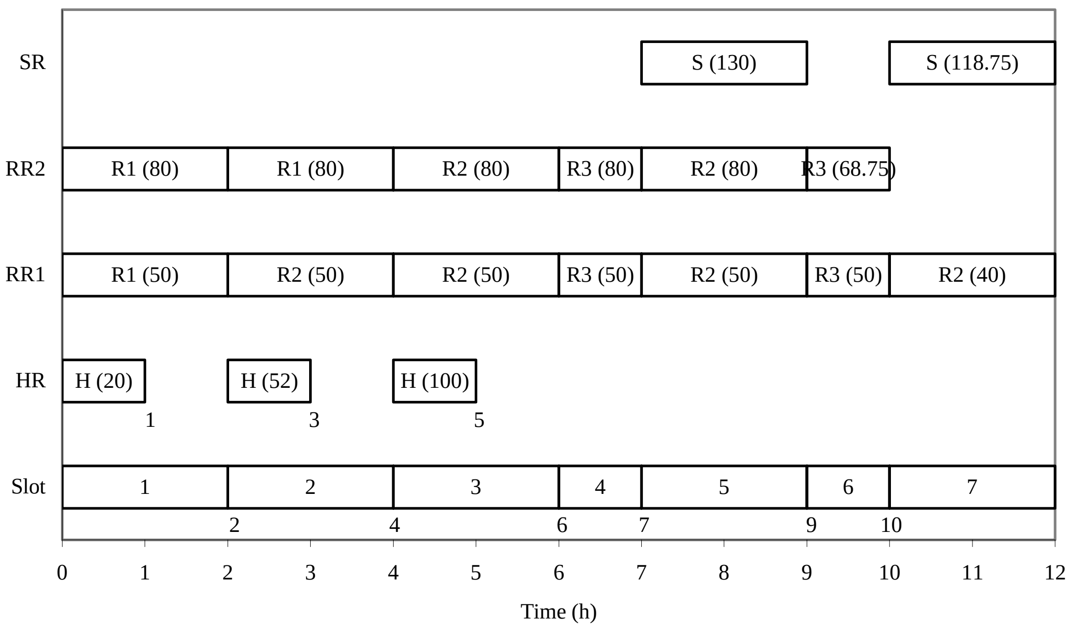


Figure 6.12: Maximum-profit schedule for Example 2 with H = 12 h, K = 7, and constant batch processing times. The numbers within the parentheses denote the batch sizes (mu) of tasks.

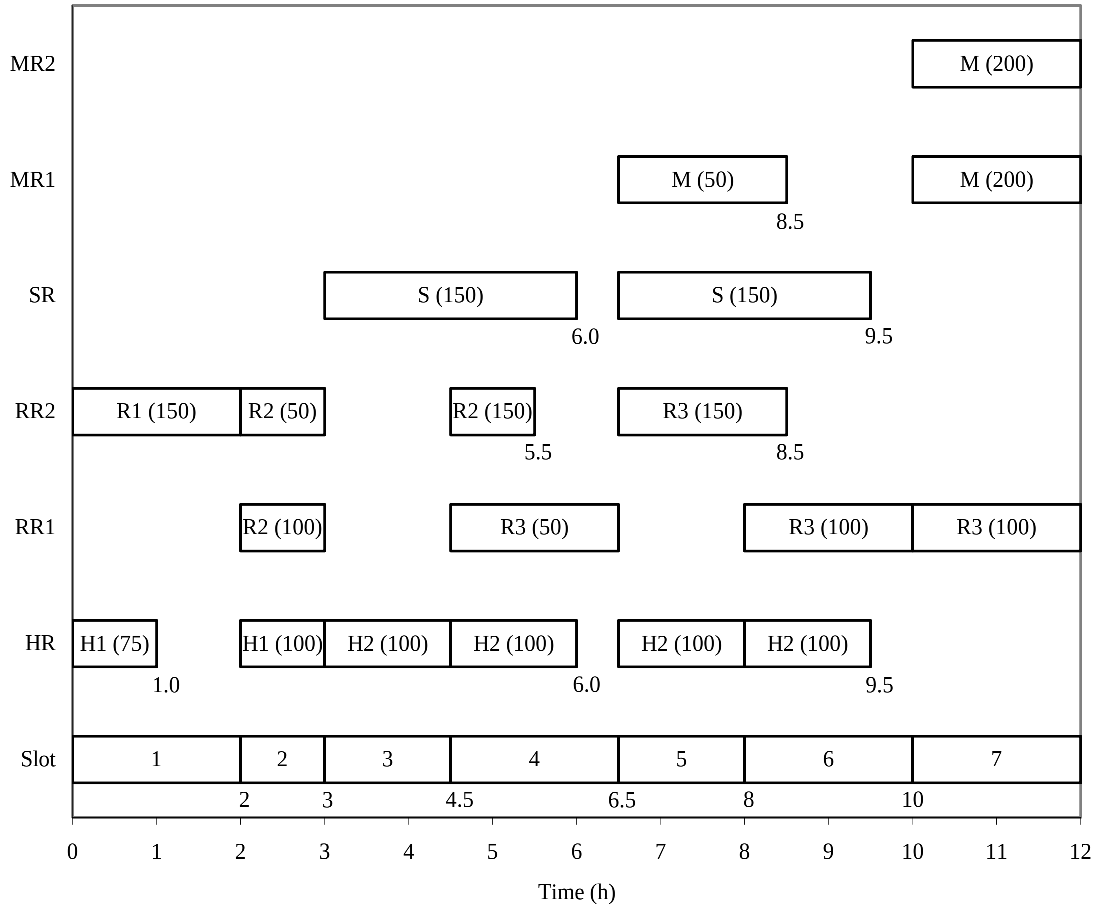


Figure 6.13: Maximum-profit schedule for Example 3 with H = 12 h, K = 7, and constant batch processing times. The numbers within the parentheses denote the batch sizes (mu) of tasks.

In summary, we conclude the following. Irrespective of the scheduling objective (makespan or profit), our model is faster than the other two models in the most difficult scenarios. In some scenarios, it provides tighter RMILP objectives. As compared to the other two models, the M&G model requires more constraints, variables, and nonzeroes. However, although computationally inferior, it is foolproof in that it does not give suboptimal solutions at all. The G&G model seems superior in model statistics and solution speed mainly because it uses explicit sequencing constraints as done by Ierapetritou & Floudas (1998), but it has some fundamental flaw that results in suboptimal solutions for some scenarios. In an attempt to ensure mass balance, Giannelos and Georgiadis (2002) force the end or start times of tasks producing or consuming the same material to be equal. This can potentially restrict the freedom of task occurrences. The other two models (our & M&G) do not use such explicit sequencing constraints.

Table 6.5: Model and solution statistics for the maximum-profit examples with constant batch processing times.

Model	*K* / *N*	CPU Time (s)	Nodes	RMILP ($)	MILP ($)	Binary Variables	Continuous	Constraints	Nonzeros
Example 1a (H = 12)									
Our	7	0.03	0	5000.0	5000.0	70	366	354	1115
M&G	8	0.05	0	5000.0	5000.0	80	353	970	3090
G&G	7	0.08	24	5000.0	5000.0	35	138	240	612
Example 1b (H = 24)									
Our	15	440.41	145630	13000.0	12000.0	150	766	786	2467
M&G	16	7572.22	1373260	13000.0	12000.0	160	705	1922	8122
G&G	13	0.08	0	11000.0	11000.0	65	252	438	1140
Example 2a (H = 12)									
Our	7	0.38	44	3799.4	3638.8	84	489	458	1582
M&G	8	1.36	252	3799.4	3638.8	128	673	1567	5313
G&G	7	0.25	370	3813.2	3638.8	56	241	463	1381
Example 2b (H = 16)									
Our	10	81.27	16553	5586.7	5162.1	120	687	665	2290
M&G	11	837.42	118764	5586.7	5162.1	176	925	2146	8112
G&G	9	3.50	6532	5054.9	4937.1	72	307	591	1779
Example 3 (H = 12)									
Our	7	5.25	2940	3465.6	3050.0	119	688	859	2455
M&G	8	9.53	3120	3465.6	3050.0	176	921	2159	7425
G&G	6	0.27	62	2871.9	2675.0	66	290	554	1583

M&G = Maravelias & Grossmann (2003a), G&G = Giannelos & Georgiadis (2002)

Having evaluated our model rigorously, we now discuss miscellaneous aspects of our model.

6.2 DISCUSSION

Here, we discuss about the unique features of our model, the impact of alternate constraints on the solution times, fixing the number of slots in slot-based or event-based formulations, and the basic criteria required for any comparison works.

Table 6.6: Remaining processing times of batches on units at various times in the examples.

Unit j	Remaining batch processing time t_{jk} on j at T_k (h) T_1	T_2	T_3	T_4	T_5	T_6
Variable batch processing times						
Example 1b (H = 12 & K = 8)						
Unit 1	-	1.999(2.666)	-	0.631(1.964)	-	-
Unit 2	0.666(3.332)	-	1.333(3.332)	-	1.987(2.618)	-
Unit 3	-	-	-	0.631(1.964)	-	-
Example 2b (H = 10 & K = 7)						
RR1	-	-	-	0.613(1.333)	-	1.856(2.146)
RR2	-	-	0.720(1.964)	-	0.290(0.903)	-
SR	-	-	-	0.903(1.623)	0.290(1.623)	-
Example 3a (H = 8 & K = 6)						
HR	-	0.905(1.812)	-	-	-	-
RR2	0.907(1.730)	-	-	-	-	-
SR	-	-	-	0.830(2.596)	-	-
Makespan minimization						
Example 3a (D_{12} = 100 mu, D_{13} = 200 mu & K = 7)						
SR	-	-	-	2.000(3.000)	-	-
Constant batch processing times						
Example 3 (H = 12 & K = 7)						
RR2	-	-	-	-	0.500(2.000)	-
SR	-	-	1.500(3.000)	-	1.500(3.000)	-
MR1	-	-	-	-	0.500(2.000)	-

The numbers in parentheses are the total batch processing times

6.2.1 Remaining Batch Processing Times

In contrast to all the previous models, our model uses the novel idea of balances. For instance, it does a balance on the remaining batch processing times and materials in each unit. Table 6.6 lists the values of t_{jk} (the time remaining in completing the current batch on unit j at T_k) for various units and time points in the three examples. It shows only the units that continue their production at different T_k. The numbers in the parentheses denote the times remaining to complete the batches. Consider Unit 1 in

Example 1b. It starts a batch of Task 1 at T_1 (see Figure 6.2) with a batch size of 100 mu and continues its production at T_2. It needs 2.665 h to complete this batch. However, the start of Task 1 on Unit 2 and that of Task 2 on Unit 3 triggers a new slot 3 at T_2. Hence, the remaining batch processing time on Unit 1 at T_2 is 2.666 + 2.665 – 3.332 = 1.999 h, where 2.666 h is the start time of the batch on Unit 1. Most t_{jk} values in Table 6.6 are the exact remaining batch processing times on units. However, note that t_{j1} for RR2 in Example 3a (see Figure 6.7) is the sum of the remaining batch processing time and the unit idle time. The exact remaining batch processing time for this case is 0 + 1.73 – 0.965 = 0.765 h, but the reported value is 0.765 + 0.142 = 0.907 h, where 0.142 h is the unit idle time on RR2 after that batch. In the optimal schedule of Figure 6.10 also, unit S (separator) shows a remaining time t_{jk} of 2 h at T_4 instead of 0.5 h. This is because eq. 5.9 being an inequality constraint allows such slack for t_{jk} without affecting the solution. Alternatively, it is possible to avoid this slack by making eq. 5.9 an equality as follows.

$$t_{j(k+1)} = t_{jk} + \sum_{i \in \boldsymbol{I}_j} (\alpha_{ij} Y_{ijk} + \beta_{ij} B_{ijk}) - SL_{(k+1)} \qquad k < K \qquad (6.1)$$

However, this would force another slot to start at the end of each batch and the null task (i = 0) to take up any idle time. For eq. 6.1 to work, we must select α_{0j} and β_{0j} values carefully. In addition, one must properly set B_{0j}^U, because SL_k is common to all units and B_{0j}^U must exceed the maximum possible batch size of any task on a unit. Hence,

$$B_{0j}^U = \max_{i \in I_j, j} \left(B_{ij}^U \right)$$

However, we do not see any need for using eq. 6.1. In fact, we prefer eq. 5.9 instead of eq. 6.1, as the former should require fewer slots.

6.2.2 Alternate Constraints

In proposed formulation (see previous chapter), eqs. 5.6 and 5.7 serve the same purpose. We may impose eq. 5.6 or 5.7 individually or both at the same time, and these options may affect the solution time. We observed that eq. 5.6 alone performs somewhat better in some scenarios, eq. 5.7 alone better in some scenarios, and both in some other scenarios. Consider Example 1b with H = 12 and K = 8 in Table 6.3. The solution time and nodes in Table 6.3 are for using eq. 5.6 alone. If we use eq. 5.7 alone, then we get the solution in 26.75 s with 28431 nodes. If we use both eqs. 5.6 and 5.7, we need 23153 nodes to find the optimal solution in 21.80 s. In most cases, however, eqs. 5.6 and 5.7 perform equally well, when used individually. We used eq. 5.6 for all examples.

6.2.3 Fixing the Number of Slots

Previous work (Ierapetritou and Floudas, 1998) argued that the event-based models avoid the need for pre-fixing the numbers of slots as done by the slot-based models. For example, Ierapetritou and Floudas (1998) state: “The proposed formulation is based on a continuous-time representation that avoids the prepostulation of unnecessary time slots or intervals. It only requires the initial consideration of a necessary number of event points corresponding to either the initiation of a task or the beginning of unit utilization”. The difference between *prepostulation* and the *initial consideration of a necessary number of event points* is not obvious. The a priori selection of the number of time slots does not appear to be any less case-dependent than the initial consideration of a necessary number of event points. Indeed, there is no single, foolproof, general formula for prefixing the number event points in the event-based formulations. The common approach has been to increase the number of event points one at a time, until the objective does not change any further. Even this approach is not foolproof, as Castro et al. (2001) have shown that the objective may not change with an increment of one additional event point, but may change with an increment of two or more. It would seem therefore that in the event-based formulations, one can only estimate and conduct tests with varying numbers of *N*, the number of event points. It is also not clear whether it is better to prepostulate a generous value of *N* and solve the problem once, or solve the same problem repeatedly by varying N. Should not the computation time for the latter be the sum of the times for all repeated runs? Whatever the strategy used, it seems that the same strategy can also be used for fixing *K* (the number of slots) in the slot-based models. Most slot-based models (e.g. Karimi and McDonald, 1997; Lim and Karimi, 2003b) have preferred to pre-postulate empirically a generous number of slots in order to avoid solving the problem repeatedly. Although not foolproof, heuristic formulas for fixing a priori the number of slots in slot-based formulations do exist and work reasonably well in many instances (Lim and Karimi, 2003a). However, as we have done in this paper, slot-based models can also use the same strategy of optimizing the number of slots by solving the problem repeatedly.

In the case of profit maximization, we increase *K* gradually by one, until the solution does not change, even if we were to increase *K* by two or more. In the case of makespan problems, we increase *K*, until we first get a feasible solution. Then, we increase it further to get the optimal solution as in the profit problems.

6.2.4 Effects of Computing Hardware and Software

As discussed by Karimi et al. (2004), hardware and software can significantly affect the performance of different MILP formulations. Although they discuss this issue in more detail, we list here some minimum prerequisites to a sound comparison.

(i) Comparative model solution times must be based on the same version of the same solver/software. It is obvious that MIP solvers and software versions change rapidly, and it is not right to compare models based on different versions of a solver/software. For example, CPLEX and OSL will differ in solution times on the same problem. Moreover, different versions of the same solver may result in different solution statistics. Conclusions based on such times can never be reliable.

(ii) Hardware, operating systems, and compilers have tremendous effects on the solution times of MILPs. In fact, as noted by Karimi et al. (2004), it is preferable to use multiple computing platforms to get a more robust evaluation of competing models. It is obvious that Random Access Memory (RAM) can play a vital role in computing speed. Moreover, the performance on a high-end workstation is normally better than that on a PC with the same CPU speed. It is even possible that machines of similar specifications but from different companies have different computing speeds. Clearly, attention to hardware and software details is must in comparing MILP models.

