國立清華大學

碩士論文

考量順序相依整備時間與期初存貨之產能限制批量排程問題

Capacitated Lot-Sizing and Scheduling Problems with Sequence-Dependent Setup Time and Initial Inventory

所別 工業工程與工程管理學系　組別 工業工程組

學號姓名 9834516 賴志皓 Chih-Hao Lai

指導教授 陳茂生 博士 Dr. Maw-Sheng Chern

中華民國一〇〇年六月

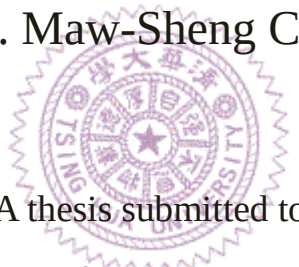
Capacitated Lot-Sizing and Scheduling Problems with Sequence-Dependent Setup Time and Initial Inventory

by

Chih-Hao Lai

Advisor

Prof. Maw-Sheng Chern



A thesis submitted to

The Department of Industrial Engineering and

Engineering Management

at

National Tsing Hua University

Hsin Chu, Taiwan 300, R. O. C

June 2011

摘要

本論文主要考慮單一機台(single-machine)產能限制批量排程問題(capacitated lot-sizing and scheduling)，它其包含順序相依整備時間(sequence-dependent setup time)以及期初存貨條件。我們提供一個方法將問題轉換成一個沒有期初存貨的問題。對於小型問題，我們建構一個混整數規劃模式，此模式將比文獻中舊有的模式更一般化。對於大型問題，我們提出一個啟發式演算法，它能夠在合理的時間內找到可行解。這個方法分成兩個階段，第一階段利用一個後推式方法產生一個好的可行解，接著在第二階段對此解做進一步改善。這個啟發式方法也被應用在平行機台問題(parallel machine)與流水線問題(flow shop)。

關鍵字: 產能限制批量排程問題; 混整數規劃; 順序相依; 整備時間; 期初存貨; 平行機台; 流水線。

Abstract

In this thesis we mainly consider single-machine capacitated lot-sizing and scheduling problems with initial inventory and sequence-dependent setup time. We provide a simple procedure to convert the problem into the one without initial inventory. To deal with small size problem we propose a mixed-integer programming formulation which generalizes the former models in the literatures. For large-scale programs, we provide a heuristic method which will generate a feasible schedule in reasonable time. The heuristic comprises two stages. In stage one a backward method is used to generate a good feasible schedule and the schedule is refined in stage two. Moreover, the proposed problem and algorithm are extended to parallel-machine and flow shop problems.

Keywords: Lot-sizing and scheduling; Mixed-integer programming;
Sequence-dependent; Setup time; Initial inventory; Parallel machine; Flow shop.

Contents

Chapter 1 Introduction

In the competitive environment, inventory problems play an important role in various industries. With the increasing of the inventories, carry cost may be increased. Moreover, opportunity cost will increase while capitals are accumulated. Lot-sizing and scheduling will be discussed in the further chapters; furthermore, lot-sizing and scheduling is a crucial and worth researched problem in production and inventory management. Pursuing a low cost inventory is the goal of in the concept of Just-in-Time. However, in the condition of dynamic demand and setup time considered environment, low cost inventory has its difficulties to be executed. In this thesis, we consider the capacitated lot-sizing and scheduling problem (CLSP) with sequence-dependent setup time. The goal is to minimize inventory cost under the restrictions of satisfying capacity, demand and setup time.

1.1. Research Background

Scheduling is a decision that helps allocating information. According to the amount of products and the due time that provided from Master Production Schedule (MPS), sequencing and timing the capacities of machines and employees are the purposes of finding a feasible schedule that satisfy the demand. Therefore, scheduling problem is to combine the results from long-term capacity planning and mid-term aggregate planning. Moreover, the results of MPS will be distributed to various works, activities, or customers, in order to balance the efficiency, inventory level and service level.

Lot-sizing and scheduling is to make decisions simultaneously on the processing time, order, and production quantity. Furthermore, the objective is to minimize the cost. There are few features for this kind of problem.

(1) Planning Horizon

Planning Horizon can be infinite or finite. Finite planning horizon usually associates dynamic demand while infinite planning tends to be steady demand. Furthermore, planning horizon can be categorized as continuous and discrete, and they are corresponded as large bucket and small bucket problems respectively. In the large bucket problems, there is enough time to manufacture various products in a period. However, there is at most one product can be manufactured in a period in the small bucket problems.

(2) Levels

Products can be divided into single-level and multi-level in a manufacturing system. If products do not include parent-components relationships in the system, then it can be called as single-level. Nevertheless, products can be called as multi-level when there are parent-components relationships.

Figure 1.1 reveals that the structure might be different providing that the relationships are vary. Haase (1994) categorized product structure into the following types:

-*Serial*: Each component is formed from at most one predecessor. Furthermore, there is at most one successor.

-*Assembly*: The final component is formed from various predecessors. Except the final component, there is only one successor for each component.

-*General*: The final component is formed from various predecessors. And it is acceptable that each component contains more than one successor.

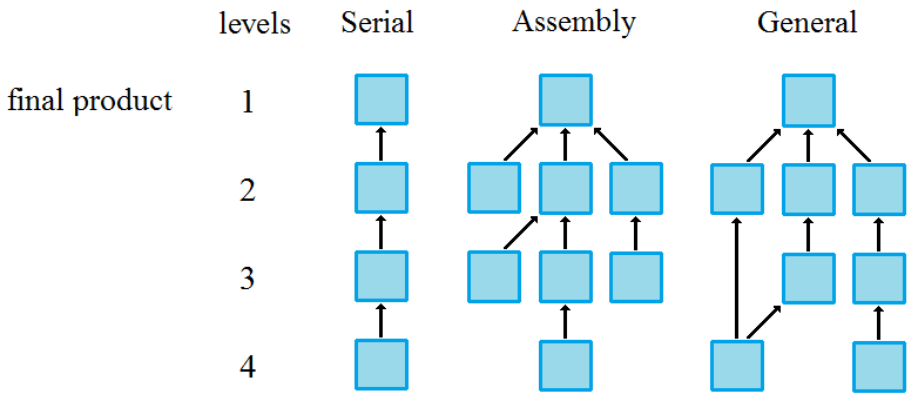


Figure 1.1. Product Structures

The final product is usually simpler in the single-level production system. Final products are finished after raw materials are produced, which means that the products do not need to be assembled. The problem in this thesis is based on the single-level production system.

(3) Number of Products

The modeling and complexity of problems are connected with the number of products. Generally speaking, the amount of products can be divided into two parts: single-item and multi-item. The complexity is higher for multi-item than single-item. This thesis is all about multi-item.

(4) Number of Machines

The amount of machines can be divided into two parts: single-machine and multi-machine. There are several cases for multi-machine problems. In parallel machine case, many products are produced on machines simultaneously, and products will be finished by only an operation on any one machine. In flow shop case, products must go through all of machines in the same order. The complexity is much higher for multi- machine than single- machine.

(5) Capacity Constraints

In the manufacturing system, capacity constraints contain human resource,

facilities, and budgets, etc. When there are no limitations, it can be called as uncapacitated. However, capacitated problems are more likely to take place in reality, and which is the main part of this thesis.

(6) Demand

Demand data can be categorized into the following:

-*Steady Demand*: The amount of demand is a stationary; it will not be changed from time to time.

-*Dynamic Demand*: The amount of demand will vary from time to time.

-*Deterministic demand*: The amount of demand can be known as an exact number in advance.

-*Probabilistic Demand*: The amount of demand is decided base on some probability.

The input data of demand in this thesis will be deterministic dynamic demand according to the reality.

(7) Setup

The definition of setup refers to two different products doing changeover within the manufacturing activities. Setup is a key point that concerns to the complexity of the problem. If setup time or costs are considered in the problem, many binary variables need to be added into mathematics model. It also upgrades the difficulty of the problem. Setup time and cost are normally essential while setup is taking place. Setup can be divided into two types normally. Firstly is simple setup, its time and cost are based on the latter product of the two different products, which refer that setup focus on individual product. If the setup is vary from the former product of the two, then the manufacturing sequence needs to be considered, and setup will be much more complicated. The triggered time and costs can be called as sequence-dependent setup time and sequence-dependent setup costs. In this thesis, we consider the sequence

related setup time. However, setup costs is not considered. One of the main example actually exist in our daily life, setup time can be seen as opportunity costs. This means that the production quantity of the loss in the setup can be seen as the loss in production capacity.

(8) Service Policy

Service policy includes several rules that relate to service level. The service level is 100% in the condition of products shortage are not allowed. If the shortage is acceptable, then the current requirements may be satisfied in the further backlogging, or the requirements will not be satisfied at all lost sales. The problems with the permission of shortage are much more complex than the ones without the shortage. What are discussed in this study is on the premise of no product shortage.

1.2. Research Motivation

In the literatures, Florian et al. (1980) proves that CLSP is NP-hard and Maes et al. (1991) claims that if setup time is considered, then the problem of finding a feasible solution is NP-complete. In this thesis, we also consider the sequence-dependent setup time. Thus, the problem will be more complicated. For dealing the problem with the consideration of setup time, mathematical programming model or heuristic algorithm on the basis of mathematical programming or enumeration, are developed for small problems (the number of products × the number of periods $\leq$ 70). Under this condition, optimal solution or good approximation solution can be achieved. However, for large-size problems, it seems to be impractical to find out the feasible solution with reasonable time consuming.

In this thesis, we consider a lot-size scheduling problem for a US food company. The product types is more than 10 and the whole planned time duration is one year (52 periods), and the sequence-dependent setup time is also considered. In this thesis,

a more general mathematical model will be presented first, and a heuristic method will be developed rapidly in order to find out a good feasible solution for the large-size problem.

1.3. Research Scope and Assumptions

There are several features of the problem that is discussed in this study: finite planning horizon, single-machine, single-level products, the limited capacity of production, the deterministic dynamic demand, costless for setup, sequence-dependent setup time, and the prevented of the products shortage. The basic assumptions are as follows. The demands are satisfied at the end of each period without shortage. The setup state at the end of this period will be delivered to the next producing, and if the products of next producing happen to be the same as the last one, then no setup is needed again. If there is initial inventory, then the inventory in the end of planning horizon should more or equal to the initial inventory, that is to say, all the demand can be satisfied by the production.

1.4. Research Method

First, we construct a mathematical model to describe the problem clearly. This model is more general than previous models for large bucket problems.

According to Haase and Drexl (1995), a backward heuristic method named BAPLSP can be used efficiently for small bucket problems. In this thesis, a two-stage heuristic method is developed. In the first stage, the method of Haase and Drexl (1995) will be modified to deal with large bucket problems and will be improved to integrate the sequence-dependent setup time and initial inventory for achieving the feasible solutions. An improved procedure will be sought based on the feasible solutions in the second stage.

1.5. Framework

There are seven chapters included: the background, motivation, objective and research methods of this study will be introduced in the first chapter. Literature review is shown in chapter 2. In chapter 3, we provide a mathematical model for proposed problem and develop a two-stage heuristic method which can find feasible solutions efficiently for large-size problems. This method will be a lot-sizing and scheduling method that is proper for addressing problems in practical producing environment. Moreover, in chapter 4, we extend the proposed problem for parallel machine. We also present a mathematical model for the parallel machine problem. the propose heuristic is modified and extended to solve parallel machine problems. In chapter 5 a flow shop type of propose problem is introduced. Finally, the conclusion and future research will be discussed in chapter 6.

Chapter 2 Literature Review

This chapter will be divided into four sections. Section 2.1 reviews the research of lot-sizing and scheduling problems which includes various mathematical models. Section 2.2 reviews the research of different algorithms. Moreover, parallel machine case and the flow shop case are reviewed in section 2.3 and section2.4.

2.1 Lot-sizing and Scheduling Problems

For lot-sizing and scheduling problem, demands are given for each product and period. Demands are satisfied without shortage in the end of each period. Inventory cost is occurred when a product holding a period. For each product, a specific processing rate is defined, that means how many quantities can be produced in a time unit. The capacity, the available production time, is limited, and it may not be a constant for each period. The changeover between two different products occurs setup time or setup cost. The problem is how to schedule the right product produced at the right time. The production quantity also should be decided simultaneously. When setup time or setup cost is significant, we prefer to produce the same product as long as possible to reduce setup time or cost, but it will also increase inventory costs. Therefore, the trade-off of this problem is to find the balance of setup costs and inventory costs such that the total cost is minimized and demands are satisfied. A numerical example is given in Example 2.1 and a corresponding Gantt chart is in Figure 2.1.

Example 2.1. There are two products and three periods being scheduled. The available production time is 50 for three periods. For two products, the processing rate is 1 and the holding cost is 5 for to keep a product a period. Demands and setups are

given in following table.

Table 2.1. Parameter Values

	demands			processing rate	setup time	setup cost	holding cost
	period 1	period 2	period 3				
product 1	20	10	20	1	5	100	5
product 2	0	40	30	1	10	200	5

A feasible schedule is given as follows: the production quantity for each period is 30, 0 and 20 for product 1 and 0, 45 and 25 for product 2. The total cost is 475.

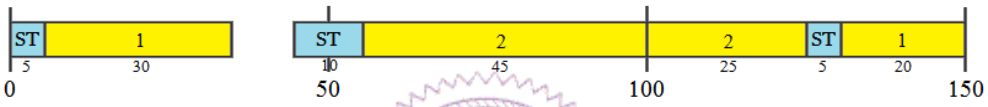


Figure 2.1. Gantt Chart of the Schedule

Economic production quantity (EMQ) is a typical model to deal with lot-sizing and scheduling problems. Basic EMQ model has assumptions which are single-level, single-item, infinite planning horizon and infinite production capacity. Furthermore, the demand rate is assumed to be a known constant. Under these strict assumptions, an optimal solution can be easily obtained. However, the assumptions of basic EMQ model are impractical. Over several decades, many researchers continuously develop more general models to solve real world problems.

Karimi et al. (2003) reviewed single-level lot-sizing problem, and it includes the uncapacitated case and the capacitated case. Drexl and Kimms (1997) also focused on the comparison of different models such as capacitated, dynamic demand and deterministic case. Staggemeier and Clark (2001) reviewed the problem with multi-level case and multi-machine case.

2.1.1 Basic Lot-Sizing and Scheduling Model

The case of uncapacitated, single-level, single-item and total T periods can be formulated as a trivial problem as follows:

$$\text{Min} \sum_{t=1}^{T} hI_t \,, \tag{2.1}$$

subject to

$$I_{t-1} + x_t - d_t = I_t \,, \quad t = 1, \ldots, T \,, \tag{2.2}$$

$$I_t \geq 0 \quad \text{and} \quad x_t \geq 0 \,, \quad t = 1, \ldots, T \,, \tag{2.3}$$

where h denotes the inventory cost of a product for holding a period and d_t denotes the demand of a product at the end of period t. The decision variable x_t represents the production quantity of the product at period t and the variable I_t represents the inventory of the product at the end of period t.

The objective function (2.1) is to minimize the total inventory cost. Equations (2.2) ensure the balance of demands and productions in each period. It can be easily solved without capacity restrained. The optimal lot-sizes exactly equals to the demand in each period, i.e. $x_t = d_t$, $t = 1, \ldots, T$ with zero inventory costs.

Staggemeier and Clark (2001) presented some basic features of the Lot-Sizing and Scheduling. By adding variant constraints to the original model, different situations can be described from the new model. Constraints are introduced as follows:

(1) Capacity Constraint

Infinite capacity does not exist in the real world. For this situation, we add a set capacity constraints.

$$x_t \leq C_t \,, \quad t = 1, \ldots, T \,, \tag{2.4}$$

where C_t denotes the available production capacity in period t. If $C_t \geq d_t$, $t = 1, \ldots, T$, then the optimal solution is also $x_t = d_t$, $t = 1, \ldots, T$. However, if $C_t < d_t$, the solution

will be more complicated.

(2) Presence of Backlogs

The presence of backlogs means the allowance of shortage and the shortage must be satisfied in the future. In order to make a description of backlogs, we have to introduce new variables and equations (2.1), (2.2) and (2.3) are revised as follows:

$$\text{Min} \sum_{t=1}^{T} hI_t^+ + bI_t^- , \tag{2.5}$$

subject to

$$I_{t-1}^+ - I_{t-1}^- + x_t - d_t = I_t^+ - I_t^- , \quad t = 1, \ldots, T , \tag{2.6}$$

$$I_t^+ \geq 0 , \ I_t^- \geq 0 \ \text{ and } \ x_t \geq 0 , \quad t = 1, \ldots, T , \tag{2.7}$$

where I_t^+ and I_t^- represent the inventory and backlog quantities at the end of period t respectively. b is the cost for delivering the backlog for a period.