(iii) Value of M in the Big-M constraints also has a notorious effect on MILP solution times as pointed by Gupta and Karimi (2003) and Lim and Karimi (2003a). Event-based models use big-M constraints profusely. This is another factor conveniently ignored by the existing literature. For models involving the big-M constraints, one must average model performances over a range of M-values.

In our opinion, comparisons made without due attention to any of the above basic criteria cannot be reliable. To reinforce the above discussion and to highlight the effects of software and hardware on solution times, we solved some scenarios of our examples using different hardware and software. Table 6.7 summarizes the results of our, M&G and G&G models on different computers and with different versions of CPLEX.

Table 6.7: Performance of models on different machines and with different CPLEX versions.

Model	$K/N\backslash H$	CPU time (s) \ Nodes[a]			
		Computer 1		CPLEX 8.1.0	
		CPLEX 7.5.0	CPLEX 8.1.0	Computer 2	Computer 3
Variable batch processing times					
Example 1b (H = 12)					
Our	8	24.3 \ 21740	23.5 \ 22850	21.86	160.71
M&G	9	41.7 \ 34557	34.0 \ 28469	31.97	225.43
G&G	6	0.09 \ 24	0.08 \ 19	0.20	0.26
Example 3b (H = 8)					
Our	6	205 \ 105130	501 \ 318290	461.66	3274.28
M&G	7	479 \ 169639	588 \ 237565	538.94	3784.09
G&G	5	0.30 \ 593	0.38 \ 600	0.41	1.95
Makespan minimization					
Example 2a ($D_8 = D_9 =$ 200 mu)					
Our	8	24.22 \ 9780	14.2 \ 5593	13.31	97.28
M&G	9 \ 50	105 \ 21822	23.1 \ 5204	21.44	152.14
G&G	8 \ 50	2.08 \ 3285	1.97 \ 2982	1.83	12.46
Example 3a ($D_{12} =$ 100 mu, $D_{13} =$ 200 mu)					
Our	7	0.61 \ 219	1.09 \ 439	1.09	7.20
M&G	8 \ 50	2.14 \ 650	4.02 \ 1203	3.89	27.56
G&G	7 \ 50	0.31 \ 238	1.47 \ 2095	1.42	8.69
Constant batch processing times					
Example 1a (H = 12)					
Our	7	0.08 \ 23	0.03 \ 0	0.09	0.21
M&G	8	0.04 \ 120	0.05 \ 0	0.19	0.35
G&G	7	0.08 \ 36	0.08 \ 24	0.20	0.22
Example 2b (H = 16)					
Our	10	149 \ 29902	81.3 \ 16553	74.92	560.17
M&G	11	1743 \ 263982	837 \ 118764	749.34	5532.46
G&G	9	3.27 \ 6667	3.50 \ 6532	3.02	22.50

M&G = Maravelias & Grossmann (2003a), G&G = Giannelos & Georgiadis (2002)

Computer 1 = DELL GX 270 (Pentium IV 2.8 GHz CPU with 1GB of RAM)

Computer 2 = DELL PWS650 workstation (3.06 GHz CPU with 3.67GB of RAM)

Computer 3 = COMPAQ PC (Pentium III 448 MHz CPU with 256MB of RAM)

[a]The number of nodes in Computer 2 and Computer 3 is same as that in Computer 1 using CPLEX 8.1.0

We considered the two objectives (maximum profit and minimum makespan) with variable processing times and two scenarios for maximum profit using constant processing times. We used three different computers (Computer 1 = the computer used in the last section; Computer 2 = DELL PWS650 workstation 3.06 GHz CPU with 3.67 GB of RAM; Computer 3 = COMPAQ PC Pentium III 448 MHz CPU with 256 MB of RAM) and two different versions of CPLEX (CPLEX 8.1.0 and CPLEX 7.5.0). Tables 6.3-6.5 also give the results for Computer 1 and CPLEX 8.1.0. We report only the solution statistics, as model statistics are the same as in those tables. In addition, the numbers of nodes do not change with the computers for the same version and operating system (e.g. Unix vs. Windows). Hence, we report the number of nodes only for the version comparison.

In comparing the versions using Computer 1, we observe that CPLEX 7.5.0 gives better solution statistics than CPLEX 8.1.0 for Example 3b (see Table 6.7) under profit maximization using variable batch processing times, Example 3a under makespan minimization, and Example 2b under profit maximization using constant batch processing times. For the remaining scenarios, CPLEX 8.1.0 performs better than CPLEX 7.5.0. Hence, it is clear from these results that the latest version need not always be faster.

In comparing the hardware using CPLEX 8.1.0, we observe that Computer 1 and Computer 2 perform almost the same. But, there is a significant difference in the solution time when we use Computer 3. Though it is obvious that workstation can easily outperform a PC, the difference in speed is appreciable. This clearly suggests that sufficient attention must be paid to the hardware used in comparing different models.

In light of the above, consider the numerical comparison of Ierapetritou and Floudas (1999) with the slot-based model of Karimi and McDonald (1997). The main point against their comparison and its subsequent conclusions is that they were based on the verbatim solution times reported by Karimi & McDonald (1997), and not on their own implementation of the model of Karimi & McDonald (1997). They compared the two models based on computations that used different hardware and software. While Karimi & McDonald (1997) used GAMS 2.25.087, Cplex 4.0, AIX operating system, and RS/6000P IBM workstation, Ierapetritou and Floudas (1999) used GAMS 2.25.??, Cplex 4.0.8, ?? operating system, and HP-C160 workstation. Firstly, it is quite likely that the two CPLEX versions differed in performance. Secondly, the difference due to the hardware, compilers, and operating systems can be substantial as we demonstrated above. Karimi et al. (2004) have shown that even the same version of CPLEX can perform quite differently on two different machines, compilers, or operating systems. In light of these basic flaws in their comparison, it is unclear how the differences in model solution times (0.72 s vs. 5.0 s, 0.31 s vs. 2.0 s, 9.92 s vs. 15 s, and so on) used by Ierapetritou and Floudas (1999) can form the basis for fair and unambiguous conclusions. As seen earlier, such differences can easily arise, even when one solves the same model on different computers. Besides this aspect of their comparison, two aspects or results of their comparison further make their conclusions highly questionable.

One aspect relates to the number of event points and also our discussion on fixing the number of slots. In their model, they adjusted and optimized the necessary number of event points. In contrast, the results reported by Karimi & McDonald (1997) assumed a fixed, generous number of slots without any optimization or adjustment. In other words, the results of Karimi & McDonald (1997) had some redundant slots, which would not exist, if they had used the necessary number of slots. It is obvious that the number of slots affects the solution time significantly and a comparison based on disproportionate numbers of event points and slots would not lead to reliable conclusions. A fair comparison would be to use the optimized numbers for both event points and slots.

The other aspect is the RMILP objective values of their model in comparison to those of Karimi and McDonald (1997). Their RMILP objective values were invariably and noticeably inferior (e.g. 3652 vs. 4292, 1287 vs. 3044, 1777 vs. 14233, and so on) for a minimization problem. Although, this does not necessarily imply that their model would be slower, it proves that the slot-based formulation of Karimi and McDonald (1997) is definitely tighter and could very well be a better formulation, if compared on an apple-to-apple basis.

6.3 CONCLUSION

The existing comparisons between the event-based and slot-based models lack thorough and rigorous analysis, thus the question of which is better still demands a convincing answer. We assessed the performance of our slot-based scheduling model in comparison with two other models (slot-based M&G and event-based G&G models) in the literature using three examples. From the assessment of three models, we conclude that our slot-based model can comfortably outperform the event-based G&G model and slot-based M&G model. In addition, we conclude that decoupling of tasks from units in a scheduling formulation cannot reduce the number of binary assignment variables. The novel continuous-time formulation presented in the last chapter uses synchronous slots and does not decouple tasks from units (i.e. uses 3-index binary assignment variables), but it still has fewer binary variables, constraints, and nonzeros, and at the same time it is simpler, more efficient, and potentially tighter than the best models (event-based or otherwise) in the literature on short-term scheduling in multipurpose batch plants. In contrast to the existing models, it is equally efficient for both profit maximization and makespan minimization even with variable batch processing times, and has no big-M constraints. We believe that the latter may be a major contributor to our model's better efficiency. Lastly, this work presents a novel idea of balances (time, mass, resource, etc.) in developing scheduling formulations, which can enable one to handle general resource-constrained scheduling problems using the proposed formulation.

Chapter 7

CONCLUSIONS AND RECOMMENDATIONS

7.1 Conclusions

We addressed two common but important problems of planning and scheduling in pharmaceutical supply chains. Firstly, we addressed a supply chain planning problem to assess the feasibility or profitability of introducing new active ingredients or intermediates in a given pharmaceutical plant. We developed a single-plant-centric, multi-period, MILP model that allows complex production recipes with multiple intermediates, outsourcing of existing intermediates, material movement among different production/supply/demand facilities, validation times for new tasks, minimum campaign lengths, line-dependent cleaning, and so on, and considers explicitly the details of campaign sequencing and timing on individual production lines in a pharmaceutical plant. The planning model is able to give reasonably quick solutions for three examples involving twelve materials, four production lines, and up to three months of horizon. It can assist the plant management in making quick, optimal assessment of outsourcing and new product introductions in a pharmaceutical plant. Although we limited ourselves to only the primary production in the pharmaceutical supply chain, one can readily apply the proposed model to secondary production as well.

Secondly, we addressed the scheduling problem in pharmaceutical supply chains. Here, we proved that decoupling of tasks from units in a scheduling formulation cannot reduce the number of binary assignment variables. In the literature, comparisons between the event-based and slot-based models lack thorough and rigorous analysis, thus the question of which is better still demands a convincing answer. We presented a novel continuous-time formulation that uses synchronous slots and does not decouple tasks from units (i.e. uses 3-index binary assignment variables), but it still has fewer binary variables, constraints, and nonzeros, and at the same time it is simpler, more efficient, and potentially tighter than the best models (event-based or otherwise) in the literature on short-term scheduling in multipurpose batch plants. Moreover, we proposed a novel idea of balances (time, mass, resource, etc.) in developing scheduling formulations, which can enable one to handle general resource-constrained scheduling problems using the proposed formulation. In contrast to the existing models, it is equally efficient for both profit maximization and makespan minimization even with variable batch processing times, and has no big-M constraints. We believe that the latter may be a major contributor to better efficiency of our model. Finally, our model is much simpler (almost one third reduction in binaries, two third reduction in constraints as well as nonzeros) and faster (almost an order of magnitude in most cases studied) than the most recent and the best model (Maravelias and Grossmann, 2003a) existing in the literature.

7.2 Recommendations for Future Work

The proposed planning model considers scheduling issues to make the plan realistic. However, we employed asynchronous slots to time campaigns in the production plan. From an assessment of the scheduling model, we anticipate that the use of synchronized slots accompanied with the novel balances could handle problems of large dimension. Hence, one can reformulate the proposed planning model using synchronized slots and compare its performance with the proposed model.

As mentioned in Chapter 1, the productions in primary and secondary sites are characterized by their respective end demands. Primary production looks at the demands from only the secondary production whereas the secondary production caters the needs of the end customers. Hence, the latter is more flexible and responsive to the varying demands of the end products. Moreover, secondary production acts as an opaque layer between the primary production and the end customers. As a result, the primary production becomes less responsive to the changes in supply chain. The overall performance of a supply chain is dependent on both primary and secondary productions. Hence, future work can attempt to integrate these two production sites and present a more general planning and scheduling model.

A major challenge facing the pharmaceutical industry is inventory management. To avoid the risk of running out of stock, pharmaceutical companies have historically created security cushions in the form of high inventory levels. Hence, a lot of working capital is tied up in the inventory of products. If one could devise an optimal inventory plan considering as much of the risk/uncertainty factors, then it will be useful for these companies.

Another challenge facing the pharmaceutical industry is the process of selecting which new products to develop. R&D can deliver many new candidates. However, not all can be developed as each involves a lot of money and time. Hence, the management can invest on only selected potential candidates for further development processes. Moreover, clinical trials or validation processes for the new products should co-ordinate with the production management. This clearly shows the need of models that can support a holistic approach to product portfolio management in the pharmaceutical industry. In addition, these models should also integrate production management, capacity management and trading structure. Hence, one can attempt to streamline the entire supply chain by integrating various corporate activities like R&D, production planning, new product introductions, outsourcing, validation, facility expansion, logistics etc.

REFERENCES

1. Balla, G. Scheduling in single stage with parallel processors. M. Eng. Thesis in progress, National University of Singapore, 2004.
2. Beard, C. and Easingwood, C. New product launch: Marketing action and launch tactics for high-technology products. Industrial Marketing Management, *25*, pp.87-103. 1996.
3. Bennett, B. and Cole, G. Pharmaceutical production: An Engineering guide. pp.75-201, Institution of Chemical Engineers, UK, 2003.
4. Blau, G., Mehta, B., Bose, S., Pekny, J., Sinclair, G., Keunkar, K. and Bunch, P. Risk management in the development of new products in highly regulated industries. Comput. Chem. Eng., *24*, pp.659-664. 2000.
5. Bose, S. and Blau, G.E. Use of a network model interface to build spreadsheet models of process systems: a productivity enhancement tool for risk management studies. Comput. Chem. Eng., *24*, pp.1511-1515. 2000.
6. Bowersox, D.J., Stank, T.P. and Daugherty, P.J. Lean launch: Managing product introduction risk through response-based logistics. Journal of Product Innovation Management, *16*, pp.557-568. 1999.
7. Brooke, A., Kendrick, D., Meeraus, A. and Raman, R. GAMS: A user’s guide. Washington, DC: GAMS Development Corporation , 1998.
8. Castro, P., Barbosa-Povoa, A.P.F.D. and Matos, H. An improved RTN continuous-time formulation for the short-term scheduling of multipurpose batch plants. Ind. Eng. Chem. Res., *40*, pp.2059-2068. 2001.

9. Gjendrum, J., Shah, N. and Papageorgiou, L.G. New product introduction supply chain planning. Contemporary Research Issues in New Product Introduction. Acta Wasaensia, *77*, pp.25-49. 2001.

10. Giannelos, N.F. and Georgiadis, M.C. A simple new continuous-time formulation for short-term scheduling of multipurpose batch processes. Ind. Eng. Chem. Res., *41*, pp.2178-2184. 2002.

11. Gupta, S. and Karimi, I.A. An improved MILP formulation for scheduling multiproduct, multistage batch plants. Ind. Eng. Chem. Res., *42*, pp.2365-2380. 2003.

12. Gupta, A. and Maranas, C.D. A hierarchical Lagrangean relaxation procedure for solving midterm planning problems. Ind. Eng. Chem. Res., *38*, pp.1937-1947. 1999.

13. Gupta, A. and Maranas, C.D. A two-stage modeling and solution framework for multisite midterm planning under demand uncertainty. Ind. Eng. Chem. Res., *39*, pp.3799-3813. 2000.

14. Ierapetritou, M.G. and Floudas, C.A. Effective continuous time formulation for short-term scheduling. 1. Multipurpose batch processes. Ind. Eng. Chem. Res., *37*, pp.4341-4359. 1998.

15. Ierapetritou, M.G. and Floudas, C.A. Effective continuous time formulation for short-term scheduling. 3. Multiple intermediate due dates. Ind. Eng. Chem. Res., *38*, pp.3446-3461. 1999.

16. Jackson, J.R. and Grossmann, I.E. Temporal decomposition scheme for nonlinear multisite production planning and distribution models. Ind. Eng. Chem. Res., *42*, pp.3045-3055. 2003.

17. Jain, V. and Grossmann, I.E. Resource-constrained scheduling of tests in new product development. Ind. Eng. Chem. Res., *38*, pp.3013-3026. 1999.
18. Karimi, I.A. and McDonald, C.M. Planning and scheduling of parallel semicontinuous processes. 2. Short-term scheduling. Ind. Eng. Chem. Res., *36*, pp.2701-2714. 1997.
19. Karimi, I.A., Tan, Z.Y.L. and Bhushan, S. An improved formulation for scheduling an automated wet-etch station. Comp. Chem. Eng., in press. 2004.
20. Kondili, E., Pantelides, C.C. and Sargent, R.W.H. A general algorithm for short-term scheduling of batch operations – I. MILP formulation. Comput. Chem. Eng., *17*, pp.211-227. 1993.
21. Lamba, N. and Karimi, I.A. Scheduling parallel production lines with resource constraints-1.Model formulation. Ind. Eng. Chem. Res., *41*, pp.779-789. 2002a.
22. Lamba, N. and Karimi, I.A. Scheduling parallel production lines with resource constraints- 2. Decomposition algorithm, Ind. Eng. Chem. Res., *41*, pp.790-800. 2002b.
23. Lim, M.F. and Karimi, I.A. A Slot-based formulation for single-stage multi-product batch plants with multiple orders per product. Ind. Eng. Chem. Res. *42*, pp.1914-1924. 2003a.
24. Lim, M.F. and Karimi, I.A. Resource-constrained scheduling of parallel production lines using asynchronous slots. Ind. Eng. Chem. Res., *42*, pp.6823-6831. 2003b.
25. Lee, K.H., Park, H.I. and Lee, I.B. A novel nonuniform discrete time formulation for short-term scheduling of batch and continuous processes. Ind. Eng. Chem. Res., *40*, pp.4902-4911. 2001.

26. Maravelias, C.T. and Grossmann, I.E. Simultaneous planning for new product development and batch manufacturing facilities. Ind. Eng. Chem. Res., *40*, pp.6147-6164. 2001.

27. Maravelias, C.T. and Grossmann, I.E. A new MILP variable resource constrained scheduling model for the testing of pharmaceuticals and agrochemicals. Proceedings Fourth International Conference on Foundations of Computer Aided Process Operations, 2003, Coral Springs, Florida, pp.253-256.

28. Maravelias, C.T. and Grossmann, I.E. New general continuous-time state-task network formulation for short-term scheduling of multipurpose batch plants. Ind. Eng. Chem. Res., *42*, pp.3056-3074. 2003a.

29. Maravelias, C.T. and Grossmann, I.E. Minimization of the makespan with a discrete-time state-task network formulation. Ind. Eng. Chem. Res., *42*, pp.6252-6257. 2003b.

30. McDonald, C.M. and Karimi, I.A. Planning and scheduling of parallel semicontinuous processes. 1. Production planning. Ind. Eng. Chem. Res., *36*, pp.2691-2700. 1997.

31. Mockus, L. and Reklaitis, G.V. Mathematical programming formulation for scheduling of batch operations based on non-uniform time discretization. Comput. Chem. Eng., *21*, pp.1147-1156. 1994.

32. Mockus, L., Vinson, J.M. and Houston, R.B. Planning and scheduling in a pharmaceutical research and development. *Computer* Aided Process Engineering, *8*, pp.1057-1062. 2000.

33. Mockus, L., Vinson, J.M. and Luo, K. The integration of production plan and operating schedule in a pharmaceutical pilot plant. Comput. Chem. Eng., *26*, pp.697-702. 2002.

34. Oh, H. and Karimi, I.A. Planning production on a single processor with sequence-dependent setups. Part 1: determination of campaigns. Comput. Chem. Eng., *25*, pp.1021-1030. 2001a.

35. Oh, H. and Karimi, I.A. Planning production on a single processor with sequence-dependent setups. Part 2: Campaign sequencing and scheduling. Comput. Chem. Eng., *25*, pp.1031-1043. 2001b.

36. Papageorgiou, L.G., Rotstein G.E. and Shah N. Strategic supply chain optimization for the pharmaceutical industries. Ind. Eng. Chem. Res., *40*, pp.275-286. 2001.

37. Sahinidis, N.V. and Grossmann, I.E. MINLP model for cyclic multiproduct scheduling on continuous parallel lines. Comput. Chem. Eng., *15*, pp.85-103. 1991.

38. Schmidt C.W. and Grossmann, I.E. Optimization models for the scheduling of testing tasks in new product development. Ind. Eng. Chem. Res., *35*, pp.3498-3510. 1996.

39. Shah, N., Pantelides, C.C. and Sargent, R.W.H. Optimal periodic scheduling of multipurpose batch plants. Comput. Chem. Eng., *17*, pp.229-244. 1993.

40. Shah, N. Pharmaceutical supply chains: key issues and strategies for optimization. Proceedings Fourth International Conference on Foundations of Computer Aided Process Operations, 2003, Coral Springs, Florida, pp.73-85.

41. Singhvi, A., Madhavan, K.P. and Shenoy, U.V. Pinch analysis for production planning in supply chains. Proceedings Fourth International Conference on Foundations of Computer Aided Process Operations, 2003, Coral Springs, Florida, pp.477-480.

42. Sivanandam, P.S. Scheduling of serial multiproduct batch processes. M. Eng. Thesis in progress, National University of Singapore. 2004.

43. Subramanian, D., Pekny, J.F. and Reklaitis, G.V. A simulation-optimization framework for addressing combinatorial and stochastic aspects of an R&D pipeline management problem. Comput. Chem. Eng., *24*, pp.1005-1011. 2000.

APPENDIX A: Files for Chapter 4

A.1 GAMS files for Examples 1-3

A.1.1 Example 1

A.1.1.1 Example 1a

```
VARIABLES
NC(i,l,t)        number of campaigns of i on l in t
PQ(m,l,t)        amount of m produced or consumed on l in t
DQ(i,l,t)        Differential production amount of i on l in t
CL(i,k,l,t)      campaign length of i in k of l in t
PT(i,l,t)        effective production time of i on l in t
SL(k,l,t)        length of k of l in t
TS(k,l,t)        starting time of k of l in t
TE(k,l,t)        ending time of k of l in t
TTS(i,k,l,t)     starting time of i in k of l in t
INV(m,t)         inventory of m at the end of t
SQ(m,t)          supply amount of m in t
Ib(m,t)          backlog amount of m in t
Id(m,t)          deviation below IL(m) in t
Y(i,k,l,t)       assignment variable
YS(i,l,t)        spill over variable
GP               net profit  ;
POSITIVE VARIABLE        DQ,CL,PT,NC,SL,TS,TE,TTS,INV,SQ,Ib,Id,YS ;
BINARY VARIABLE          Y ;

EQUATIONS
Netprofit              net profit to be maximized
production(m,l,t)      amount of m produced or consumed on l in t
effectcamp(i,l,t)      effective production time of i on l in t
diffprod(i,l,t)        differential production amount of i on l in t
camlength(i,k,l,t)     upper limit for campaign length of i in k of l in t
ipers(k,l,t)           max no of i per k of l in t
speri(i,l,t)           max no of k for i on l in t
slotlen1(k,l,t)        length of slot k on l in t
slotlen2(l,t)          sum of slot lengths
numcamp(i,l,t)         number of campaigns of i on l in t
relax1(i,k,l,t)        relaxation of mcl constraint-1
relax2(i,k,l,t)        relaxation of mcl constraint-2
spilla(i,k,l,t)        condition-1 for spill over
spillb(i,*,l,t)        condition-2 for spill over
maxspill(l,t)          constraint for the number of spill over per k of l in t
startsame(k,l,t)       starting time of diff i on same l in t
endsame(k,l,t)         ending time of diff i on same l in t
starttask1(k,l,t)      starting time of task i assigned to slot k
```

```
start2diff(*,*,l,ll,t)    starting time of task 2 on diff l in t
start3diff(*,*,l,ll,t)    starting time of task 3 on diff l in t
start33diff(*,*,l,ll,t)   starting time of task 3 on diff l in t
start4diff(*,*,l,ll,t)    starting time of task 4 on diff l in t
start5diff(*,*,l,ll,t)    starting time of task 5 on diff l in t
end2diff(*,*,l,ll,t)      ending time of task 2 on diff l in t
end3diff(*,*,l,ll,t)      ending time of task 3 on diff l in t
end33diff(*,*,l,ll,t)     ending time of task 3 on diff l in t
end4diff(*,*,l,ll,t)      ending time of task 4 on diff l in t
end5diff(*,*,l,ll,t)      ending time of task 5 on diff l in t
inventory(m,t)            inventory balance equation
backlog(m,t)              backlogging equation
supply(m,t)               supply of m in t
dip(m,t)                  dip below target level of m in t ;


netprofit.. GP =e= sum((m,t), g(m)*SQ(m,t))- sum((m,t), ho1(m,t)*INV(m,t))-
sum((m,l,t), PQ(m,l,t)*ho2(m,t)/2)- sum((m,t), a(m,t)*Id(m,t))- sum((m,t),
g(m)*Ib(m,t))- sum((l,t),sum(i$il1(i,l), cc(i,l)*(NC(i,l,t)-YS(i,l,t-1))));
effectcamp(i,l,t)$(il1(i,l)).. PT(i,l,t) =e= sum(k$(kl(k,l)),CL(i,k,l,t))-CT(i,l)*(NC(i,l,t)-
YS(i,l,t-1));
production(m,l,t).. PQ(m,l,t) =e= sum(i$il1(i,l), sigma(m,i)*(RL(i,l)* PT(i,l,t)+
DQ(i,l,t))/meu(i));
diffprod(i,l,t)$(il1(i,l)).. DQ(i,l,t) =l= (RU(i,l)-RL(i,l))* PT(i,l,t);
camlength(i,k,l,t)$(il1(i,l) and kl(k,l)).. TTS(i,k,l,t)+ CL(i,k,l,t) =l= H(l,t)*Y(i,k,l,t);
ipers(k,l,t)$kl(k,l).. sum(i$il1(i,l), Y(i,k,l,t)) =l= 1;
speri(i,l,t)$il1(i,l).. sum(k$kl(k,l), Y(i,k,l,t)) =l= 1;
slotlen1(k,l,t)$(kl(k,l)).. SL(k,l,t) =e= sum(i$il1(i,l), CL(i,k,l,t));
slotlen2(l,t).. sum(k$kl(k,l), SL(k,l,t))=l= H(l,t);
numcamp(i,l,t)$(il1(i,l)).. NC(i,l,t) =e= sum(k$kl(k,l), Y(i,k,l,t));
relax1(i,k,l,t)$(il1(i,l)and kl(k,l)).. CL(i,k,l,t) =g= MCL(i,l)*Y(i,k,l,t)-
MCL(i,l)*(YS(i,l,t)$(ord(k)=nk(l))+YS(i,l,t-1)$(ord(k)=1));
relax2(i,k,l,t)$(il1(i,l)and kl(k,l) and ord(k)=nk(l)).. CL(i,k,l,t-1)+ CL(i,'1',l,t) =g=
MCL(i,l)*YS(i,l,t-1);
spilla(i,k,l,t)$(il1(i,l) and kl(k,l) and ord(k)=nk(l)).. YS(i,l,t) =l= Y(i,k,l,t);
spillb(i,'1',l,t)$(il1(i,l)and kl('1',l)).. YS(i,l,t-1) =l= Y(i,'1',l,t);
maxspill(l,t).. sum(i$il1(i,l), YS(i,l,t)) =l= 1;
startsame(k,l,t)$(kl(k,l) and ord(k)< nk(l)).. TS(k+1,l,t) =g= TE(k,l,t);
endsame(k,l,t)$(kl(k,l)).. TE(k,l,t)   =e= TS(k,l,t) + SL(k,l,t);
starttask1(k,l,t)$(kl(k,l)).. TS(k,l,t)   =e= sum(i$il1(i,l), TTS(i,k,l,t));
start2diff('1','2',l,ll,t)$(il1('2',l)and ill1('1',ll)).. sum(k$kl(k,l), TTS('2',k,l,t)) =g=
sum(kk$kkll(kk,ll), TTS('1',kk,ll,t)) + dt('1','2')*(NC('2',l,t) + NC('1',ll,t)-1);
start3diff('2','3',l,ll,t)$(il1('3',l)and ill1('2',ll)).. sum(k$kl(k,l), TTS('3',k,l,t)) =g=
sum(kk$kkll(kk,ll), TTS('2',kk,ll,t)) + dt('2','3')*(NC('3',l,t) + NC('2',ll,t)-1);
start33diff('5','3',l,ll,t)$(il1('3',l)and ill1('5',ll)).. sum(k$kl(k,l), TTS('3',k,l,t)) =g=
sum(kk$kkll(kk,ll), TTS('5',kk,ll,t)) + dt('5','3')*(NC('3',l,t) + NC('5',ll,t)-1);
start4diff('4','5',l,ll,t)$(il1('5',l)and ill1('4',ll)).. sum(k$kl(k,l), TTS('5',k,l,t)) =g=
sum(kk$kkll(kk,ll), TTS('4',kk,ll,t)) + dt('4','5')*(NC('5',l,t) + NC('4',ll,t)-1);
```

```
start5diff('5','6',l,ll,t)$(il1('6',l)and ill1('5',ll)).. sum(k$kl(k,l), TTS('6',k,l,t)) =g=
sum(kk$kkll(kk,ll), TTS('5',kk,ll,t)) + dt('5','6')*(NC('6',l,t) + NC('5',ll,t)-1);
end2diff('1','2',l,ll,t)$(il1('2',l)and ill1('1',ll)).. sum(k$kl(k,l), TTS('2',k,l,t)+
CL('2',k,l,t)) =g= sum(kk$kkll(kk,ll), TTS('1',kk,ll,t)+ CL('1',kk,ll,t)) +
dt('1','2')*(NC('2',l,t) + NC('1',ll,t)-1);
end3diff('2','3',l,ll,t)$(il1('3',l)and ill1('2',ll)).. sum(k$kl(k,l), TTS('3',k,l,t)+
CL('3',k,l,t)) =g= sum(kk$kkll(kk,ll), TTS('2',kk,ll,t)+ CL('2',kk,ll,t)) +
dt('2','3')*(NC('3',l,t) + NC('2',ll,t)-1);
end33diff('5','3',l,ll,t)$(il1('3',l)and ill1('5',ll)).. sum(k$kl(k,l), TTS('3',k,l,t)+
CL('3',k,l,t)) =g= sum(kk$kkll(kk,ll), TTS('5',kk,ll,t)+ CL('5',kk,ll,t)) +
dt('5','3')*(NC('3',l,t) + NC('5',ll,t)-1);
end4diff('4','5',l,ll,t)$(il1('5',l)and ill1('4',ll)).. sum(k$kl(k,l), TTS('5',k,l,t)+
CL('5',k,l,t)) =g= sum(kk$kkll(kk,ll), TTS('4',kk,ll,t)+ CL('4',kk,ll,t)) +
dt('4','5')*(NC('5',l,t) + NC('4',ll,t)-1);
end5diff('5','6',l,ll,t)$(il1('6',l)and ill1('5',ll)).. sum(k$kl(k,l), TTS('6',k,l,t)+
CL('6',k,l,t)) =g= sum(kk$kkll(kk,ll), TTS('5',kk,ll,t)+ CL('5',kk,ll,t)) +
dt('5','6')*(NC('6',l,t) + NC('5',ll,t)-1);
inventory(m,t).. INV(m,t) =e= (I0(m))$(ord(t)=1)+(INV(m,t-1))$(ord(t)>1)+ sum(l,
PQ(m,l,t))-SQ(m,t);
backlog(m,t).. Ib(m,t) =g= Ib(m,t-1)$(ord(t)>1)+ D(m,t)-SQ(m,t);
supply(m,t).. sum(tt$(ord(tt)<= ord(t)), SQ(m,tt)) =l= sum(tt, D(m,tt));
dip(m,t).. Id(m,t) =g= IL(m)- INV(m,t);

CL.lo(i,k,l,t) = 0;
PT.lo(i,l,t) = 0;
SL.lo(k,l,t) = 0;
TS.lo(k,l,t) = 0;
TE.lo(k,l,t) = 0;
TTS.lo(i,k,l,t) = 0;
INV.lo(m,t) = 0;
Ib.lo(m,t) = 0;
Id.lo(m,t) = 0;
NC.lo(i,l,t) = 0;
CL.up(i,k,l,t) = H(l,t);
PT.up(i,l,t) = H(l,t);
SL.up(k,l,t) = H(l,t);
TS.up(k,l,t) = H(l,t);
TE.up(k,l,t) = H(l,t);
TTS.up(i,k,l,t) = H(l,t);
INV.up(m,t) = ST(m);
SQ.up(m,t) = sum(tt$(ord(tt) <= ord(t)), D(m,tt));
Ib.up(m,t) = sum(tt$(ord(tt) <= ord(t)), D(m,tt));
Id.up(m,t) = IL(m);
NC.up(i,l,t) = 1;

OPTION
SOLPRINT = OFF
limrow = 120
```

```
optcr = 0
reslim = 1000
mip = cplex ;

MODEL       planning /all/ ;

SOLVE       planning using mip maximizing GP ;

DISPLAY     GP.l,CL.l,PT.l,TS.l,TE.l,TTS.l,Y.l,YS.l,INV.l,PQ.l,Ib.l,Id.l,SQ.l;
```