The object of this problem is to minimize the total of inventory costs and backlog costs. Note that, in the optimal solution, the quantity of backlog and inventory in each period may not be greater than zero at the same time. Since it can save costs by reducing both two quantities simultaneously until one of them reaches zero.

(3) Presence of Setup Costs

In this case, setup cost occurs in each production in a period; therefore, it may produce more to satisfy the demand in the future. It may elevate inventory cost but save setup costs. The new objective function (2.8) as follows:

$$\text{Min} \sum_{t=1}^{T} \left(hI_t + sy_t \right), \tag{2.8}$$

where s is the setup cost. The object is minimizing the sum of setup and inventory costs. We have to introduce a set of binary variables y_t to indicate whether the setup occurs in period t (y_t = 1) or not (y_t = 0). Constraints (2.9) describe that if the product is produced in period t ($x_t > 0$) then y_t is forced to be one.

$$x_t \leq M_t \cdot y_t , \quad t = 1, \ldots, T , \tag{2.9}$$

where M_t is the upper bound of x_t. This is a mixed-integer programming (MIP) problem.

(4) Presence of Setup Time

In this case, setups not only occur costs but also consume the capacity which is called setup time. The capacity constraints are modified as follows:

$$x_t + y_t \cdot s \le C_t, \quad t = 1, \ldots, T, \tag{2.10}$$

where s denotes the setup time for the production.

(5) Multiple Products

We can construct the model with multiple products by introducing new sets of variables. Basically, it is the same as single-product. The model is as follows:

$$\text{Min} \sum_{j=1}^{J} \sum_{t=1}^{T} h_j I_{jt}, \tag{2.11}$$

subject to

$$I_{j,t-1} + x_{jt} - d_{jt} = I_{jt}, \quad j = 1, \ldots, J, t = 1, \ldots, T. \tag{2.12}$$

In this case, the variables I_{jt}, x_{jt} and d_{jt} denote the inventory, production quantities and the demand for product j at the end of period t, respectively. Moreover, h_j denotes the inventory cost of product j. However, it is essential to add a set of new constraints in order to relate the capacity as follows:

$$\sum_{j=1}^{J} u_j x_{jt} \le C_t, \quad t = 1, \ldots, T, \tag{2.13}$$

where u_j is the capacity required for producing one unit of product j.

(6) Multiple Machines

There are M parallel machines. It can make scheduling more flexible but more complicated. We revised the balance equations (2.12) as follows:

$$I_{j,t-1} + \sum_{m=1}^{M} x_{jmt} - d_{jt} = I_{jt}, \quad j = 1, \ldots, J, t = 1, \ldots, T, \tag{2.14}$$

where x_{jmt} denotes the production quantity of product j on machine m in period t.

The capacity constraints are also modified as follows:

$$\sum_{j=1}^{J} u_{jm} x_{jmt} \le C_{mt}, \quad m = 1,\ldots,M, t = 1,\ldots,T, \tag{2.15}$$

where u_{jm} denotes the capacity required for product j on machine m and C_{mt} denotes the capacity of machine m in period t.

2.1.2 Capacitated Lot-Sizing Problem (CLSP)

This is a well-known single-machine problem that has J products which are arranged in T periods. Backlog is not allowed. Haase (1994) introduced the CLSP and presented an improved model. We will discuss the new model in section 2.1.3.

Table 2.2. Decision Variables

Variables	Definition
I_{jt}	Inventory of product j at the end of period t
x_{jt}	Production quantities of product j in period t
y_{jt}	y_{jt} = 1, if a setup for product j occurs in period t.(y_{jt} = 0, otherwise.)

Table 2.3. Parameters

Parameters	Definition
J	Number of products
T	Number of periods
C_t	Available production capacity in period t
d_{jt}	Deterministic demand for product j in period t
h_j	Holding costs for product j
I_{i0}	Initial inventory for product j
u_j	Capacity needs for produce one unit of product j
s_j	Setup costs for product j

The CLSP needs a basic assumption: Setup costs occur for each lot in a period.

The decision variables and the parameters of the CLSP are given in Table 2.2 and Table 2.3. The CLSP can be formulated as a MIP as follows:

$$\text{Min} \sum_{j=1}^{J} \sum_{t=1}^{T} \left(s_j y_{jt} + h_j I_{jt} \right), \tag{2.16}$$

subject to

$$I_{j,t-1} + x_{jt} - d_{jt} = I_{jt}, \qquad j = 1, \ldots, J, t = 1, \ldots, T, \tag{2.17}$$

$$\sum_{j=1}^{J} u_j x_{jt} \le C_t, \qquad t = 1, \ldots, T, \tag{2.18}$$

$$u_j x_{jt} \le C_j \cdot y_{jt}, \qquad j = 1, \ldots, J, t = 1, \ldots, T, \tag{2.19}$$

$$y_{jt} \in \{0, 1\}, \qquad j = 1, \ldots, J, t = 1, \ldots, T, \tag{2.20}$$

$$I_{jt} \ge 0 \quad \text{and} \quad x_{jt} \ge 0, \qquad j = 1, \ldots, J, t = 1, \ldots, T. \tag{2.21}$$

The object (2.16) is to minimize the sum of inventory costs and setup costs. Constraints (2.17) and (2.18) describe the inventory balance and the capacity restriction, respectively. Constraints (2.19) ensure that if the production takes place in period t ($x_{jt} > 0$) then the corresponding binary variable should be one ($y_{jt} = 1$). Furthermore, the non-negativity and binary condition are shown by (2.20) and (2.21).

The CLSP is a large bucket problem, because it allows producing more than one product in a period. In a large bucket real world problem, a period usually represents a week or a day, and in the small bucket problem, a period usually represents an hour or a shift.

The problem is to decide what products are produced in each period and lot-sizes. The setup cost of this model is determined by the individual product. It is sequence-independent. Therefore, the total cost is independent of the product sequence in each period. Usually, we obtain lot-sizes by solving the CSLP and then

use other methods to determine product sequences. Many researchers have mentioned the CLSP and its variants such as considering of sequence-dependent setup, setup time, multi-level and multi-machine, etc.

However, solving the single-product CLSP optimally is proven to be a NP-hard problem by Florian et al. (1980) and Bitran and Yanasse (1982). The classical CLSP with multi-product is proven to be *strongly* NP-hard by Chen and Thizy (1990). So there are no many researchers developed exact method to solve optimally. Many heuristic methods have also been developed in the last 20 years. We will review the algorithms in section 2.2.

Haase (1994) mentioned a drawback of the CLSP. The setup state of the machine cannot be kept to the next period. It means that if product k is the last product produced in period t and also the first product produced in period $t + 1$ then it will occurs a new setup cost in period $t + 1$. In order to illustrate the previous concept, let us consider the following example.

Example 2.2. Let $J = 2$, $T = 2$, $C_1 = C_2 = 10$. There is no initial inventory. Demands d_{jt}, setup time s_j, capacity needs u_j and holding costs h_j are shown in Table 2.4.

Table 2.4. Parameter Values

	d_{j1}	d_{j2}	s_j	I_{j0}	u_j	h_j
$j = 1$	4	3	10	0	1	4
$j = 2$	0	5	10	0	1	9

The optimal solution is $x_{11} = 4$, $x_{12} = 3$ and $x_{22} = 5$ with total costs 150 that has zero inventory cost. It is obvious that if the setup state at the end of period 1 can be preserved, then total cost will reduce 10 units by saving a setup for product 1. A Gantt chart is given in Figure 2.2.

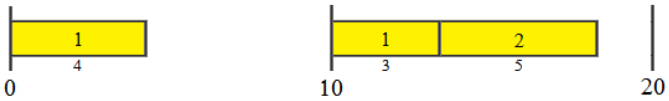


Figure 2.2. Gantt Chart of the Solution

To improve the above situation, some extensions of the CLSP have been developed which can save setup costs by linking productions of adjacent periods. We discuss it next section.

2.1.3 The CLSP with Linked Lots (CLSPL)

Kaczmarczyk (2009) stated the CLSP with linked lots in detail. The CLSP considers that it can produce multiple products in one period, which means setups may occur in one period, without considering that in the adjacent period setup state can be reserved. But it is rare to happen without the setup reservation in practical.