A.1.1.2 Example 1b

```
VARIABLES
NC(i,l,t)      number of campaigns of i on l in t
PQ(m,l,t)      amount of m produced or consumed on l in t
DQ(i,l,t)      differential production amount of i on l in t
CL(i,k,l,t)    campaign length of i in k of l in t
PT(i,l,t)      effective production time of i on l in t
SL(k,l,t)      length of k of l in t
TS(k,l,t)      starting time of k of l in t
TE(k,l,t)      ending time of k of l in t
TTS(i,k,l,t)   starting time of i in k of l in t
INV(m,t)       inventory of m at the end of t
SQ(m,t)        supply amount of m in t
Ib(m,t)        backlog amount of m in t
Id(m,t)        deviation below IL(m) in t
Y(i,k,l,t)     assignment variable
YS(i,l,t)      spill over variable
Z(i,k,l,t)     scaleup variable
GP             net profit  ;
POSITIVE VARIABLE      DQ,CL,PT,NC,SL,TS,TE,TTS,INV,SQ,Ib,Id,Z,YS ;
BINARY VARIABLE        Y;

EQUATIONS
netprofit              net profit to be maximized
production(m,l,t)      amount of m produced or consumed on l in t
effectcamp(i,l,t)      effective production time of i on l in t
diffprod(i,l,t)        differential production amount of i on l in t
camlength(i,k,l,t)     upper limit for campaign length of i in k of l in t
ipers(k,l,t)           max no of i per k of l in t
speri(i,l,t)           max no of k for i on l in t
slotlen1(k,l,t)        length of slot k on l in t
slotlen2(l,t)          sum of slot lengths
numcamp(i,l,t)         number of campaigns of i on l in t
relax1(i,k,l,t)        relaxation of mcl constraint-1
relax2(i,k,l,t)        relaxation of mcl constraint-2
spilla(i,k,l,t)        condition-1 for spill over
spillb(i,*,l,t)        condition-2 for spill over
```

```
maxspill(l,t)            constraint for the number of spill over per k of l in t
scaleup1(i,k,l,t)        scaleup time required for i in k of l in t
scaleup2(i,l)            scaleup time required for i on l in t
startsame(k,l,t)         starting time of diff i on same l in t
endsame(k,l,t)           ending time of diff i on same l in t
starttask1(k,l,t)        starting time of task i assigned to slot k
start2diff(*,*,l,ll,t)   starting time of task 2 on diff l in t
start3diff(*,*,l,ll,t)   starting time of task 3 on diff l in t
start33diff(*,*,l,ll,t)  starting time of task 3 on diff l in t
start4diff(*,*,l,ll,t)   starting time of task 4 on diff l in t
start5diff(*,*,l,ll,t)   starting time of task 5 on diff l in t
start6diff(*,*,l,ll,t)   starting time of task 6 on diff l in t
start7diff(*,*,l,ll,t)   starting time of task 7 on diff l in t
end2diff(*,*,l,ll,t)     ending time of task 2 on diff l in t
end3diff(*,*,l,ll,t)     ending time of task 3 on diff l in t
end33diff(*,*,l,ll,t)    ending time of task 3 on diff l in t
end4diff(*,*,l,ll,t)     ending time of task 4 on diff l in t
end5diff(*,*,l,ll,t)     ending time of task 5 on diff l in t
end6diff(*,*,l,ll,t)     ending time of task 6 on diff l in t
end7diff(*,*,l,ll,t)     ending time of task 7 on diff l in t
inventory(m,t)           inventory balance equation
backlog(m,t)             backlogging equation
supply(m,t)              supply of m in t
dip(m,t)                 dip below target level of m in t ;

netprofit.. GP =e= sum((m,t), g(m)*SQ(m,t))- sum((m,t), ho1(m,t)*INV(m,t))-
sum((m,l,t), PQ(m,l,t)*ho2(m,t)/2)- sum((m,t), a(m,t)*Id(m,t))- sum((m,t),
g(m)*Ib(m,t))- sum((l,t),sum(i$il1(i,l), cc(i,l)*(NC(i,l,t)-YS(i,l,t-1))));
effectcamp(i,l,t)$(il1(i,l)).. PT(i,l,t) =e= sum(k$(kl(k,l)),CL(i,k,l,t))-CT(i,l)*(NC(i,l,t)-
YS(i,l,t-1))- sum(k$kl(k,l), SUT(i,l)*Z(i,k,l,t));
production(m,l,t).. PQ(m,l,t) =e= sum(i$il1(i,l), sigma(m,i)*(RL(i,l)* PT(i,l,t)+
DQ(i,l,t))/meu(i));
diffprod(i,l,t)$(il1(i,l)).. DQ(i,l,t) =l= (RU(i,l)-RL(i,l))* PT(i,l,t);
camlength(i,k,l,t)$(il1(i,l)and kl(k,l)).. TTS(i,k,l,t)+ CL(i,k,l,t) =l= H(l,t)*Y(i,k,l,t);
ipers(k,l,t)$kl(k,l).. sum(i$il1(i,l), Y(i,k,l,t)) =l= 1;
speri(i,l,t)$il1(i,l).. sum(k$kl(k,l), Y(i,k,l,t)) =l= 1;
slotlen1(k,l,t)$(kl(k,l)).. SL(k,l,t) =e= sum(i$il1(i,l), CL(i,k,l,t));
slotlen2(l,t).. sum(k$kl(k,l), SL(k,l,t))=l= H(l,t);
numcamp(i,l,t)$(il1(i,l)).. NC(i,l,t) =e= sum(k$kl(k,l), Y(i,k,l,t));
relax1(i,k,l,t)$(il1(i,l)and kl(k,l)).. CL(i,k,l,t) =g= MCL(i,l)*Y(i,k,l,t)-
MCL(i,l)*(YS(i,l,t)$(ord(k)=nk(l))+YS(i,l,t-1)$(ord(k)=1));
relax2(i,k,l,t)$(il1(i,l)and kl(k,l) and ord(k)=nk(l)).. CL(i,k,l,t-1)+ CL(i,'1',l,t) =g=
MCL(i,l)*YS(i,l,t-1);
spilla(i,k,l,t)$(il1(i,l) and kl(k,l) and ord(k)=nk(l)).. YS(i,l,t) =l= Y(i,k,l,t);
spillb(i,'1',l,t)$(il1(i,l)and kl('1',l)).. YS(i,l,t-1) =l= Y(i,'1',l,t);
maxspill(l,t).. sum(i$il1(i,l), YS(i,l,t)) =l= 1;
scaleup1(i,k,l,t)$(il1(i,l)and kl(k,l)).. Z(i,k,l,t) =g= Y(i,k,l,t)-
sum(tt$(ord(tt)<ord(t)),sum(kk,Z(i,kk,l,tt)));
```

```
scaleup2(i,l)$(il1(i,l)).. sum(t,sum(k$(kl(k,l)),Z(i,k,l,t))) =l= 1;
startsame(k,l,t)$(kl(k,l)and ord(k)< nk(l)).. TS(k+1,l,t) =g= TE(k,l,t);
endsame(k,l,t)$(kl(k,l)).. TE(k,l,t)   =e= TS(k,l,t) + SL(k,l,t);
starttask1(k,l,t)$(kl(k,l)).. TS(k,l,t)   =e= sum(i$il1(i,l), TTS(i,k,l,t));
start2diff('1','2',l,ll,t)$(il1('2',l)and ill1('1',ll)).. sum(k$kl(k,l), TTS('2',k,l,t)) =g=
sum(kk$kkll(kk,ll), TTS('1',kk,ll,t)) + dt('1','2')*(NC('2',l,t) + NC('1',ll,t)-1);
start3diff('2','3',l,ll,t)$(il1('3',l)and ill1('2',ll)).. sum(k$kl(k,l), TTS('3',k,l,t)) =g=
sum(kk$kkll(kk,ll), TTS('2',kk,ll,t)) + dt('2','3')*(NC('3',l,t) + NC('2',ll,t)-1);
start33diff('5','3',l,ll,t)$(il1('3',l)and ill1('5',ll)).. sum(k$kl(k,l), TTS('3',k,l,t)) =g=
sum(kk$kkll(kk,ll), TTS('5',kk,ll,t)) + dt('5','3')*(NC('3',l,t) + NC('5',ll,t)-1);
start4diff('4','5',l,ll,t)$(il1('5',l)and ill1('4',ll)).. sum(k$kl(k,l), TTS('5',k,l,t)) =g=
sum(kk$kkll(kk,ll), TTS('4',kk,ll,t)) + dt('4','5')*(NC('5',l,t) + NC('4',ll,t)-1);
start5diff('5','6',l,ll,t)$(il1('6',l)and ill1('5',ll)).. sum(k$kl(k,l), TTS('6',k,l,t)) =g=
sum(kk$kkll(kk,ll), TTS('5',kk,ll,t)) + dt('5','6')*(NC('6',l,t) + NC('5',ll,t)-1);
start6diff('1','7',l,ll,t)$(il1('7',l)and ill1('1',ll)).. sum(k$kl(k,l), TTS('7',k,l,t)) =g=
sum(kk$kkll(kk,ll), TTS('1',kk,ll,t)) + dt('1','7')*(NC('7',l,t) + NC('1',ll,t)-1);
start7diff('7','8',l,ll,t)$(il1('8',l)and ill1('7',ll)).. sum(k$kl(k,l), TTS('8',k,l,t)) =g=
sum(kk$kkll(kk,ll), TTS('7',kk,ll,t)) + dt('7','8')*(NC('8',l,t) + NC('7',ll,t)-1);
end2diff('1','2',l,ll,t)$(il1('2',l)and ill1('1',ll)).. sum(k$kl(k,l), TTS('2',k,l,t)+
CL('2',k,l,t)) =g= sum(kk$kkll(kk,ll), TTS('1',kk,ll,t)+ CL('1',kk,ll,t)) +
dt('1','2')*(NC('2',l,t) + NC('1',ll,t)-1);
end3diff('2','3',l,ll,t)$(il1('3',l)and ill1('2',ll)).. sum(k$kl(k,l), TTS('3',k,l,t)+
CL('3',k,l,t)) =g= sum(kk$kkll(kk,ll), TTS('2',kk,ll,t)+ CL('2',kk,ll,t)) +
dt('2','3')*(NC('3',l,t) + NC('2',ll,t)-1);
end33diff('5','3',l,ll,t)$(il1('3',l)and ill1('5',ll)).. sum(k$kl(k,l), TTS('3',k,l,t)+
CL('3',k,l,t)) =g= sum(kk$kkll(kk,ll), TTS('5',kk,ll,t)+ CL('5',kk,ll,t)) +
dt('5','3')*(NC('3',l,t) + NC('5',ll,t)-1);
end4diff('4','5',l,ll,t)$(il1('5',l)and ill1('4',ll)).. sum(k$kl(k,l), TTS('5',k,l,t)+
CL('5',k,l,t)) =g= sum(kk$kkll(kk,ll), TTS('4',kk,ll,t)+ CL('4',kk,ll,t)) +
dt('4','5')*(NC('5',l,t) + NC('4',ll,t)-1);
end5diff('5','6',l,ll,t)$(il1('6',l)and ill1('5',ll)).. sum(k$kl(k,l), TTS('6',k,l,t)+
CL('6',k,l,t)) =g= sum(kk$kkll(kk,ll), TTS('5',kk,ll,t)+ CL('5',kk,ll,t)) +
dt('5','6')*(NC('6',l,t) + NC('5',ll,t)-1);
end6diff('1','7',l,ll,t)$(il1('7',l)and ill1('1',ll)).. sum(k$kl(k,l), TTS('7',k,l,t)+
CL('7',k,l,t)) =g= sum(kk$kkll(kk,ll), TTS('1',kk,ll,t)+ CL('1',kk,ll,t)) +
dt('1','7')*(NC('7',l,t) + NC('1',ll,t)-1);
end7diff('7','8',l,ll,t)$(il1('8',l)and ill1('7',ll)).. sum(k$kl(k,l), TTS('8',k,l,t)+
CL('8',k,l,t)) =g= sum(kk$kkll(kk,ll), TTS('7',kk,ll,t)+ CL('7',kk,ll,t)) +
dt('7','8')*(NC('8',l,t) + NC('7',ll,t)-1);
inventory(m,t).. INV(m,t) =e= (I0(m))$(ord(t)=1)+(INV(m,t-1))$(ord(t)>1)+ sum(l,
PQ(m,l,t))-SQ(m,t);
backlog(m,t).. Ib(m,t) =g= Ib(m,t-1)$(ord(t)>1)+ D(m,t)-SQ(m,t);
supply(m,t).. sum(tt$(ord(tt)<= ord(t)), SQ(m,tt)) =l= sum(tt, D(m,tt));
dip(m,t).. Id(m,t) =g= IL(m)- INV(m,t);

CL.lo(i,k,l,t) = 0;
PT.lo(i,l,t) = 0;
SL.lo(k,l,t) = 0;
```

```
TS.lo(k,l,t) = 0;
TE.lo(k,l,t) = 0;
TTS.lo(i,k,l,t) = 0;
INV.lo(m,t) = 0;
Ib.lo(m,t) = 0;
Id.lo(m,t) = 0;
NC.lo(i,l,t) = 0;
Z.lo(i,k,l,t) = 0;
YS.lo(i,l,t) = 0;

CL.up(i,k,l,t) = H(l,t);
PT.up(i,l,t) = H(l,t);
SL.up(k,l,t) = H(l,t);
TS.up(k,l,t) = H(l,t);
TE.up(k,l,t) = H(l,t);
TTS.up(i,k,l,t) = H(l,t);
INV.up(m,t) = ST(m);
SQ.up(m,t) = sum(tt$(ord(tt) <= ord(t)), D(m,tt));
Ib.up(m,t) = sum(tt$(ord(tt) <= ord(t)), D(m,tt));
Id.up(m,t) = IL(m);
NC.up(i,l,t) = 1;
Z.up(i,k,l,t) = 1;
YS.up(i,l,t) = 1;

OPTION
SOLPRINT = OFF
limrow = 60
optcr = 0
reslim = 10000
mip = cplex ;

MODEL      planning /all/;

SOLVE      planning using mip maximizing GP;

DISPLAY    GP.l,CL.l,PT.l,TS.l,TE.l,TTS.l,Y.l,Z.l,YS.l,INV.l,PQ.l,Ib.l,Id.l,SQ.l;
```