To preserve the setup state between adjacent periods is one of the ways to improve the CSLP. It will save the additional setup costs reasonably. In past literatures the preservation of setups is called setup carry over or lot-linking. The difference of total costs with and without lot-linking is shown in Example 2.3. Many models which consider the lot-linking have been developed. The complete mathematical model about CLSPL can be found in Haase (1994), Haase (1996), Sox and Gao (1999), Suerie and Stadtler (2003).

Example 2.3. Let $J = 1$, $T = 3$, $s_1 = 50$, $u_1 = 1$ and $h_1 = 1$. The demand d_{jt} and capacity C_t are shown in Table 2.5.

Table 2.5. Parameter Values

	$t = 1$	$t = 2$	$t = 3$
d_{jt}	3	9	7
C_t	10	10	10

The optimal solution is x_{11} = 3, x_{12} = 9 and x_{13} = 7. Then total costs with and without lot-linking is 150 and 50, respectively. A Gantt chart is given in Figure 2.3.

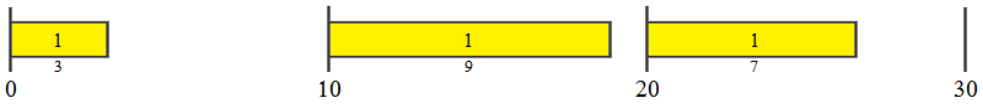


Figure 2.3. Gantt Chart of the Schedule

2.1.4 The Discrete Lot-Sizing and Scheduling Problem (DLSP)

The DSLP is another famous model and it is a small bucket problem. More than one product can be produced in each period in CLSP. And it can be called as macro-period. However, there is at most one product can be produced in each period in DLSP, and it is called as micro-period. Micro-period is split from macro-period.

The DSLP is based on the following assumption: The production process always runs full periods without changeover. This assumption leads the DLSP to make a feature of the 'all-or-nothing' production. The DLSP can also be formulated as a MIP with the same objective function as CLSP. Details can be found in Fleischmann (1990) and Haase (1994). The complexity analysis is in Salomon et al. (1991). The DLSP is also proven to be NP-hard.

2.1.5 The Continuous Setup Lot-Sizing Problem (CSLP)

In the DLSP, the 'all-or-nothing' assumption is strict. Even the hard assumption can make the model implementable, and the implementation is more efficient. However, this makes the model limited and it is not suitable for the reality. The CSLP lot-sizes are considered to be continuous quantities instead of full period or zero. The 'all-or-nothing' is not considered in the new model. However, in each period there is at most one product can be produced in the CSLP. Several literatures describe this model

in detail, e.g. Haase (1994) or Drexl and Kimms (1997).

The CSLP is based on the following assumption: At most one product can be produced per period. To compare with DLSP, there were only few researches focused on CSLP. Please refer Bitran and Matsuo (1986) and Karmarkar et al. (1987).

2.1.6 The Proportional Lot-Sizing and Scheduling Problem (PLSP)

Drexl and Haase (1995) presented a new model, the proportional lot-sizing and scheduling problem and provided a backward-oriented heuristic method to solve it. The PLSP revised the CSLP's shortcoming that capacities of periods left over because it can only produce a product per period. Basically, the idea of the PLSP is to produce the second product consuming the remaining capacity in periods to increase the utilization of whole system.

The PLSP is based on the following assumption: At most one Changeover is allowed within each period. Drexl and Haase (1995) also mentioned some extensions of the PLSP. Setup times and the multi-machines case are considered respectively. For more variants of PLSP, please refer to Haase (1994), Drexl and Haase (1996) and Kimms (1996).

2.1.7 Models with Setup Times and Sequence-Dependent Setup Times

For lot-sizing and scheduling it is more complicated to incorporate the setup time, because it is difficult to check the feasibility of the problem, especially with sequence-dependent setup times.

For small bucket problems, Haase (1994) provided an extension of PLSP with setup times, named PLSPST, and it can be revised to the sequence-dependent case by only to change a set of constraints.

For large bucket problems, for the CLSP, setup times can be considered by to

replace constraints (2.18) to the following:

$$\sum_{j=1}^{J}\left(u_j x_{jt} + s_j y_{jt}\right) \le C_t, \quad t = 1, \ldots, T, \tag{2.22}$$

where s_j denotes the setup time for product j.

The constraints (2.22) show that if product j produced in period t ($y_{jt} = 1$) then the setup time of product j (s_j) also occupied a part of capacity.

Trigeiro et al. (1989) have shown that bin packing problem can be reduced to a special case of CLSP with setup times. Hence, it is NP-hard.

With the sequence-dependent setup time, two kinds of mathematical models are developed in the past literature. One is formulating directly and the other is only considering the efficient sequence. The efficient sequence is defined as the sequence that has the smallest setup time given the first and last products, and the sets of all products produced in the period. Gupta and Magnusson (2005) presented a mathematical model which is formulated directly and considered with sequence-dependent setup time and also provide a heuristic, named ISI heuristic, to solve the problem. Almada-Lobo et al. (2008) found that there were some mistakes in Gupta's model and correct it. Haase and Kimms (2000) presented a MIP model, named LSPSD, which is formulated only with the efficient sequence obtained by solving separate Travelling Salesman Problems (TSP). The approach solves problem optimally and efficiently. Kovács et al. (2009) presented a new MIP model which is similar to Haase and Kimms (2000). They provided a method based on Dynamic Programming to determine all the efficient sequence and a heuristic to control the tightness of the MIP problem. As long as the product of the quantity of products and the periods is less than 60 to 70, a high-quality solution can be obtained from the model in a reasonable time.

Furthermore, all the mathematical models which describe CLSP with setup time

or sequence-dependent time in the past have a basic assumption: Each setup will be completed within a period. This assumption is unreasonable for us, because it may make some specific problems insolvable. We will modify such situation by constructing a new mathematical model without the assumption in chapter 3.

2.1.8 Summary

Many mathematical models have been developed to solve variant lot-sizing and scheduling problems. However, the basic model, like the CLSP or the DLSP, is simple and has been known as NP-hard problems. Although lots of new models skillfully contain far more situations, like multi-level product, multi-machine, setup time, sequence-dependent setup time, these models are too complicated because they include too many binary variables and constraints. Therefore, they can only deal with very small-size problems.

We make a summary by the following table. Table 2.6 provides features of basic models. Note that, although the PLSP is small bucket problem, this model can produce at most two products in one period.

Table 2.6. Basic Models

Bucket	Model	Max. no. of different items per period	Preservation of setups	Lot-size	Setups
Small	DLSP	1	No	Multiples of a full period production	No
Small	CSLP	1	Yes	Continuous quantities	No
Small	Haase (1994); PLSP	2	Yes	Continuous quantities	No
Large	CLSP	No restriction	No	Continuous quantities	No

Table 2.7 provides features for large bucket problem, involved typical CLSP and variants.

Table 2.7. Variants of CLSP

Model	Formulation Type	Preservation of setups	Setups
CLSP	Directly	No	No
Haase (1994); CLSPL	Directly	Yes	No
Almada-Lobo et al. (2008)	Directly	Yes	Sequence-dependent setup time and costs
Haase and Kimms (2000)	Efficient sequence	Yes	Sequence-dependent setup time and costs
Kovács et al. (2009)	Efficient sequence	Yes	Sequence-dependent setup time and costs

The complexity for models can be summarized as Table 2.8.

Table 2.8. Complexity of Models

Bucket	Model	# of real variables	# of binary variables	# of constraints
Small	DLSP	$J \times T$	$2 \times J \times T$	$T + 3 \times J \times T$
Small	CSLP	$2 \times J \times T$	$2 \times J \times T$	$T + 5 \times J \times T$
Small	Haase (1994); PLSP	$2 \times J \times T$	$2 \times J \times T$	$2 \times T + 3 \times J \times T$
Small	Haase (1994); PLSPST	$3 \times J \times T$	$3 \times J \times T$	$2 \times T + 7 \times J \times T$
Small	Haase (1994); PLSPSDSC	$2 \times J \times T$	$J \times T$	$2 \times T + 5 \times J \times T$
Large	CLSP	$2 \times J \times T$	$J \times T$	$T + 4 \times J \times T$
Large	Haase (1994); CLSPL	$T + 2 \times J \times T$	$2 \times J \times T$	$4 \times T + 7 \times J \times T$
Large	Almada-Lobo et al. (2008)	$2 \times T + 3 \times J \times T$	$4 \times J \times T + J \times T^2$	$5 \times T + 8 \times J \times T + 4 \times J \times T^2$
Large	Haase and Kimms (2000)	$2 \times J \times T$	$O(J^2 \times 2^J)$	$2 \times T + 6 \times J \times T$
Large	Kovács et al. (2009)*	$T + 3 \times J \times T$	$2 \times J \times T$	$O(J^2 \times T \times 2^J)$

*Kovács et al. (2009) is a mathematical model based heuristic.

T: Number of periods
J: Number of products

In terms of most scheduling problems, they are often too complicated because they contain many integer variables. We cannot solve problems within reasonable time by using integer programming directly, so mathematical models can only offer systematical descriptions, but they cannot be used to solve problems. We will mention several ways of solving problems in literatures in next section.

2.2 Algorithms

According to the former literatures, Karimi et al. (2003) classified solution methods of capacitated lot-sizing problem into three groups. They are *exact methods*, *common-sense* or *specialized heuristics* and *mathematical programming-based heuristics*. We will review in accordance with this classification.

2.2.1 Exact Methods

Since CLSP is a NP-hard problem, it is difficult to solve the CLSP optimally. Therefore, most of the study tends to heuristics. However, there are still some researchers making efforts in exact methods. The most direct method is to use branch and bound technique to solve a mixed-integer programming. In addition, there are two exact methods. Cut-generation technique was presented in Barany et al. (1984) and Leung et al. (1989). Eppen and Martin (1987) provided another approach by using the variable redefinition technique. Until now, there are few literatures solving CLSP with setup time by using exact method.

The cut-generation technique is introduced in Barany et al. (1984). The typical CLSP is reformulated by adding valid inequalities using a cutting plane algorithm. The inequalities are related to separate single-item uncapacitated problems. This reformulation can make the solution processed faster and provides an approximation of the convex hull of solutions for the CLSP. The reformulated problem is then solved by using a branch-and-bound algorithm. Problems that are made up of 20 items and 13 periods can be solved. Lots of commercial software can be used via this approach.

The other approach, a variable redefinition technique, is presented in Eppen G.D. and Martin (1987). This approach transforms the typical CLSP into a graph-based representation. This approach uses branch-and-bound procedure and solves LP-relaxation problems to find the optimal solution. Despite the fact that this approach will generate much more variables and constraints, it has tighter linear

relaxation than the original one. Therefore, its solution time is much shorter.

The above mentioned solution method are not suitable to solve production scheduling problems in realistic for the reason that they take excessive solution times. Many practical heuristic methods have been extensively developed. We will review several heuristic methods in next section.

2.2.2 Common-Sense or Specialized Heuristics

According to Maes and Van Wassenhove (1988) and Karimi et al. (2003), common-sense or specialize heuristics can be separated into two sorts in terms of the procedure of solving problems: *period-by-period heuristics* and *improvement heuristics*.

Period-by-period heuristics work, as its name, one period after another. Two classifications can be employed: forward method and backward method. The forward method works from period 1 to period *T*. Demands of specify period will be firstly satisfied in each period. And the rest capacity will be used to satisfy demands in the future in order to avoid the following shortage of capacity and save setup costs. The choice of product produced in advance and lot-sizes are determined by using the priority index, which is used in most of literatures. The priority index can be determined by using some simple criteria such that the value of saving cost or relation to setup time. Such approach is presented first in Eisenhut (1975), and also used in Maes and Van Wassenhove (1986), and Kirca and Kokten (1994). The other way is backward method, working from period *T* to 1. In each period, after satisfying the demand, the rest can be idle capacity. If we cannot fulfill all the demands, we can move the unsatisfied demand to the earlier period which contains excess capacity. Compared to forward method, backward method is easier to obtain a feasible solution. Products can be produced more precisely by implementing backward method. The

demand, that is hard to be satisfied in the future, will be produced in advance. This method is employed in Haase (1994) and Drexl and Haase (1995).

Improvement heuristics often begins with an initial solution. Sometimes it is an infeasible solution, and it can be generated by the technique to solve uncapacitated problems. Normally, improvement heuristics has three steps: the initial step, feasibility check routine and the improvement step. During the initial step, we often ignore capacity constraints and create a rough initial solution, which is often an infeasible solution. To obtain an initial solution, methods might be differ in different heuristic. Lot-for-lot is a frequently-used method. Sequentially, in feasibility check routine, productions in certain period may exceed the capacity and lead to shortage due to capacity constraints. At this point, we should move parts of the production to earlier period which holds idle capacity, trying to eliminate all the infeasibility (the shortage). If setup time is taken into consideration, it will be added into schedule at the step. Finally, several rules such that rearranging the sequence in periods or exchanging productions are applied to refine and improve the solution obtained by previous procedures in the improvement step. Trigeiro (1989) presented a heuristic method, named Simple Heuristics, for CLSP with setup time. More recent work such as Gupta and Magnusson (2005) provided a heuristic named ISI Heuristic for CLSP with sequence-dependent setup costs and setup time.

2.2.3 Mathematical Programming-Based Heuristics

These kinds of heuristics usually generate a solution by solving a mathematical programming problem. The main advantage is that such heuristics can obtain a high quality solution, and can be implemented by using many commercial solvers such that LINGO (www.lindo.com) and CPLEX (www.ilog.com). Another advantage is that such heuristics can usually provide a reasonable lower bound for the optimal solution,

allowing users to distinguish good solutions from bad ones. However, on the contrary, the method is not only more complicated than common-sense heuristics, but also harder to achieve. We usually use Lagrangian relaxation technique to loosen the capacity constraints, dividing the original problem into several minor uncapacitated problems. According to Karimi et al. (2003), we can subdivide such heuristics on the basis of the differences of methods: relaxation heuristics, branch and bound heuristics, set partitioning and column generation heuristics and other approaches. There are related literatures, such as Thizy and Van Wassenhove (1985) for CLSP, Süral et al. (2009) for CLSP with setup time, and Kovács et al. (2009) for CLSP with sequence-dependent setups.

Table 2.9. Features of Heuristics

Literatures	Problem	Approaches
Exact Method		
Barany et al. (1984)	CLSP	Exact method: Cut-generation
Leung et al. (1989)	CLSP	Exact method: Cut-generation
Eppen and Martin (1987)	CLSP	Exact method: Variable redefinition
Haase and Kimms (2000)	CLSP with SDST and SDSC	Exact method: Branch and bound
Almada-Lobo et al. (2008)	CLSP with SDST and SDSC	Mixed-integer programming
Common-Sense or Specialized Heuristics		
Eisenhut (1975)	CLSP	Period-by-period heuristic
Maes and Van Wassenhove (1986)	CLSP	Period-by-period heuristic
Kirca and Kokten (1994)	CLSP	Period-by-period heuristic
Drexl and Haase (1995)	PLSP	Period-by-period heuristic
Trigeiro et al. (1989)	CLSP with ST	Improvement heuristics
Gupta and Magnusson (2005)	CLSP with ST and SDSC	Improvement heuristics
Mathematical Programming-Based Heuristics		
Thizy and Van Wassenhove (1985)	CLSP	Lagrangian relaxation based heuristic
Süral et al. (2009)	CLSP with setup time	Lagrangian relaxation based heuristic
Kovács András et al. (2009)	CLSP with SDST and SDSC	LP relaxations based heuristic

*ST: Setup time
*SDST: Sequence-Dependent Setup Time
*SDSC: Sequence-Dependent Setup Cost

2.2.4 Summary

We summarize all the characteristics of approaches and the corresponding

problems in Table 2.9.

2.3 Parallel Machine Problems

In this case each product can be produced on any one of the given machines. Demands are given as single-machine problem. For each product, the processing rate may be different for different machines. Capacities are also dynamic for each machine and each period. Moreover, the setup also depends on machines. We have to decide production sequences for each machine and production quantities simultaneously. An example is given in the following.

Example 2.4. There are two products and three periods being scheduled on two parallel machines. For each machine the available production time is 50 for three periods. Two machines are assumed to be identical, thus there are the same processing rates, setup costs and setup times for each product. Setup costs and setup times are sequence-independent. Demands and other parameters are given in following table.

Table 2.10. Parameter Values

	demands			processing rate for each machine	setup time for each machine	setup cost for each machine	holding cost
	period 1	period 2	period 3				
product 1	40	20	40	1	5	100	5
product 2	0	90	60	1	10	200	5

A feasible schedule is given in Figure 2.4 with total cost 850.