A.1.2 Example 2

```
VARIABLES
NC(i,l,t)      number of campaigns of i on l in t
PQ(m,l,t)      amount of m produced or consumed on l in t
DQ(i,l,t)      Differential production amount of i on l in t
CL(i,k,l,t)    campaign length of i in k of l in t
PT(i,l,t)      effective production time of i on l in t
SL(k,l,t)      length of k of l in t
TS(k,l,t)      starting time of k of l in t
TE(k,l,t)      ending time of k of l in t
```

```
TTS(i,k,l,t)      starting time of i in k of l in t
INV(m,t)          inventory of m at the end of t
SQ(m,t)           supply amount of m in t
OQ(m,t)           outsourcing amount of m in t
Ib(m,t)           backlog amount of m in t
Id(m,t)           deviation below IL(m) in t
Y(i,k,l,t)        assignment variable
YS(i,l,t)         spill over variable
GP                net profit  ;
POSITIVE VARIABLE       DQ,CL,PT,NC,SL,TS,TE,TTS,INV,SQ,OQ,Ib,Id,YS ;
BINARY VARIABLE         Y ;

EQUATIONS
netprofit                 net profit to be maximized
production(m,l,t)         amount of m produced or consumed on l in t
effectcamp(i,l,t)         effective production time of i on l in t
diffprod(i,l,t)           differential production amount of i on l in t
camlength(i,k,l,t)        upper limit for campaign length of i in k of l in t
ipers(k,l,t)              max no of i per k of l in t
speri(i,l,t)              max no of k for i on l in t
slotlen1(k,l,t)           length of slot k on l in t
slotlen2(l,t)             sum of slot lengths
numcamp(i,l,t)            number of campaigns of i on l in t
relax1(i,k,l,t)           relaxation of mcl constraint-1
relax2(i,k,l,t)           relaxation of mcl constraint-2
spilla(i,k,l,t)           condition-1 for spill over
spillb(i,*,l,t)           condition-2 for spill over
maxspill(l,t)             constraint for the number of spill over per k of l in t
startsame(k,l,t)          starting time of diff i on same l in t
endsame(k,l,t)            ending time of diff i on same l in t
starttask1(k,l,t)         starting time of task i assigned to slot k
start2diff(*,*,l,ll,t)    starting time of task 2 on diff l in t
start3diff(*,*,l,ll,t)    starting time of task 3 on diff l in t
start33diff(*,*,l,ll,t)   starting time of task 3 on diff l in t
start4diff(*,*,l,ll,t)    starting time of task 4 on diff l in t
start5diff(*,*,l,ll,t)    starting time of task 5 on diff l in t
end2diff(*,*,l,ll,t)      ending time of task 2 on diff l in t
end3diff(*,*,l,ll,t)      ending time of task 3 on diff l in t
end33diff(*,*,l,ll,t)     ending time of task 3 on diff l in t
end4diff(*,*,l,ll,t)      ending time of task 4 on diff l in t
end5diff(*,*,l,ll,t)      ending time of task 5 on diff l in t
inventory(m,t)            inventory balance equation
backlog(m,t)              backlogging equation
supply(m,t)               supply of m in t
dip(m,t)                  dip below target level of m in t ;

netprofit.. GP =e= sum((m,t), g(m)*SQ(m,t))- sum((m,t), ho1(m,t)*INV(m,t))-
sum((m,l,t), PQ(m,l,t)*ho2(m,t)/2)- sum((m,t), a(m,t)*Id(m,t))- sum((m,t),
```

```
g(m)*Ib(m,t))- sum((m,t), rc(m)* OQ(m,t))- sum((l,t),sum(i$il1(i,l), cc(i,l)*(NC(i,l,t)-
YS(i,l,t-1))));
effectcamp(i,l,t)$(il1(i,l)).. PT(i,l,t) =e= sum(k$(kl(k,l)),CL(i,k,l,t))-CT(i,l)*(NC(i,l,t)-
YS(i,l,t-1));
production(m,l,t).. PQ(m,l,t) =e= sum(i$il1(i,l), sigma(m,i)*(RL(i,l)* PT(i,l,t)+
DQ(i,l,t))/meu(i));
diffprod(i,l,t)$(il1(i,l)).. DQ(i,l,t) =l= (RU(i,l)-RL(i,l))* PT(i,l,t);
camlength(i,k,l,t)$(il1(i,l) and kl(k,l)).. TTS(i,k,l,t)+ CL(i,k,l,t) =l= H(l,t)*Y(i,k,l,t);
ipers(k,l,t)$kl(k,l).. sum(i$il1(i,l), Y(i,k,l,t)) =l= 1;
speri(i,l,t)$il1(i,l).. sum(k$kl(k,l), Y(i,k,l,t)) =l= 1;
slotlen1(k,l,t)$(kl(k,l)).. SL(k,l,t) =e= sum(i$il1(i,l), CL(i,k,l,t));
slotlen2(l,t).. sum(k$kl(k,l), SL(k,l,t))=l= H(l,t);
numcamp(i,l,t)$(il1(i,l)).. NC(i,l,t) =e= sum(k$kl(k,l), Y(i,k,l,t));
relax1(i,k,l,t)$(il1(i,l)and kl(k,l)).. CL(i,k,l,t) =g= MCL*Y(i,k,l,t)-
MCL*(YS(i,l,t)$(ord(k)=nk(l))+YS(i,l,t-1)$(ord(k)=1));
relax2(i,k,l,t)$(il1(i,l)and kl(k,l) and ord(k)=nk(l)).. CL(i,k,l,t-1)+ CL(i,'1',l,t) =g=
MCL*YS(i,l,t-1);
spilla(i,k,l,t)$(il1(i,l) and kl(k,l) and ord(k)=nk(l)).. YS(i,l,t) =l= Y(i,k,l,t);
spillb(i,'1',l,t)$(il1(i,l)and kl('1',l)).. YS(i,l,t-1) =l= Y(i,'1',l,t);
maxspill(l,t).. sum(i$il1(i,l), YS(i,l,t)) =l= 1;
startsame(k,l,t)$(kl(k,l) and ord(k)< nk(l)).. TS(k+1,l,t) =g= TE(k,l,t);
endsame(k,l,t)$(kl(k,l)).. TE(k,l,t)   =e= TS(k,l,t) + SL(k,l,t);
starttask1(k,l,t)$(kl(k,l)).. TS(k,l,t)   =e= sum(i$il1(i,l), TTS(i,k,l,t));
start2diff('1','2',l,ll,t)$(il1('2',l)and ill1('1',ll)).. sum(k$kl(k,l), TTS('2',k,l,t)) =g=
sum(kk$kkll(kk,ll), TTS('1',kk,ll,t)) + dt('1','2')*(NC('2',l,t) + NC('1',ll,t)-1);
start3diff('2','3',l,ll,t)$(il1('3',l)and ill1('2',ll)).. sum(k$kl(k,l), TTS('3',k,l,t)) =g=
sum(kk$kkll(kk,ll), TTS('2',kk,ll,t)) + dt('2','3')*(NC('3',l,t) + NC('2',ll,t)-1);
start33diff('4','3',l,ll,t)$(il1('3',l)and ill1('4',ll)).. sum(k$kl(k,l), TTS('3',k,l,t)) =g=
sum(kk$kkll(kk,ll), TTS('4',kk,ll,t)) + dt('4','3')*(NC('3',l,t) + NC('4',ll,t)-1);
start4diff('4','5',l,ll,t)$(il1('5',l)and ill1('4',ll)).. sum(k$kl(k,l), TTS('5',k,l,t)) =g=
sum(kk$kkll(kk,ll), TTS('4',kk,ll,t)) + dt('4','5')*(NC('5',l,t) + NC('4',ll,t)-1);
start5diff('1','4',l,ll,t)$(il1('4',l)and ill1('1',ll)).. sum(k$kl(k,l), TTS('4',k,l,t)) =g=
sum(kk$kkll(kk,ll), TTS('1',kk,ll,t)) + dt('1','4')*(NC('4',l,t) + NC('1',ll,t)-1);
end2diff('1','2',l,ll,t)$(il1('2',l)and ill1('1',ll)).. sum(k$kl(k,l), TTS('2',k,l,t)+
CL('2',k,l,t)) =g= sum(kk$kkll(kk,ll), TTS('1',kk,ll,t)+ CL('1',kk,ll,t)) +
dt('1','2')*(NC('2',l,t) + NC('1',ll,t)-1);
end3diff('2','3',l,ll,t)$(il1('3',l)and ill1('2',ll)).. sum(k$kl(k,l), TTS('3',k,l,t)+
CL('3',k,l,t)) =g= sum(kk$kkll(kk,ll), TTS('2',kk,ll,t)+ CL('2',kk,ll,t)) +
dt('2','3')*(NC('3',l,t) + NC('2',ll,t)-1);
end33diff('4','3',l,ll,t)$(il1('3',l)and ill1('4',ll)).. sum(k$kl(k,l), TTS('3',k,l,t)+
CL('3',k,l,t)) =g= sum(kk$kkll(kk,ll), TTS('4',kk,ll,t)+ CL('4',kk,ll,t)) +
dt('4','3')*(NC('3',l,t) + NC('4',ll,t)-1);
end4diff('4','5',l,ll,t)$(il1('5',l)and ill1('4',ll)).. sum(k$kl(k,l), TTS('5',k,l,t)+
CL('5',k,l,t)) =g= sum(kk$kkll(kk,ll), TTS('4',kk,ll,t)+ CL('4',kk,ll,t)) +
dt('4','5')*(NC('5',l,t) + NC('4',ll,t)-1);
end5diff('1','4',l,ll,t)$(il1('4',l)and ill1('1',ll)).. sum(k$kl(k,l), TTS('4',k,l,t)+
CL('4',k,l,t)) =g= sum(kk$kkll(kk,ll), TTS('1',kk,ll,t)+ CL('1',kk,ll,t)) +
dt('1','4')*(NC('4',l,t) + NC('1',ll,t)-1);
```

```
inventory(m,t).. INV(m,t) =e= (I0(m))$(ord(t)=1)+(INV(m,t-1))$(ord(t)>1)+ OQ(m,t)+
sum(l, PQ(m,l,t))-SQ(m,t);
backlog(m,t).. Ib(m,t) =g= Ib(m,t-1)$(ord(t)>1)+ D(m,t)-SQ(m,t);
supply(m,t).. sum(tt$(ord(tt)<= ord(t)), SQ(m,tt)) =l= sum(tt, D(m,tt));
dip(m,t).. Id(m,t) =g= IL(m)- INV(m,t);

OQ.lo(m,t) = 0;
CL.lo(i,k,l,t) = 0;
PT.lo(i,l,t) = 0;
SL.lo(k,l,t) = 0;
TS.lo(k,l,t) = 0;
TE.lo(k,l,t) = 0;
TTS.lo(i,k,l,t) = 0;
INV.lo(m,t) = 0;
Ib.lo(m,t) = 0;
Id.lo(m,t) = 0;
NC.lo(i,l,t) = 0;
YS.lo(i,l,t) = 0;

OQ.up(m,t) = out(m);
CL.up(i,k,l,t) = H(l,t);
PT.up(i,l,t) = H(l,t);
SL.up(k,l,t) = H(l,t);
TS.up(k,l,t) = H(l,t);
TE.up(k,l,t) = H(l,t);
TTS.up(i,k,l,t) = H(l,t);
INV.up(m,t) = ST(m);
SQ.up(m,t) = sum(tt$(ord(tt) <= ord(t)), D(m,tt));
Ib.up(m,t) = sum(tt$(ord(tt) <= ord(t)), D(m,tt));
Id.up(m,t) = IL(m);
NC.up(i,l,t) = 1;
YS.up(i,l,t) = 1;

OPTION
SOLPRINT = OFF
limrow = 40
optcr = 0
reslim = 1000
mip = cplex ;

MODEL       planning /all/ ;

SOLVE       planning using mip maximizing GP ;

DISPLAY     GP.l,CL.l,PT.l,TS.l,TE.l,TTS.l,Y.l,YS.l,INV.l,PQ.l,Ib.l,Id.l,SQ.l;
```