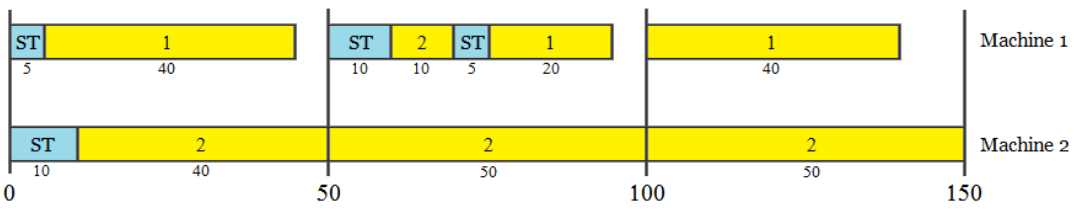
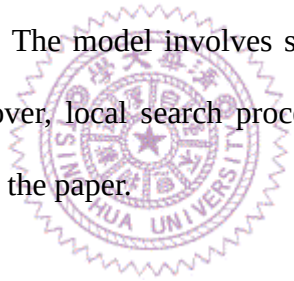


Figure 2.4. Gantt Chart of the Schedule

In Lasdon and Terjung (1971) the discrete lot-sizing and scheduling problem (DLSP) without setups in identical parallel machine is solved by using a heuristic approach. A dynamic programming algorithm is used to solve a single-product problem in parallel machine without capacity constraints and without shortage in Sung (1986). Carreno (1990) presented a heuristic for the economic lot scheduling problem (ELSP) on identical parallel machines with setups. The ELSP is model with constant demand rate for each product planned in continuous time and infinite planning horizon. Toledo and Armentano (2006) proposes a heuristic based on the Lagrangian relaxation of the capacity constraints and subgradient optimization for solving the CLSP with setup time on unrelated parallel machines. Meyr (2002) proposed a model for simultaneous lot-sizing and scheduling of multi-product on non-identical parallel machine. The model involves sequence-dependent setup times and disallows shortage. Moreover, local search procedures threshold accepting and simulated annealing are used in the paper.



2.4 Flow Shop Problems

In this case products must go through all of the machines in the same order. A product is called a semi-finished product of level *m* when the product is finished on machine *m*. When the product is finished on this machine, it moves to the next machine only if its production quantity is greater than or equal to a specific minimum lot-size. The task is to decide one production sequence and the lot-size for each production. An example is given in Example5.1.

The flow shop problem is a special case of the multi-level capacitated lot-sizing problem (MLCLSP) which was introduced by Billington et al. (1983). The MLCLSP describes that several products are produced on *M* non-identical machines with limited capacities in periods. For each product, a unique assignment to a single

machine exists. Thus, we can interpret a product as the result of an operation. This operation is part of a process plan describing how to produce the item. The products or rather operations are interrelated through input–output relationships. In the flow shop case, products are produced in a set of serially arranged machines. Sarker and Yu (1996) presented two heuristics to obtain optimal schedule. These algorithms are used in deriving the optimal solution and in providing the production schedules. In Fandel and Stammen-Hegene (2006), a job shop problem, which is more general then flow shop, is considered. They construct a large bucket mathematical model with sequence-dependent setup time. However, it is not appropriate for realistic problems since too many binary variables exist. In Mohammadi et al. (2010), one exact formulation and five MIP-based heuristics have been derived and tested and they are all relied on successive resolutions of reduced size MIPs for lot-sizing in capacitated pure flow shop with sequence-dependent setups. For multi-level problems, a thorough survey is in Buschkühl et al. (2010).

Chapter 3 Mathematical Model and Solution Method

In this chapter we will present an extension of CLSP with sequence-dependent setup time and initial inventory. We provide a simple procedure which converts the problem to the simplified problem without initial inventory. We present a mathematical model, which is more general and complete than former models in the literatures. This model can solve small-size problems optimally. In terms of large-size problem, we provide a period-by-period heuristic, revised from Haase and Drexl (1995). The heuristic can find a high-quality solution for large-size problem in a short time. Finally, we use a small example to illustrate how the heuristic operates.

3.1 Problem Description, Notations and Assumptions

A lot-sizing and scheduling problem is considered with following features. The deterministic and dynamic demand is given for each product and each period. Demands are satisfied without shortage in the end of each period. Inventory cost is occurred when a product holding a period. The initial inventory is considered. A specific processing rate is given for each product. The finite and dynamic capacity is given for each period. The changeover between two different products occurs sequence-dependent setup time. Compared with the typical CLSP, setup cost is ignored. The reason is that setup costs is involved in worker's salary and lost opportunity costs. Being different from previous CLSP, the setup state can be kept up over periods and idle time. Furthermore, setup time in our model can be split up into two parts which are performed in the back and the front between the adjacent periods, respectively. In addition, the upper bound (storage limit), the lower bound (safety stock) and the shelf life of inventory are also considered in the mathematical model.

The aim is to determine the production sequence and the lot-size for each production, such that the total cost is minimized and demands are satisfied.

Indices we used are introduced from Table 3.1. Deterministic parameters and variables are introduced in Table 3.2 and Table 3.3, respectively.

Table 3.1. Indices

Index	Definition
i, j	The index of products, $i, j = 1, \dots, J$
t	The index of periods, $t = 1, \dots, T$
τ	The index of position in a sequence, $\tau = 1, \dots, K$

Table 3.2. Parameters

Parameters	Definition
J	Number of products
T	Number of periods
K	Number of products can be produced in a period
d_{jt}	Deterministic demand for product j in period t (quantity unit)
C_t	Available production capacity in period t (hour)
p_j	Processing rate of product j (quantity unit/hour)
s_{ij}	Setup time of a changeover from product i to product j (hour)
u^{st}	The upper bound of total setup time (hour)
u_j	The upper bound of inventory for product j (unit)
l_j	The lower bound of inventory for product j (unit)
v_j	The shelf life of inventory for product j (period)
h_j	Holding costs for product j (dollar/quantity unit/period)
I_{j0}	Initial inventory for product j (quantity unit)
z_{j0}	$z_{j0} = 1$, if the machine is setup for product j at the beginning ($z_{j0} = 0$, otherwise.)

Table 3.3. Variables

Variables	Definition
I_{jt}	Inventory of product j at the end of period t (quantity unit)
x_{jt}	Processing time of product j in period t (hour)
y_{ijt}	$y_{ijt} \geq 1$, if a setup occurs from product i to product j in period t ($y_{ijt} = 0$, otherwise.)
$z_{jt\tau}$	$z_{jt\tau} = 1$, if product j is τ^{th} produced product in period t ($z_{jt\tau} = 0$, otherwise.)
S_t	Setup time used in period t (hour)
α_t	Part of setup time, at the beginning of period t (hour)
β_t	Part of setup time, at the end of period t (hour)
e_t	Idle time in period t (hour)

Following assumptions are made for the problem:

1. Dynamic deterministic external demands for products are given and satisfied at the end of each period.
2. Inventory holding costs are computed based on the end of period inventory.
3. Finite and dynamic capacities for periods are given.
4. Backlogging is not allowed.
5. Sequence-dependent setup time occurs when there is machine setup and setup cost is not considered.
6. The setup state is kept up over periods and idle time.
7. At most K products can be produced in a period where $K \leq J$.
8. Inventory at the end of the planning horizon must be greater than or equal to the initial inventory, i.e. all the demand can be satisfied by the production.
9. Inventory in periods must be greater than or equal to the safety stock (lower bound) and less than the upper bound for each product.
10. Inventory will deteriorate if it exceeds the shelf life and it is not allowed.

3.2 A Simple Procedure for Initial Inventory Simplification

Initial inventory plays an essential part in the proposed problem, because the amount of initial inventory seriously associates whether the feasible solution exists or not. Before planning, initial inventory are usually set to be the remaining stock produced in the past, and they may not be all zero. In the practical production environment, the initial inventory may be re-planed once in a while (maybe a month). This time, the initial inventory is the current stock. In order to let the construction of model easier, we still have to assume initial inventory zero. Thus, we present a simple procedure which converts the original problem to a new one.