A.1.3 Example 3

```
VARIABLES
NC(i,l,t)        number of campaigns of i on l in t
PQ(m,l,t)        mount of m produced or consumed on l in t
DQ(i,l,t)        differential production amount of i on l in t
CL(i,k,l,t)      campaign length of i in k of l in t
PT(i,l,t)        effective production time of i on l in t
SL(k,l,t)        ength of k of l in t
TS(k,l,t)        starting time of k of l in t
TE(k,l,t)        ending time of k of l in t
TTS(i,k,l,t)     starting time of i in k of l in t
INV(m,t)         inventory of m at the end of t
SQ(m,t)          supply amount of m in t
OQ(m,t)          outsourcing amount of m in t
Ib(m,t)          backlog amount of m in t
Id(m,t)          deviation below IL(m) in t
Y(i,k,l,t)       assignment variable
YS(i,l,t)        spill over variable
Z(i,k,l,t)       scaleup variable
GP               net profit  ;
POSITIVE VARIABLE       DQ,CL,PT,NC,SL,TS,TE,TTS,INV,SQ,OQ,Ib,Id,Z,YS ;
BINARY VARIABLE         Y ;

EQUATIONS
netprofit              net profit to be maximized
production(m,l,t)      amount of m produced or consumed on l in t
effectcamp(i,l,t)      effective production time of i on l in t
diffprod(i,l,t)        differential production amount of i on l in t
camlength(i,k,l,t)     upper limit for campaign length of i in k of l in t
ipers(k,l,t)           max no of i per k of l in t
speri(i,l,t)           max no of k for i on l in t
slotlen1(k,l,t)        length of slot k on l in t
slotlen2(l,t)          sum of slot lengths
numcamp(i,l,t)         number of campaigns of i on l in t
relax1(i,k,l,t)        relaxation of mcl constraint-1
relax2(i,k,l,t)        relaxation of mcl constraint-2
spilla(i,k,l,t)        condition-1 for spill over
spillb(i,*,l,t)        condition-2 for spill over
maxspill(l,t)          constraint for the number of spill over per k of l in t
scaleup1(i,k,l,t)      scaleup time required for i in k of l in t
scaleup2(i,l)          scaleup time required for i on l in t
startsame(k,l,t)       starting time of diff i on same l in t
endsame(k,l,t)         ending time of diff i on same l in t
starttask1(k,l,t)      starting time of task i assigned to slot k
start2diff(*,*,l,ll,t) starting time of task 2 on diff l in t
start3diff(*,*,l,ll,t) starting time of task 3 on diff l in t
start33diff(*,*,l,ll,t) starting time of task 3 on diff l in t
```

```
start4diff(*,*,l,ll,t)    starting time of task 4 on diff l in t
start5diff(*,*,l,ll,t)    starting time of task 5 on diff l in t
start6diff(*,*,l,ll,t)    starting time of task 4 on diff l in t
start7diff(*,*,l,ll,t)    starting time of task 5 on diff l in t
end2diff(*,*,l,ll,t)      ending time of task 2 on diff l in t
end3diff(*,*,l,ll,t)      ending time of task 3 on diff l in t
end33diff(*,*,l,ll,t)     ending time of task 3 on diff l in t
end4diff(*,*,l,ll,t)      ending time of task 4 on diff l in t
end5diff(*,*,l,ll,t)      ending time of task 5 on diff l in t
end6diff(*,*,l,ll,t)      ending time of task 4 on diff l in t
end7diff(*,*,l,ll,t)      ending time of task 5 on diff l in t
inventory(m,t)            inventory balance equation
backlog(m,t)              backlogging equation
supply(m,t)               supply of m in t
dip(m,t)                  dip below target level of m in t ;

netprofit.. GP =e= sum((m,t), g(m)*SQ(m,t))- sum((m,t), ho1(m,t)*INV(m,t))-
sum((m,l,t), PQ(m,l,t)*ho2(m,t)/2)- sum((m,t), a(m,t)*Id(m,t))- sum((m,t),
g(m)*Ib(m,t))- sum((l,t),sum(i$il1(i,l), cc(i,l)*(NC(i,l,t)-YS(i,l,t-1)$(ord(t)>1)-
YS0(i,l)$(ord(t)=1))));
effectcamp(i,l,t)$(il1(i,l)).. PT(i,l,t) =e= sum(k$(kl(k,l)),CL(i,k,l,t))-CT(i,l)*(NC(i,l,t)-
YS(i,l,t-1)$(ord(t)>1)-YS0(i,l)$(ord(t)=1))- sum(k$kl(k,l), SUT(i,l)*Z(i,k,l,t));
production(m,l,t).. PQ(m,l,t) =e= sum(i$il1(i,l), sigma(m,i)*(RL(i,l)* PT(i,l,t)+
DQ(i,l,t))/meu(i));
diffprod(i,l,t)$(il1(i,l)).. DQ(i,l,t) =l= (RU(i,l)-RL(i,l))* PT(i,l,t);
camlength(i,k,l,t)$(il1(i,l)and kl(k,l)).. TTS(i,k,l,t)+ CL(i,k,l,t) =l= H(l,t)*Y(i,k,l,t);
ipers(k,l,t)$kl(k,l).. sum(i$il1(i,l), Y(i,k,l,t)) =l= 1;
speri(i,l,t)$il1(i,l).. sum(k$kl(k,l), Y(i,k,l,t)) =l= 1;
slotlen1(k,l,t)$(kl(k,l)).. SL(k,l,t) =e= sum(i$il1(i,l), CL(i,k,l,t));
slotlen2(l,t).. sum(k$kl(k,l), SL(k,l,t))=l= H(l,t);
numcamp(i,l,t)$(il1(i,l)).. NC(i,l,t) =e= sum(k$kl(k,l), Y(i,k,l,t));
relax1(i,k,l,t)$(il1(i,l)and kl(k,l)).. CL(i,k,l,t) =g= MCL*Y(i,k,l,t)-
MCL*(YS(i,l,t)$(ord(k)=nk(l))+YS(i,l,t-1)$(ord(k)=1 and
ord(t)>1)+YS0(i,l)$(ord(k)=1 and ord(t)=1));
relax2(i,k,l,t)$(il1(i,l)and kl(k,l)and ord(k)=nk(l)).. CL(i,'1',l,t) =g= MCL*(YS(i,l,t-
1)$(ord(t)>1)+YS0(i,l)$(ord(t)=1))-CL(i,k,l,t-1)$(ord(t)>1)-CL0(i,l)$(ord(t)=1);
spilla(i,k,l,t)$(il1(i,l) and kl(k,l) and ord(k)=nk(l)).. YS(i,l,t) =l= Y(i,k,l,t);
spillb(i,'1',l,t)$(il1(i,l)and kl('1',l)).. Y(i,'1',l,t) =g= YS(i,l,t-
1)$(ord(t)>1)+YS0(i,l)$(ord(t)=1);
maxspill(l,t).. sum(i$il1(i,l), YS(i,l,t)) =l= 1;
scaleup1(i,k,l,t)$(il1(i,l)and kl(k,l)).. Z(i,k,l,t) =g= Y(i,k,l,t)-
sum(tt$(ord(tt)<ord(t)),sum(kk,Z(i,kk,l,tt)));
scaleup2(i,l)$(il1(i,l)).. sum(t,sum(k$(kl(k,l)),Z(i,k,l,t))) =l= 1;
startsame(k,l,t)$(kl(k,l)and ord(k)< nk(l)).. TS(k+1,l,t) =g= TE(k,l,t);
endsame(k,l,t)$(kl(k,l)).. TE(k,l,t)   =e= TS(k,l,t) + SL(k,l,t);
starttask1(k,l,t)$(kl(k,l)).. TS(k,l,t)   =e= sum(i$il1(i,l), TTS(i,k,l,t));
start2diff('1','2',l,ll,t)$(il1('2',l)and ill1('1',ll)).. sum(k$kl(k,l), TTS('2',k,l,t)) =g=
sum(kk$kkll(kk,ll), TTS('1',kk,ll,t)) + dt('1','2')*(NC('2',l,t) + NC('1',ll,t)-1);
```

```
start3diff('2','3',l,ll,t)$(il1('3',l)and ill1('2',ll)).. sum(k$kl(k,l), TTS('3',k,l,t)) =g=
sum(kk$kkll(kk,ll), TTS('2',kk,ll,t)) + dt('2','3')*(NC('3',l,t) + NC('2',ll,t)-1);
start33diff('4','3',l,ll,t)$(il1('3',l)and ill1('4',ll)).. sum(k$kl(k,l), TTS('3',k,l,t)) =g=
sum(kk$kkll(kk,ll), TTS('4',kk,ll,t)) + dt('4','3')*(NC('3',l,t) + NC('4',ll,t)-1);
start4diff('4','5',l,ll,t)$(il1('5',l)and ill1('4',ll)).. sum(k$kl(k,l), TTS('5',k,l,t)) =g=
sum(kk$kkll(kk,ll), TTS('4',kk,ll,t)) + dt('4','5')*(NC('5',l,t) + NC('4',ll,t)-1);
start5diff('1','4',l,ll,t)$(il1('4',l)and ill1('1',ll)).. sum(k$kl(k,l), TTS('4',k,l,t)) =g=
sum(kk$kkll(kk,ll), TTS('1',kk,ll,t)) + dt('1','4')*(NC('4',l,t) + NC('1',ll,t)-1);
start6diff('6','7',l,ll,t)$(il1('7',l)and ill1('6',ll)).. sum(k$kl(k,l), TTS('7',k,l,t)) =g=
sum(kk$kkll(kk,ll), TTS('6',kk,ll,t)) + dt('6','7')*(NC('7',l,t) + NC('6',ll,t)-1);
start7diff('7','8',l,ll,t)$(il1('8',l)and ill1('7',ll)).. sum(k$kl(k,l), TTS('8',k,l,t)) =g=
sum(kk$kkll(kk,ll), TTS('7',kk,ll,t)) + dt('7','8')*(NC('8',l,t) + NC('7',ll,t)-1);
end2diff('1','2',l,ll,t)$(il1('2',l)and ill1('1',ll)).. sum(k$kl(k,l), TTS('2',k,l,t)+
CL('2',k,l,t)) =g= sum(kk$kkll(kk,ll), TTS('1',kk,ll,t)+ CL('1',kk,ll,t)) +
dt('1','2')*(NC('2',l,t) + NC('1',ll,t)-1);
end3diff('2','3',l,ll,t)$(il1('3',l)and ill1('2',ll)).. sum(k$kl(k,l), TTS('3',k,l,t)+
CL('3',k,l,t)) =g= sum(kk$kkll(kk,ll), TTS('2',kk,ll,t)+ CL('2',kk,ll,t)) +
dt('2','3')*(NC('3',l,t) + NC('2',ll,t)-1);
end33diff('4','3',l,ll,t)$(il1('3',l)and ill1('4',ll)).. sum(k$kl(k,l), TTS('3',k,l,t)+
CL('3',k,l,t)) =g= sum(kk$kkll(kk,ll), TTS('4',kk,ll,t)+ CL('4',kk,ll,t)) +
dt('4','3')*(NC('3',l,t) + NC('4',ll,t)-1);
end4diff('4','5',l,ll,t)$(il1('5',l)and ill1('4',ll)).. sum(k$kl(k,l), TTS('5',k,l,t)+
CL('5',k,l,t)) =g= sum(kk$kkll(kk,ll), TTS('4',kk,ll,t)+ CL('4',kk,ll,t)) +
dt('4','5')*(NC('5',l,t) + NC('4',ll,t)-1);
end5diff('1','4',l,ll,t)$(il1('4',l)and ill1('1',ll)).. sum(k$kl(k,l), TTS('4',k,l,t)+
CL('4',k,l,t)) =g= sum(kk$kkll(kk,ll), TTS('1',kk,ll,t)+ CL('1',kk,ll,t)) +
dt('1','4')*(NC('4',l,t) + NC('1',ll,t)-1);
end6diff('6','7',l,ll,t)$(il1('7',l)and ill1('6',ll)).. sum(k$kl(k,l), TTS('7',k,l,t)+
CL('7',k,l,t)) =g= sum(kk$kkll(kk,ll), TTS('6',kk,ll,t)+ CL('6',kk,ll,t)) +
dt('6','7')*(NC('7',l,t) + NC('6',ll,t)-1);
end7diff('7','8',l,ll,t)$(il1('8',l)and ill1('7',ll)).. sum(k$kl(k,l), TTS('8',k,l,t)+
CL('8',k,l,t)) =g= sum(kk$kkll(kk,ll), TTS('7',kk,ll,t)+ CL('7',kk,ll,t)) +
dt('7','8')*(NC('8',l,t) + NC('7',ll,t)-1);
inventory(m,t).. INV(m,t) =e= (I0(m))$(ord(t)=1)+(INV(m,t-1))$(ord(t)>1)+ sum(l,
PQ(m,l,t))-SQ(m,t);
backlog(m,t).. Ib(m,t) =g=  Ib0(m)$(ord(t)=1)+Ib(m,t-1)$(ord(t)>1)+ D(m,t)-SQ(m,t);
supply(m,t).. sum(tt$(ord(tt)<= ord(t)), SQ(m,tt)) =l= sum(tt, D(m,tt))+Ib0(m);
dip(m,t).. Id(m,t) =g= IL(m)- INV(m,t);

CL.lo(i,k,l,t) = 0;
PT.lo(i,l,t) = 0;
SL.lo(k,l,t) = 0;
TS.lo(k,l,t) = 0;
TE.lo(k,l,t) = 0;
TTS.lo(i,k,l,t) = 0;
INV.lo(m,t) = 0;
Ib.lo(m,t) = 0;
Id.lo(m,t) = 0;
```

```
NC.lo(i,l,t) = 0;
YS.lo(i,l,t) = 0;
Z.lo(i,k,l,t) = 0;

CL.up(i,k,l,t) = H(l,t);
PT.up(i,l,t) = H(l,t);
SL.up(k,l,t) = H(l,t);
TS.up(k,l,t) = H(l,t);
TE.up(k,l,t) = H(l,t);
TTS.up(i,k,l,t) = H(l,t);
INV.up(m,t) = ST(m);
SQ.up(m,t) = sum(tt$(ord(tt) <= ord(t)), D(m,tt))+Ib0(m);
Ib.up(m,t) = sum(tt$(ord(tt) <= ord(t)), D(m,tt))+Ib0(m);
Id.up(m,t) = IL(m);
NC.up(i,l,t) = 1;
YS.up(i,l,t) = 1;
Z.up(i,k,l,t) = 1;

OPTION
SOLPRINT = OFF
limrow = 40
optcr = 0
reslim = 10000
mip = cplex ;

MODEL       planning /all/;

SOLVE       planning using mip maximizing GP;

DISPLAY     GP.l,CL.l,PT.l,TS.l,TE.l,TTS.l,Y.l,YS.l,INV.l,PQ.l,Ib.l,Id.l,SQ.l;
```

A.2 DATA files for Examples 1-3

A.2.1 Example 1

A.2.1.1 Example 1a

```
SETS
m materials /1*9/
i tasks /1*6/
k slots /1*2/
l lines /1*3/
t time periods /1*4/
il1(i,l) suitability of tasks to lines /(1,4).1,(2,5).2,(3,6).3/
kl(k,l)  slots on lines /(1,2).1,(1,2).2,(1,2).3/
ALIAS   (i,ii),(k,kk),(l,ll),(t,tt);
SETS
```

```
iil1(ii,l)   suitability of tasks to lines /(1,4).1,(2,5).2,(3,6).3/
ill1(i,ll)   suitability of tasks to lines /(1,4).1,(2,5).2,(3,6).3/
iill1(ii,ll) suitability of tasks to lines /(1,4).1,(2,5).2,(3,6).3/
kkl(kk,l)    slots on lines /(1,2).1,(1,2).2,(1,2).3/
kll(k,ll)    slots on lines /(1,2).1,(1,2).2,(1,2).3/
kkll(kk,ll)  slots on lines /(1,2).1,(1,2).2,(1,2).3/

PARAMETERS
I0(m) initial inventory of m
/1 1000000,2 0,3 0,4 0,5 1000000,6 0,7 0,8 0,9 0/
IL(m) target level of m
/1 0,2 0,3 0,4 1500,5 0,6 0,7 0,8 1000,9 0/
ST(m) storage limit of m
/1 1000000,2 3500,3 3000,4 1000000,5 1000000,6 4000,7 5000,8 1000000,9 3000/
g(m)  revenue per unit of m
/1 0,2 0,3 0,4 1.2,5 0,6 0,7 0,8 0.8,9 0/
period(t) time length of t in hrs
/1 360,2 360,3 720,4 720/
nk(l) total number of slots on l
/1 2,2 2,3 2/
meu(i) primary material of i
/1*5 1,6 0.9/ ;

TABLE
dt(i,ii)  batch processing time of i on l
      1*6
1*6  0   ;
TABLE
ct(i,l)  changeover time of i on l
     1    2    3
1    2
2         2
3              3
4    3
5         3
6              2    ;
TABLE
sigma(m,i) mass balance coefficient of m to i
       1     2     3    4    5    6
1     -1     0     0    0    0    0
2      1    -1     0    0    0    0
3      0     1   -0.8   0    0    0
4      0     0     1    0    0    0
5      0     0     0   -1    0    0
6      0     0     0    1   -1    0
7      0     0   -0.2   0    1   -1
8      0     0     0    0    0    0.9
9      0     0     0    0    0    0.1   ;
```

```
TABLE
D(m,t) demand for m in t
        1       2       3       4
4    2000   1500   6000   5000
8    1000   1500   6000   2500 ;
TABLE
H(l,t) uptime of l in t(in hrs)
     1      2      3      4
1   360  360  720  720
2   360  360  720  720
3   360  360  720  720  ;
TABLE
RU(i,l) max rate of production for i on l

     1    2    3
1    20
2         15
3              25
4    15
5         10
6              10  ;
TABLE
MCL(i,l) max rate of production for i on l

     1    2    3
1    100
2         110
3              140
4    110
5         120
6              130 ;
TABLE
cc(i,l) changeover cost of i on l
      1    2    3
1    40
2         50
3              70
4    50
5         60
6              65    ;
TABLE
hc(m,t)  holding cost of m per 1000 per day
      1*4
1    0
2    1.30
3    1.23
4    1.76
5    0
```

```
6    1.60
7    1.40
8    1.82
9    1.50;

PARAMETER
RL(i,l) min rate of production for i on l;
RL(i,l) = RU(i,l)/4;
PARAMETER
ho1(m,t)   holding cost of m during t+1;
ho1(m,t)  = hc(m,t)* period(t+1)/(1000*24);
PARAMETER
ho2(m,t)   holding cost of m during t;
ho2(m,t)  = hc(m,t)* period(t)/(1000*24);
PARAMETER
a(m,t) penalty cost of m for dipping below IL(m) in t ;
a(m,t)= 2*ho1(m,t) ;
```

A.2.1.2 Example 1b

```
SETS
m materials /1*12/
i tasks /1*8/
k slots /1*3/
l lines /1*3/
t time periods /1*4/
il1(i,l) suitability of tasks to lines /(1,4).1,(2,5,7).2,(3,6,8).3/
kl(k,l)  slots on lines /(1,2).1,(1,2,3).2,(1,2,3).3/
ALIAS   (i,ii),(k,kk),(l,ll),(t,tt);
SETS
iil1(ii,l)   suitability of tasks to lines /(1,4).1,(2,5,7).2,(3,6,8).3/
ill1(i,ll)   suitability of tasks to lines /(1,4).1,(2,5,7).2,(3,6,8).3/
iill1(ii,ll) suitability of tasks to lines /(1,4).1,(2,5,7).2,(3,6,8).3/
kkl(kk,l)    slots on lines /(1,2).1,(1,2,3).2,(1,2,3).3/
kll(k,ll)    slots on lines /(1,2).1,(1,2,3).2,(1,2,3).3/
kkll(kk,ll)  slots on lines /(1,2).1,(1,2,3).2,(1,2,3).3/

PARAMETERS
I0(m) initial inventory of m
/1 1000000,2 0,3 0,4 0,5 1000000,6 0,7 0,8 0,9 0,10 1000000,11 0,12 0/
IL(m) target level of m
/1 0,2 0,3 0,4 1500,5 0,6 0,7 0,8 1000,9 0,10 0,11 0,12 1000/
ST(m) storage limit of m
/1 1000000,2 3500,3 3000,4 1000000,5 1000000,6 4000,7 5000,8 1000000,9 3000,10
1000000,11 3000,12 1000000/
g(m)  revenue per unit of m
/1 0,2 0,3 0,4 1.2,5 0,6 0,7 0,8 0.8,9 0,10 0,11 0,12 2.3/
period(t) time length of t in hrs
```

```
/1 360,2 360,3 720,4 720/
nk(l) total number of slots on l
/1 2,2 3,3 3/
meu(i) primary material of i
/1*5 1,6 0.9,7*8 1/  ;

TABLE
dt(i,ii)  batch processing time of i on l
      1*8
1*8  0   ;
TABLE
ct(i,l)  changeover time of i on l
     1    2    3
1    2
2         2
3              3
4    3
5         3
6              2
7         2
8              3   ;
TABLE
SUT(i,l)  changeover time of i on l
    1    2    3
7         15
8              10   ;
TABLE
sigma(m,i) mass balance coefficient of m to i
       1     2     3    4     5     6     7      8
1     -1     0     0    0     0     0     0      0
2      1    -1     0    0     0     0    -0.25   0
3      0     1    -0.8  0     0     0     0      0
4      0     0     1    0     0     0     0      0
5      0     0     0   -1     0     0     0      0
6      0     0     0    1    -1     0     0      0
7      0     0    -0.2  0     1    -1     0      0
8      0     0     0    0     0     0.9   0      0
9      0     0     0    0     0     0.1   0      0
10     0     0     0    0     0     0    -0.75   0
11     0     0     0    0     0     0     1     -1
12     0     0     0    0     0     0     0      1  ;
TABLE
D(m,t) demand for m in t
      1        2        3        4
4     2000   1500   6000   5000
8     1000   1500   6000   2500
12    1000   1000   1200   1500   ;
TABLE
```

```
H(l,t) uptime of l in t(in hrs)
     1     2     3     4
1   360  360  720  720
2   360  360  720  720
3   360  360  720  720  ;
TABLE
RU(i,l) max rate of production for i on l
     1    2    3
1    20
2         15
3              25
4    15
5         10
6              10
7         20
8              25;
TABLE
MCL(i,l) max rate of production for i on l
     1    2    3
1    100
2         110
3              140
4    110
5         120
6              130
7         100
8              125  ;
TABLE
cc(i,l) changeover cost of i on l
     1    2    3
1    40
2         50
3              70
4    50
5         60
6              65
7         55
8              80  ;
TABLE
hc(m,t)  holding cost of m per 1000 per day
     1*4
2    1.30
3    1.23
4    1.76
6    1.60
7    1.40
8    1.82
9    1.50
```

```
11   1.28
12   1.90  ;

PARAMETER
RL(i,l) min rate of production for i on l;
RL(i,l) = RU(i,l)/4;
PARAMETER
ho1(m,t)   holding cost of m during t+1;
ho1(m,t)  = hc(m,t)* period(t+1)/(1000*24);
PARAMETER
ho2(m,t)   holding cost of m during t;
ho2(m,t)  = hc(m,t)* period(t)/(1000*24);
PARAMETER
a(m,t) penalty cost of m for dipping below IL(m) in t ;
a(m,t)= 2*ho1(m,t) ;
```

A.2.2 Example 2

```
SETS
m materials /1*7/
i tasks /1*5/
k slots /1*2/
l lines /1*4/
t time periods /1*4/
il1(i,l) suitability of tasks to lines /1.1,(2,4).(2,3),(3,5).4/
kl(k,l)  slots on lines /1.1,(1,2).(2,3),(1,2).4/
ALIAS   (i,ii),(k,kk),(l,ll),(t,tt);
SETS
iil1(ii,l)   suitability of tasks to lines /1.1,(2,4).(2,3),(3,5).4/
ill1(i,ll)   suitability of tasks to lines /1.1,(2,4).(2,3),(3,5).4/
iill1(ii,ll) suitability of tasks to lines /1.1,(2,4).(2,3),(3,5).4/
kkl(kk,l)    slots on lines /1.1,(1,2).(2,3),(1,2).4/
kll(k,ll)    slots on lines /1.1,(1,2).(2,3),(1,2).4/
kkll(kk,ll)  slots on lines /1.1,(1,2).(2,3),(1,2).4/

PARAMETERS
I0(m) initial inventory of m
/1 1000000,2 0,3 0,4 0,5 0,6 0,7 0/
IL(m) target level of m
/1 0,2 0,3 0,4 1500,5 0,6 1000,7 0/
ST(m) storage limit of m
/1 1000000,2 3500,3 3000,4 1000000,5 4000,6 1000000,7 3000/
out(m) outsourcing amount of m
/1*4 0,5 0,6*7 0/
*/1*2 0,3 1000000,4*7 0/
*/1*4 0,5 1000000,6*7 0/
*/1*2 0,3 1000000,4 0,5 1000000,6*7 0/
*/1*2 0,3 1000000,4 0,5 1000,6*7 0/
```

```
rc(m) purchase cost per unit of m
/1*4 0,5 0,6*7 0/
g(m)  revenue per unit of m
/1 0,2 0,3 0,4 1.2,5 0,6 0.8,7 0/
period(t) time length of t in hrs
/1 360,2 360,3 720,4 720/
nk(l) total number of slots on l
/1 1,2*4 2/
meu(i) primary material of i
/1*4 1,5 0.9/ ;

TABLE
dt(i,ii)  batch processing time of i on l
    1    2    3    4    5
1        4         4
2             2
3
4             3         3
5                          ;
TABLE
ct(i,l)  changeover time of i on l
    1    2    3    4
1
2        2    1.5
3                  3
4        2    2
5                  3   ;
TABLE
sigma(m,i) mass balance coefficient of m to i
       1     2     3    4    5
1     -1     0     0    0    0
2      1    -1     0   -1    0
3      0     1    -0.8  0    0
4      0     0     1    0    0
5      0     0    -0.2  1   -1
6      0     0     0    0    0.9
7      0     0     0    0    0.1   ;
TABLE
D(m,t) demand for m in t
      1        2        3        4
4    1000   1500   6000   2500
6    2000   1500   6000   5000 ;
TABLE
H(l,t) uptime of l in t(in hrs)
    1      2      3      4
1   360  360  720  720
2   360  360  720  720
3   360  360  720  720
```

```
4   360  360  720  720 ;
TABLE
RU(i,l) max rate of production for i on l
    1    2    3    4
1    20
2         8    7
3                   15
4         5    5
5                   20        ;
TABLE
cc(i,l) changeover cost of i on l
    1    2     3    4
1
2         50   45
3                     70
4         40   40
5                     80        ;
TABLE
hc(m,t)  holding cost of m per 1000 per day
     1*4
2    1.30
3    1.23
4    1.76
5    1.36
6    1.60
7    1.40 ;

PARAMETER
RL(i,l) min rate of production for i on l;
RL(i,l) = RU(i,l)/4;
PARAMETER
ho1(m,t)   holding cost of m during t+1;
ho1(m,t)  = hc(m,t)* period(t+1)/(1000*24);
PARAMETER
ho2(m,t)   holding cost of m during t;
ho2(m,t)  = hc(m,t)* period(t)/(1000*24);
PARAMETER
a(m,t) penalty cost of m for dipping below IL(m) in t ;
a(m,t)= 2*ho1(m,t) ;
SCALAR        MCL  minimum campaign length in hrs /100/ ;
```

A.2.3 Example 3

```
SETS
m materials /1*11/
i tasks /1*8/
k slots /1*3/
l lines /1*4/
```

```
t time periods /1*3/
il1(i,l) suitability of tasks to lines /(1,6).1,(2,4,7).(2,3),(3,5,8).4/
kl(k,l)  slots on lines  /(1,2).1,(1,2,3).(2,3,4)/
ALIAS (i,ii),(k,kk),(l,ll),(t,tt);
SETS
iil1(ii,l)   suitability of tasks to lines /(1,6).1,(2,4,7).(2,3),(3,5,8).4/
ill1(i,ll)   suitability of tasks to lines /(1,6).1,(2,4,7).(2,3),(3,5,8).4/
iill1(ii,ll) suitability of tasks to lines /(1,6).1,(2,4,7).(2,3),(3,5,8).4/
kkl(kk,l)    slots on lines  /(1,2).1,(1,2,3).(2,3,4)/
kll(k,ll)    slots on lines  /(1,2).1,(1,2,3).(2,3,4)/
kkll(kk,ll)  slots on lines  /(1,2).1,(1,2,3).(2,3,4)/

PARAMETERS
I0(m) initial inventory of m
/1 994876.778,2 1276.389,3 233.5,4*5 0,6 352,7 261.333,8 1000000,9*11 0/
IL(m) target level of m
/1 0,2 0,3 0,4 1500,5 0,6 1000,7 0,8*10 0,11 1000/
ST(m) storage limit of m
/1 1000000,2 3500,3 3000,4 1000000,5 4000,6 1000000,7 3000,8 1000000,9 3500,10
4000,11 1000000/
g(m)  revenue per unit of m
/1 0,2 0,3 0,4 1.2,5 0,6 0.8,7 0,8*10 0,11 2.3/
period(t) time length of t in hrs
/1 360,2 720,3 720/
nk(l) total number of slots on l
/1 2,2*4 3/
Ib0(m) initial backlog of m
/1*11 0/
meu(i) primary material of i
/1*4 1,5 0.9,6*8 1/ ;

TABLE
dt(i,ii)  batch processing time of i on l
    1   2   3   4   5   6   7   8
1       4       4
2           2
3
4           3       3
5
6                       4
7                           3
8                              ;
TABLE
ct(i,l)  changeover time of i on l
    1   2   3   4
1   3
2       2   1.5
3               3
```

```
4         2    2
5                  3
6    3
7         2    1.5
8                  3   ;
TABLE
SUT(i,l)  changeover time of i on l
     1    2    3    4
6    10
7         12   12
8                   15   ;
TABLE
sigma(m,i) mass balance coefficient of m to i
       1     2     3     4    5     6     7     8
1     -1     0     0     0    0     0     0     0
2      1    -1     0    -1    0     0     0     0
3      0     1    -0.8   0    0     0     0     0
4      0     0     1     0    0     0     0     0
5      0     0    -0.2   1   -1     0     0     0
6      0     0     0     0   0.9    0     0     0
7      0     0     0     0   0.1    0     0     0
8      0     0     0     0    0    -1     0     0
9      0     0     0     0    0     1    -1     0
10     0     0     0     0    0     0     1    -1
11     0     0     0     0    0     0     0     1    ;
TABLE
D(m,t) demand for m in t
    1        2        3
4   1500   6000   2500
6   1500   6000   5000
11  1500   3500   4500  ;
TABLE
H(l,t) uptime of l in t(in hrs)
   1      2      3
1  360  720  600
2  360  720  720
3  360  720  720
4  360  720  720  ;
TABLE
RU(i,l) max rate of production for i on l
     1    2    3    4
1    20
2         16   7
3                   15
4         10   5
5                   20
6    20
7         16   7
```

```
8                  15   ;
TABLE
cc(i,l) changeover cost of i on l
      1    2    3   4
1    45
2         50   45
3                   70
4         40   40
5                   80
6    50
7         45   50
8                   90    ;
TABLE
CL0(i,l) initial campaign length of i in final slot of l in t
       1*4
1*8  0    ;
TABLE
YS0(i,l) initial spillover of i in final slot of l in t
       1*4
1*8  0    ;
TABLE
hc(m,t)  holding cost of m per 1000 per day
      1*3
2    1.30
3    1.23
4    1.76
5    1.36
6    1.60
7    1.40
9    1.35
10   1.28
11   1.80    ;

PARAMETER
RL(i,l) min rate of production for i on l;
RL(i,l) = RU(i,l)/4;
PARAMETER
ho1(m,t)   holding cost of m during t+1;
ho1(m,t)  = hc(m,t)* period(t+1)/(1000*24);
PARAMETER
ho2(m,t)   holding cost of m during t;
ho2(m,t)  = hc(m,t)* period(t)/(1000*24);
PARAMETER
a(m,t) penalty cost of m for dipping below IL(m) in t ;
a(m,t)= 2*ho1(m,t) ;
SCALAR
MCL  minimum campaign length in hrs /100/  ;
```

APPENDIX B: Files for Chapter 6

B.1 GAMS files for Examples 1-3

B.1.1 Profit Maximization

```
VARIABLES
P               profit
DB(i,j,k)       amount that i consumed on j at Tk
b(i,j,k)        residual amount of task i on j at Tk
BE(i,j,k)       amount of batch discharged by task i on j at Tk
SL(k)           slot length
T(j,k)          remaining time on j at Tk
INV(m,k)        inventory of m at Tk
Y(i,j,k)        task assignment variable 1
Z(j,k)          task assignment variable 2
YR(i,j,k)       residual binary
YE(i,j,k)       ending binary      ;
POSITIVE VARIABLE        DB,b,BE,SL,T,INV,Z,YR,YE ;
BINARY VARIABLE          Y ;

EQUATIONS
profit                  profit to be maximized
allocate1(j,k)          allocation constraint 1
endj2(j,k)              ending time constraint 2
endj3(j,k)              ending time constraint 3
inventory(m,k)          inventory constraint 2
residual1(i,j,k)        residual 1
residual22(j,k)         residual 22
material1(i,j,k)        material constraint 1
material2(i,j,k)        material constraint 2
material3(i,j,k)        material constraint 3
material4(i,j,k)        material constraint 2
sumslot                 sum of slot lengths     ;

profit.. P =e= sum((m,k)$(ord(k)=NK), g(m)*INV(m,k))  ;
allocate1(j,k)$(ord(k) < NK).. sum(i$(ij1(i,j)), Y(i,j,k)) =e= Z(j,k);
endj2(j,k)$(ord(k) > 1 and ord(k) < NK ).. T(j,k) =l= sum(i$(ij1(i,j)),alpha(i,j)*
YR(i,j,k)) + sum(i$(ij1(i,j)), beta(i,j)*b(i,j,k));
endj3(j,k) $(ord(k) < NK).. T(j,k+1) =g= T(j,k)+
sum(i$(ij1(i,j)),alpha(i,j)*Y(i,j,k)+beta(i,j)*DB(i,j,k))-SL(k+1);
inventory(m,k).. INV(m,k)  =e= (INV(m,k-1)$(ord(k)>1)+I0(m)$(ord(k)=1))+
sum(i$(OI(m,i) and ord(i) < NI), (sigma(m,i)/meu(i))*
sum(j$ij1(i,j),BE(i,j,k)))+sum(i$(II(m,i) and ord(i) < NI), (sigma(m,i)/meu(i))*
sum(j$ij1(i,j),DB(i,j,k)));
residual1(i,j,k)$(ij1(i,j) and ord(k) > 1 and ord(k) < NK).. YR(i,j,k) =e= YR(i,j,k-1) +
Y(i,j,k-1) - YE(i,j,k);
```

```
residual22(j,k)$(ord(k) > 1 and ord(k) < NK).. sum(i$ij1(i,j), YE(i,j,k)) =e= Z(j,k);
material1(i,j,k)$(ij1(i,j) and ord(i) < NI and ord(k) gt 1).. b(i,j,k) =e= b(i,j,k-1) +
DB(i,j,k-1) -  BE(i,j,k);
material2(i,j,k) $( ij1(i,j) and ord(i) < NI and ord(k) gt 1 and ord(k) lt NK).. BE(i,j,k)
=l= bmax(j)* YE(i,j,k);
material3(i,j,k) $( ij1(i,j) and  ord(i) < NI and ord(k) lt NK).. DB(i,j,k) =l=
bmax(j)*Y(i,j,k);
material4(i,j,k)$(ij1(i,j) and ord(i) < NI and ord(k) > 1 and ord(k) < NK).. b(i,j,k) =l=
bmax(j)* YR(i,j,k);
sumslot.. sum(k, SL(k)) =l= H;

DB.lo(i,j,k) = 0;
DB.up(i,j,k) = bmax(j);
DB.fx(i,j,klast) = 0;
BE.lo(i,j,k) = 0;
BE.up(i,j,k) = bmax(j);
BE.fx(izero,j,k) = 0;
BE.fx(i,j,kzero) = 0;
b.lo(i,j,k) = 0;
b.up(i,j,k) = bmax(j);
b.fx(i,j,kzero) = 0;
b.fx(i,j,klast) = 0;
b.fx(izero,j,k) = 0;
SL.lo(k) = 0;
SL.up(k) = smax((i,j)$(ij1(i,j)and ord(i) < NI), alpha(i,j)+beta(i,j)*bmax(j));
T.lo(j,k) = 0;
T.up(j,k) = smax(i$(ij1(i,j)and ord(i) < NI), alpha(i,j)+beta(i,j)*bmax(j));
T.fx(j,klast) = 0;
T.fx(j,kzero) = 0;
INV.lo(m,k) = 0;
INV.up(m,k) = IL(m);
Z.lo(j,k) = 0;
Z.up(j,k) = 1;
z.fx(j,klast) = 1;
YR.lo(i,j,k) = 0;
YR.up(i,j,k) = 1;
YR.fx(i,j,kzero) = 0;
YR.fx(i,j,klast) = 0;
YE.lo(i,j,k) = 0;
YE.up(i,j,k) = 1;
YE.fx(i,j,kzero) = 0;

OPTION
SOLPRINT = OFF
limrow = 20
limcol = 10
optcr = 0
reslim = 10000
```

```
mip = cplex ;

MODEL      planning /all/ ;

SOLVE      planning using mip maximizing P ;

DISPLAY    P.l,SL.l,T.l,Y.l,Z.l,YR.l, YE.l,DB.l,b.l,BE.l,INV.l ;
```