The idea of the procedure is to combine the demand of the final period with the initial inventory and then consume all the initial inventory. A high level code of the procedure of the simplification of initial inventory is given as follows:

Procedure_Initial_Inventory_Simplification
Begin
  For $j := 1, \ldots, J$ **do** $I_{j0} < \sum_t d_{jt}$
    $t := 1$;
    $d_{jT} := d_{jT} + I_{j0}$; // combine final demand and initial inventory
    Do Until $I_{j0} = 0$ // do until initial inventory equals 0
      If $I_{j0} > d_{jt}$ **then**
        $I_{j0} := I_{j0} - d_{jt}$; // use initial inventory to satisfy demands
        $d_{jt} := 0$; // reset all the satisfied demands as zero
      Else
        $d_{jt} := d_{jt} - I_{j0}$; // reset demands
      End If
      $t := t + 1$;
    End
  End
End

The following example is considered to show that how to deal with non-zero initial inventory.

Example 3.1. There is one product ($J = 1$) and four periods ($T = 4$) being planned.

The initial inventory is 50. In the following, we transform the problem into one with zero initial inventory. Demands and the process are shown in Table 3.4.

Table 3.4. A Transformation of Problem

step	initial inventory	demands of period				total demand
	I_{10}	d_{11}	d_{12}	d_{13}	d_{14}	
0	50	30	40	30	20	120
1	50	30	40	30	70	170
2	20	0	40	30	70	140
3	0	0	20	30	70	120

The total demand is 120 in the original problem in step 0. In step 1, the demand in period 4 increases to 70 by adding the initial inventory. In step 2, we start to consume initial inventory, using it to satisfy demand for the first period. In step 3, we use initial inventory to satisfy demand for the second period. At this step, the procedure stops due to the exhausting of initial inventory and create a new problem.

If the safety stock (lower bound of inventory) of some products is not zero, it should be deducted first from initial inventory. When the initial inventory becomes a new one, we then transform it by method mentioned above.

3.3 Mathematical Model

In this section we provide a general model with features include: upper bound of inventory and lower bound of inventory, shelf life, sequence-dependent setup time and cross-period setup time. The problem we considered can be formulated as a mixed-integer programming as follows:

$$\text{Minimize} \sum_{t=1}^{T} \sum_{j=1}^{J} h_j I_{jt}, \tag{3.1}$$

subject to

$$I_{j,t-1} + p_j x_{jt} - I_{jt} = d_{jt}, \qquad j = 1,\ldots,J, t = 1,\ldots,T, \tag{3.2}$$

$$\sum_{j=1}^{J} x_{jt} + S_t + e_t = C_t, \qquad t = 1,\ldots,T, \tag{3.3}$$

$$x_{jt} \le C_t \sum_{\tau=1}^{K} z_{jt\tau}, \qquad j = 1,\ldots,J, t = 1,\ldots,T, \tag{3.4}$$

$$\sum_{j=1}^{J} z_{jt\tau} = 1, \qquad t = 1,\ldots,T, \tau = 1,\ldots,K, \tag{3.5}$$

$$S_t = \sum_{i=1}^{J}\sum_{j=1}^{J} y_{ijt} s_{ij} + \alpha_t + \beta_t, \qquad t = 1,\ldots,T, \tag{3.6}$$

$$s_{ij}\left(z_{i,t-1,K} + z_{jt1} - 1 \right) \le \alpha_t + \beta_{t-1}, \qquad i, j = 1,\ldots,J, t = 2,\ldots,T, \tag{3.7}$$

$$z_{it\tau} + z_{jt,\tau+1} \le y_{ijt} + 1, \qquad i, j = 1,\ldots,J, t = 1,\ldots,T, \tau = 1,\ldots,K-1, \tag{3.8}$$

$$\sum_{t=1}^{T} S_t \le u^{st}, \tag{3.9}$$

$$I_{jt} \le u_j, \qquad j = 1,\ldots,J, t = 1,\ldots,T, \tag{3.10}$$

$$I_{jt} \ge l_j, \qquad j = 1,\ldots,J, t = 1,\ldots,T, \tag{3.11}$$

$$I_{jt} \le \sum_{\sigma=t}^{t+v_j} d_{j\sigma}, \qquad j = 1,\ldots,J, t = 1,\ldots,T, \tag{3.12}$$

$$z_{jt\tau} \in \{0,1\}, \qquad i, j = 1,\ldots,J, t = 1,\ldots,T, \tau = 1,\ldots,K, \tag{3.13}$$

$$\alpha_t, \beta_t, S_t, r_t \ge 0, \qquad t = 1,\ldots,T, \tag{3.14}$$

$$I_{jt}, x_{jt} \ge 0, \qquad j = 1,\ldots,J, t = 1,\ldots,T, \tag{3.15}$$

$$y_{ijt} \ge 0, \qquad i, j = 1,\ldots,J, t = 1,\ldots,T. \tag{3.16}$$

The objective function (3.1) is to minimize the total inventory cost without setup costs. Constraints (3.2) are typical inventory balance equations and constraints (3.3) are capacity restrictions. Constraints (3.4) ensure that as long as $x_{jt} > 0$, product j should be scheduled in period t. Constraints (3.5) mean that there is an exact one

product being scheduled at each position even with zero production. In the case, it is the preservation for setup state. Setup time for each period is computed by constraints (3.6). Constraints (3.7) divide setup time into two part, α and β, which are performed in two adjacent periods, if setup state are different in the end of last period and in the beginning of this period. Constraints (3.8) show that the indicator variable $y_{ijt} \geq 1$ if product i and product j are produced in sequence in period t. Constraint (3.9) is the restriction of total setup time for the whole planning horizon. Constraints (3.10) and (3.11) are inventory upper bound and lower bound constraints. In constraints (3.12), inventory cannot exceed the sum of all the demand in shelf life, or the inventory will deteriorate and it is not allowed. Finally, constraints (3.13) are binary restrictions and constraints (3.14) to (3.16) are non-negative constraints. The number of real variables, binary variables and constraints are summary in Table 3.5. We provide a small example to illustrate this model as follows.

Table 3.5. Complexity of the Model

Real variables	$4 \times T + 2 \times J \times T + J^2 \times T$
Binary variables	$J \times T \times K$
Constraints	$1 + 2 \times T + 5 \times J \times T + T \times K + J^2 \times T + J^2 \times T \times K$

Example 3.2. There are two products ($J = 2$) and three periods ($T = 3$) being scheduled. We assume that at most $K = 2$ products can be produced in a period. Available production time is 50 for all periods. Lower bound of inventory is zero; upper bound of inventory, shelf life and upper bound of setup time are assumed to be infinite. Demands, setup time and other known parameters are given by Table 3.6.

Table 3.6. Parameter Values

	d_{j1}	d_{j2}	d_{j3}	s_{1j}	s_{2j}	I_{j0}	p_j	h_j	z_{j0}
$j = 1$	45	0	40	*0*	10	0	1	4	1
$j = 2$	0	45	0	10	0	0	1	9	0

We formulated this example via the model and solved by using LINGO. Table 3.7 shows the optimal solution with objective value 0. However, note that the setup time cross period 1 and period 2, and this is not allowed in past models. It can be shown clearly in Figure 3.1.

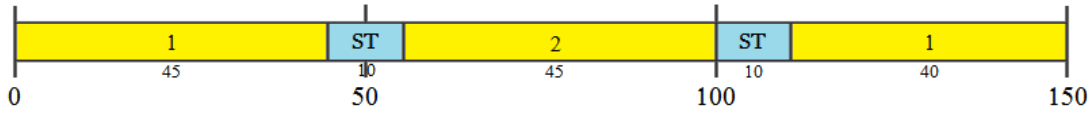


Figure 3.1. Gantt Charts of the Solution

Table 3.7. Optimal Solution

	$t = 1$	$t = 2$	$t = 3$
x_{1t}	45	0	40
x_{2t}	0	45	0
S_t	5	5	10
α_t	0	5	10
β_t	5	0	0
e_t	0	0	0
(z_{1t1}, z_{1t2})	(1, 1)	(0, 0)	(1, 1)
(z_{2t1}, z_{2t2})	(0, 0)	(1, 1)	(0, 0)

3.4 Solution Method

The solution method in this thesis comprises two stages. The purpose of the first stage is to generate a good feasible solution. In this stage we provide a backward method. Based on review of Chapter 2, it is a Period-by-period method. Subsequently, we refine this feasible solution in the second stage. By means of rearrangement and

exchange, we attempt to discover a solution which is superior to that obtained in first stage. For simplicity, we assume that there are no limitations of upper bound of inventory and shelf life.