B.1.2 Makespan Minimization

```
VARIABLES
MS          makespan
DB(i,j,k)   amount that i consumed on j at Tk
b(i,j,k)    residual amount of task i on j at Tk
BE(i,j,k)   amount of batch discharged by task i on j at Tk
SL(k)       slot length
T(j,k)      remaining time on j at Tk
INV(m,k)    inventory of m at Tk
Y(i,j,k)    task assignment variable 1
Z(j,k)      task assignment variable 2
YR(i,j,k)   residual binary
YE(i,j,k)   ending binary    ;
POSITIVE VARIABLE      DB,b,BE,SL,T,INV,Z,YR,YE ;
BINARY VARIABLE        Y ;

EQUATIONS
makespan             profit to be minimized
allocate1(j,k)       allocation constraint 1
endj2(j,k)           ending time constraint 2
endj3(j,k)           ending time constraint 3
inventory(m,k)       inventory constraint 2
residual1(i,j,k)     residual 1
residual22(j,k)      residual 22
material1(i,j,k)     material constraint 1
material2(i,j,k)     material constraint 2
material3(i,j,k)     material constraint 3
material4(i,j,k)     material constraint 2
demand(m,k)          demand constraint       ;

makespan.. MS =e= sum(k, SL(k))  ;
allocate1(j,k)$(ord(k) < NK).. sum(i$(ij1(i,j)), Y(i,j,k)) =e= Z(j,k);
endj2(j,k)$(ord(k) > 1 and ord(k) < NK ).. T(j,k) =l= sum(i$(ij1(i,j)),alpha(i,j)*
YR(i,j,k)) + sum(i$(ij1(i,j)), beta(i,j)*b(i,j,k));
endj3(j,k) $(ord(k) < NK).. T(j,k+1) =g= T(j,k)+
sum(i$(ij1(i,j)),alpha(i,j)*Y(i,j,k)+beta(i,j)*DB(i,j,k))-SL(k+1);
inventory(m,k).. INV(m,k)  =e= (INV(m,k-1)$(ord(k)>1)+I0(m)$(ord(k)=1))+
sum(i$(OI(m,i) and ord(i) < NI), (sigma(m,i)/meu(i))*
```

```
sum(j$ij1(i,j),BE(i,j,k)))+sum(i$(II(m,i) and ord(i) < NI), (sigma(m,i)/meu(i))*
sum(j$ij1(i,j),DB(i,j,k)));
residual1(i,j,k)$(ij1(i,j) and ord(k) > 1 and ord(k) < NK).. YR(i,j,k) =e= YR(i,j,k-1) +
Y(i,j,k-1) - YE(i,j,k);
residual22(j,k)$(ord(k) > 1 and ord(k) < NK).. sum(i$ij1(i,j), YE(i,j,k)) =e= Z(j,k);
material1(i,j,k)$(ij1(i,j) and ord(i) < NI and ord(k) gt 1).. b(i,j,k) =e= b(i,j,k-1) +
DB(i,j,k-1) -  BE(i,j,k);
material2(i,j,k) $( ij1(i,j) and ord(i) < NI and ord(k) gt 1 and ord(k) lt NK).. BE(i,j,k)
=l= bmax(j)* YE(i,j,k);
material3(i,j,k) $( ij1(i,j) and  ord(i) < NI and ord(k) lt NK).. DB(i,j,k) =l=
bmax(j)*Y(i,j,k);
material4(i,j,k)$(ij1(i,j) and ord(i) < NI and ord(k) > 1 and ord(k) < NK).. b(i,j,k) =l=
bmax(j)* YR(i,j,k);
demand(m,k)$(ord(k)=NK).. INV(m,k) =g= dem(m);

DB.lo(i,j,k) = 0;
DB.up(i,j,k) = bmax(j);
DB.fx(i,j,klast) = 0;
BE.lo(i,j,k) = 0;
BE.up(i,j,k) = bmax(j);
BE.fx(izero,j,k) = 0;
BE.fx(i,j,kzero) = 0;
b.lo(i,j,k) = 0;
b.up(i,j,k) = bmax(j);
b.fx(i,j,kzero) = 0;
b.fx(i,j,klast) = 0;
b.fx(izero,j,k) = 0;
SL.lo(k) = 0;
SL.up(k) = smax((i,j)$(ij1(i,j)and ord(i) < NI), alpha(i,j)+beta(i,j)*bmax(j));
T.lo(j,k) = 0;
T.up(j,k) = smax(i$(ij1(i,j)and ord(i) < NI), alpha(i,j)+beta(i,j)*bmax(j));
T.fx(j,klast) = 0;
T.fx(j,kzero) = 0;
INV.lo(m,k) = 0;
INV.up(m,k) = IL(m);
Z.lo(j,k) = 0;
Z.up(j,k) = 1;
z.fx(j,klast) = 1;
YR.lo(i,j,k) = 0;
YR.up(i,j,k) = 1;
YR.fx(i,j,kzero) = 0;
YR.fx(i,j,klast) = 0;
YE.lo(i,j,k) = 0;
YE.up(i,j,k) = 1;
YE.fx(i,j,kzero) = 0;

OPTION
SOLPRINT = OFF
```

```
limrow = 20
limcol = 10
optcr = 0
reslim = 200000
mip = cplex ;

MODEL       planning /all/ ;

SOLVE       planning using mip minimizing MS ;

DISPLAY     MS.l,SL.l,T.l,Y.l,Z.l,YR.l, YE.l,DB.l,b.l,BE.l,INV.l ;
```

B.1.3 Constant Batch Processing Times

```
VARIABLES
P              profit
DB(i,j,k)      amount that i consumed on j at k
b(i,j,k)       residual amount of task i on j at k
BE(i,j,k)      amount of batch discharged by task i on j at k
SL(k)          slot length
T(j,k)         remaining time on j in k
INV(m,k)       inventory of m at k
Y(i,j,k)       task assignment variable 1
Z(j,k)         task assignment variable 2
YR(i,j,k)      residual binary
YE(i,j,k)      ending binary      ;
POSITIVE VARIABLE       DB,b,BE,SL,T,INV,Z,YR,YE ;
BINARY VARIABLE         Y ;

EQUATIONS
profit               profit to be maximized
allocate1(j,k)       allocation constraint 1
endj2(j,k)           ending time constraint 2
endj3(j,k)           ending time constraint 3
inventory(m,k)       inventory constraint 2
residual1(i,j,k)     residual 1
residual22(j,k)      residual 22
material1(i,j,k)     material constraint 1
material2(i,j,k)     material constraint 2
material3(i,j,k)     material constraint 3
material4(i,j,k)     material constraint 2
sumslot              sum of slot lengths    ;

profit.. P =e= sum((m,k)$(ord(k)=NK), g(m)*INV(m,k))  ;
allocate1(j,k)$(ord(k) < NK).. sum(i$(ij1(i,j)), Y(i,j,k)) =e= Z(j,k);
endj2(j,k)$(ord(k) > 1 and ord(k) < NK ).. T(j,k) =l= sum(i$(ij1(i,j)),tau(i,j)* YR(i,j,k))
;
```

```
endj3(j,k) $(ord(k) < NK).. T(j,k+1) =g= T(j,k)+ sum(i$(ij1(i,j)),tau(i,j)*Y(i,j,k))-
SL(k+1);
inventory(m,k).. INV(m,k)  =e= (INV(m,k-1)$(ord(k)>1)+I0(m)$(ord(k)=1))+
sum(i$(OI(m,i) and ord(i) < NI), (sigma(m,i)/meu(i))*
sum(j$ij1(i,j),BE(i,j,k)))+sum(i$(II(m,i) and ord(i) < NI), (sigma(m,i)/meu(i))*
sum(j$ij1(i,j),DB(i,j,k)));
residual1(i,j,k)$(ij1(i,j) and ord(k) > 1 and ord(k) < NK).. YR(i,j,k) =e= YR(i,j,k-1) +
Y(i,j,k-1) - YE(i,j,k);
residual22(j,k)$(ord(k) > 1 and ord(k) < NK).. sum(i$ij1(i,j), YE(i,j,k)) =e= Z(j,k);
material1(i,j,k)$(ij1(i,j) and ord(i) < NI and ord(k) gt 1).. b(i,j,k) =e= b(i,j,k-1) +
DB(i,j,k-1) -  BE(i,j,k);
material2(i,j,k) $( ij1(i,j) and ord(i) < NI and ord(k) gt 1 and ord(k) lt NK).. BE(i,j,k)
=l= bmax(j)* YE(i,j,k);
material3(i,j,k) $( ij1(i,j) and  ord(i) < NI and ord(k) lt NK).. DB(i,j,k) =l=
bmax(j)*Y(i,j,k);
material4(i,j,k)$(ij1(i,j) and ord(i) < NI and ord(k) > 1 and ord(k) < NK).. b(i,j,k) =l=
bmax(j)* YR(i,j,k);
sumslot.. sum(k, SL(k)) =l= H;

DB.lo(i,j,k) = 0;
DB.up(i,j,k) = bmax(j);
DB.fx(i,j,klast) = 0;
BE.lo(i,j,k) = 0;
BE.up(i,j,k) = bmax(j);
BE.fx(izero,j,k) = 0;
BE.fx(i,j,kzero) = 0;
b.lo(i,j,k) = 0;
b.up(i,j,k) = bmax(j);
b.fx(i,j,kzero) = 0;
b.fx(i,j,klast) = 0;
b.fx(izero,j,k) = 0;
SL.lo(k) = 0;
SL.up(k) = smax((i,j)$(ij1(i,j)and ord(i) < NI), tau(i,j));
T.lo(j,k) = 0;
T.up(j,k) = smax(i$(ij1(i,j)and ord(i) < NI), tau(i,j));
T.fx(j,klast) = 0;
T.fx(j,kzero) = 0;
INV.lo(m,k) = 0;
INV.up(m,k) = IL(m);
Z.lo(j,k) = 0;
Z.up(j,k) = 1;
z.fx(j,klast) = 1;
YR.lo(i,j,k) = 0;
YR.up(i,j,k) = 1;
YR.fx(i,j,kzero) = 0;
YR.fx(i,j,klast) = 0;
YE.lo(i,j,k) = 0;
YE.up(i,j,k) = 1;
```

```
YE.fx(i,j,kzero) = 0;

OPTION
SOLPRINT = OFF
limrow = 20
limcol = 10
optcr = 0
reslim = 200000
mip = cplex ;

MODEL        planning /all/ ;

SOLVE        planning using mip maximizing P ;

DISPLAY      P.l,SL.l,T.l,Y.l,Z.l,YR.l, YE.l,DB.l,b.l,BE.l,INV.l ;
```

B.2 DATA files for Examples 1-3

B.2.1 Profit Maximization

```
* We provide the data file for Example 1 only. However, the data for Examples 2-3
can be inputted using the following file

SETS
m materials /1*4/
i tasks /1*4/
j units /1*5/
k slots /k0*k8/
ij1(i,j) suitability of tasks to units /1.(1,2), 2.3, 3.(4,5)/
OI(m,i)  tasks that produce materials /2.1, 3.2, 4.3/
II(m,i)  tasks that consume materials /1.1, 2.2, 3.3/   ;
ALIAS    (k,kk) ;
SCALAR
H    scheduling horizon in hrs /12/
NK   total number of k(including k0) per unit /9/
NI   total number of tasks /4/                 ;
SET kzero(k), klast(k),izero(i);
kzero(k) $(ord(k) eq 1) = yes;
klast(k) $(ord(k) eq card(k)) = yes;
izero(i) $(ord(i) eq ni) = yes;
ij1(izero,j) = yes;

PARAMETERS
I0(m) initial inventory of m
/1 1000000, 2*4 0/
IL(m) storage limit of m
/1 1000000, 2 200, 3 250, 4 1000000/
```

```
g(m)   revenue per unit of m
/1*3 0, 4 5/
bmax(j) max capacity of j
/1 100, 2 150, 3 200, 4*5 150/
meu(i) primary material of task i
/1*3 1/ ;

TABLE
sigma(m,i) mass balance coefficient of m to i
      1     2     3
1    -1
2     1    -1
3           1    -1
4                 1     ;
TABLE
alpha(i,j) constant production term
       1         2        3    4         5
1      1.333  1.333
2                         1
3                              0.667  0.667     ;
TABLE
beta(i,j) constant production term
      1            2           3        4            5
1      0.01333  0.01333
2                              0.005
3                                       0.00445  0.00445     ;
```

B.2.2 Makespan Minimization

```
* We provide the data file for Example 2 only. However, the data for Examples 1 & 3
can be inputted using the following file

SETS
m materials /1*9/
i tasks /1*6/
j units /1*4/
k slots /k0*k8/
ij1(i,j) suitability of tasks to units /1.1,(2,3,4).(2,3),5.4/
OI(m,i)  tasks that produce materials /4.1, 5.(3,5), 6.2, 7.4, 8.3, 9.5/
II(m,i)  tasks that consume materials /1.1, 2.2, 3.(2,4), 4.3, 5.4, 6.3, 7.5/  ;
ALIAS (k,kk) ;
SCALAR
NK   total number of k(including k0) per unit /9/
NI   total number of tasks /6/                 ;
SET kzero(k), klast(k),izero(i) ;
kzero(k) $(ord(k) eq 1) = yes;
klast(k) $(ord(k) eq card(k)) = yes;
```

```
izero(i) $(ord(i) eq ni) = yes;
ij1(izero,j) = yes;

PARAMETERS
I0(m) initial inventory of m
/1*3 1000000, 4*9 0/
IL(m) storage limit of m
/1*3 1000000, 4 100, 5 200, 6 150, 7 200, 8*9 1000000/
g(m)  revenue per unit of m
/1*7 0, 8*9 10/
dem(m) demand of m
/1*7 0, 8*9 200/
bmax(j) max capacity of j
/1 100, 2 50, 3 80, 4 200/
meu(i) primary material of task i
/1*5 1/       ;

TABLE
sigma(m,i) mass balance coefficient of m to i
       1    2    3     4     5
1    -1
2         -0.5
3         -0.5        -0.2
4     1        -0.4
5               0.6   -0.8   0.1
6          1   -0.6
7                     1     -1
8               0.4
9                            0.9    ;
TABLE
alpha(i,j) constant production term
       1      2        3      4
1     0.667
2            1.334  1.334
3            1.334  1.334
4            0.667  0.667
5                          1.3342      ;
TABLE
beta(i,j) constant production term
       1          2            3          4
1     0.00667
2                 0.02664   0.01665
3                0.02664   0.01665
4                0.01332   0.008325
5                                      0.00666         ;
```

B.2.3 Constant Batch Processing Times

* We provide the data file for Example 3 only. However, the data for Examples 1-2 can be inputted using the following file

```
SETS
m materials /1*13/
i tasks /1*8/
j units /1*6/
k slots /k0*k7/
ij1(i,j) suitability of tasks to units /(1,5).1, (2,3,7).(2,3), 4.4, 6.(5,6)/
OI(m,i)  tasks that produce materials /3.1, 4.(2,4), 5.3, 6.4, 7.4, 9.5, 10.7, 12.6, 13.7/
II(m,i)  tasks that consume materials /1.1, 2.2, 3.3, 4.3, 5.4, 6.5, 7.6, 8.5, 9.7, 10.6,
11.6/  ;
ALIAS (k,kk) ;
SCALAR
H    scheduling horizon in hrs /12/
NK   total number of k(including k0) per unit /8/
NI   total number of tasks /8/                  ;
SET kzero(k), klast(k),izero(i) ;
kzero(k) $(ord(k) eq 1) = yes;
klast(k) $(ord(k) eq card(k)) = yes;
izero(i) $(ord(i) eq ni) = yes;
ij1(izero,j) = yes;

PARAMETERS
I0(m) initial inventory of m
/1*2 1000000, 3*4 0,5 0,6*7 50, 8 1000000, 9*10 0, 11 1000000, 12*13 0/
IL(m) storage limit of m
/1*2 1000000, 3*4 100, 5 300, 6*7 150, 8 1000000, 9*10 150, 11*13 1000000/
g(m)  revenue per unit of m
/1*11 0, 12 5, 13 5/
bmin(j) min capacity of j
/1 0, 2*4 0, 5*6 20/
bmax(j) max capacity of j
/1 100, 2 100, 3 150, 4 300, 5*6 200/
meu(i) primary material of task i
/1*7 1/       ;

TABLE
sigma(m,i) mass balance coefficient of m to i
       1    2    3    4       5        6        7
1     -1
2          -1
3      1        -0.5
4           1   -0.5   0.1
5                 1    -1
6                      0.4    -0.25
```

```
7               0.5         -0.4
8                     -0.75
9                      1           -1
10                          -0.4     0.6
11                          -0.2
12                           1
13                                   0.4     ;
TABLE
tau(i,j) constant production term
     1    2    3    4    5    6
1    1
2         2    2
3         1    1
4                  3
5    1.5
6                        2    2
7         2    2                      ;
```