3.4.1 The Backward Method

The notion of this heuristic method is to plan backward-oriented. This is a solution construction procedure, starting with nothing and finish with a complete solution. It will start from period T and finish in period 1. In each iteration a proper product will be selected to be produce. We estimate a measure for each product according to unsatisfied demand and setup time. And assign a probability, which is based on the proportion of the measure, to each product. Then we select a product at random. In the next iteration, if we select different products, corresponding setup time will be added into schedule. The whole process goes iteratively until obtain a feasible solution.

According to Haase and Drexl (1995), small bucket problems can be solved efficiently by a heuristic method named BAPLSP without the consideration of the setup time and initial inventory. The method will be modified and be improved here. We construct several criteria to diversify the estimation of priority measure and make the method more flexible to fit problems. Furthermore, we ignore old rules in BAPLSP which are designed for the PLSP, the small bucket problem. Then we create new rules to deal with large bucket problems and to integrate the sequence-dependent setup time and initial inventory for achieving high-quality feasible solutions.

Notations used in description are the same in Table 3.3, Table 3.4 and Table 3.5. In this method, instead of binary variables $z_{jt\tau}$, we introduce new variables $seq_t = (i_1, \dots, i_k, \dots, i_{N_t})$ to represent the producing sequence of N_t products in period t, where i_1, i_k and i_{N_t} are the first, the k-th and the last produced product, respectively.

The procedure will finally output a set of data, includes x_{jt}, α_t, β_t, r_t and seq_t, which can represent a production schedule. An example of the output is shown in Table 3.8 and a corresponding Gantt chart is in Figure 3.2. Note that, if there is any remaining capacity in this period, an idle will appear in the head of this period, since the schedule is constructed backward.

Table 3.8. A Schedule

	$t = 1$	$t = 2$	$t = 3$
x_{1t}	40	0	20
x_{2t}	0	40	15
α_t	0	10	0
β_t	5	0	0
e_t	5	0	5
seq_t	(1)	(2)	(2,1)

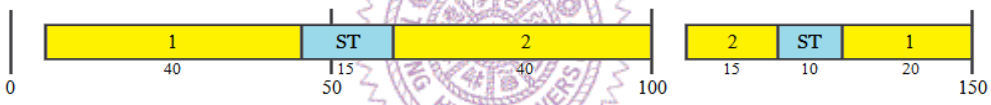


Figure 3.2. A Gantt Chart of the Schedule in Table 3.8

The backward method consists of four main steps: *Initialization*, *Selection*, *Setup* and *Production*. The Four steps will be introduced in detail in following description.

Initialization: There are several data that should be initialized in advance. Lot-sizes x_{jt} are set to be zero for all product j and period t in the beginning. Let D_{jt} be the cumulative unsatisfied demand in terms of capacity unit (hour) from period t to period T. The D_{jt} are determined by equations (3.17). Without loss of generality we set $D_{j0} = 0$ for all j.

$$D_{jt} := \max\left\{ \sum_{\sigma=t}^{T} \left(\frac{d_{j\sigma}}{p_j} - x_{j\sigma} \right), 0 \right\}, \quad j = 1,\ldots,J, t = 1,\ldots,T. \tag{3.17}$$

From (3.17) D_{j1} means total demand of product j. Let TD be the total unsatisfied

demand and computed by the following equation.

$$TD := \sum_{j=1}^{J} D_{j1} . \tag{3.18}$$

For each product j, let nd_{jt} be the next but last period which is less than or equal to t with unsatisfied demand and be determined by the following equation.

$$nd_{jt} := \begin{cases} 0 & \text{if } t = 0 \\ t & \text{if } d_{jt} > 0 , \quad j = 1,\ldots,J, t = 1,\ldots,T . \\ nd_{j,t-1} & \text{otherwise} \end{cases} \tag{3.19}$$

Moreover, $\tilde{C}_t$ denotes the cumulative capacity from period 1 to period t.

$$\tilde{C}_t := \sum_{\sigma=1}^{t} C_\sigma , \quad t = 1,\ldots,T . \tag{3.20}$$

The following example is used to illustrate the initialization.

Example 3.3. All the parameters are same to Example 3.2. The problem is initialized by using above method and the initialized data is shown in Table 3.9.

Table 3.9. Initial Data

	t	0	1	2	3
Original Problem	d_{1t}		45		40
	d_{2t}			45	
	C_t	0	50	50	50
Cumulative Demand	D_{1t}	0	85	40	40
	D_{2t}	0	45	45	0
Total Demand	TD		130		
Next-Demand Period	nd_{1t}	0	1	1	3
	nd_{2t}	0	0	2	2
Cumulative Capacity	$\tilde{C}_t$	0	50	100	150

Selection: Before introducing how to select products, we must explain the following things: If machine is setup for product j in period t, then product j will be scheduled until $D_{j\sigma} = 0$ ($\sigma < t$), i.e. all the demand of product j after period σ will be

satisfied. Then $t' = nd_{j,\sigma-1}$ is the next period with unsatisfied demand of product j. At this time, we have to decide to produce other product but occur setup time or "jump-back" from period σ to t' (stop running the machine for a short while but not occur setup time). A "jump-back" is infeasible if total demand in terms of capacity is more than the cumulative capacity in period σ. Nevertheless, different products $j' \neq j$ can be chosen to produce in period t only if $D_{j't} > 0$, if $D_{j't} = 0$ then $x_{j't}$ will be zero as well. Note that, changing setup state to a product with zero production and then changing again to another product is always not an optimal decision since setup times satisfy the triangle inequality.

We use a simple biased sampling method to decide what product should to be produced. This sampling is based on measures determined from a function which consists of unsatisfied demand and setup time, etc. The higher of the product's measure is, the much possibility it will be selected. These functions are designed on a basis of a fundamental concept: The more a product's unsatisfied demand, the great its measure should be, because the amount of unsatisfied demand indicates the product's urgency of being produced. Besides, since we do not want to spend too much time on setup, the shorter of a product's setup time is, the greater its measure should be.

Following functions are used to determine measures. The γ is a control parameter used to balance inventory and setup time. Based on the current setup state i, the measure of producing product j in period t is defined as follows:

Measure 1

$$r_{jt}(i) := \begin{cases} (1-\gamma)\cdot D_{jt} - \gamma \cdot s_{ji} & \text{if } (j \neq i) \wedge (D_{jt} > 0) \quad \text{case 1} \\ (1-\gamma)\cdot D_{j,nd_{j,t-1}} & \text{if } (j = i) \wedge (TD \leq \tilde{C}_{nd_{j,t-1}}) \quad \text{case 2}, \\ -\infty & \text{otherwise} \quad \text{case 3} \end{cases} \tag{3.21}$$

$$j = 1, \ldots, J, 0 \leq \gamma \leq 1.$$

Function (3.21) has three cases. Case 1 occurs if current setup state i is not product j and there exists unsatisfied demand in period t. Case 2 occurs if current

setup state is product j and a jump-back to the period $nd_{j,t-1}$ is feasible. Case 3 occurs if current setup state is not product j and $D_{jt} = 0$ in period t or a jump-back to the period $nd_{j,t-1}$ is infeasible. Note that, in our problem, setup costs are not considered. So if γ closes to one, we will expect that lot-sizes are relatively large and "jump-back" takes place. The schedule will be computed with a policy: Save setups. Thus the inventory cost (total cost) may be relatively high. On the contrary, if γ closes to zero, then we expect that lot-sizes are relatively small and no "jump-back" takes place. Thus, the inventory cost may be relatively low.

In order to make a precise decision of selection, we construct several criteria differ to Haase's to compute measures. As following:

Measure 2

$$r_{jt}(i) := \begin{cases} \dfrac{1-\gamma}{\gamma} \cdot \dfrac{D_{jt}}{s_{ji}} & \text{if } (j \neq i) \wedge (D_{jt} > 0) \\ \dfrac{1-\gamma}{\gamma} \cdot D_{j,nd_{j,t-1}} & \text{if } (j = i) \wedge (TD \leq \tilde{C}_{nd_{j,t-1}}), \\ -\infty & \text{otherwise} \end{cases} \tag{3.22}$$

$j = 1, \ldots, J, 0 \leq \gamma \leq 1.$

Measure 3

$$r_{jt}(i) := \begin{cases} (1-\gamma) \cdot \left(D_{jt} - \sum_{j' \neq j} D_{j't} \right) - \gamma \cdot s_{ji} & \text{if } (j \neq i) \wedge (D_{jt} > 0) \\ (1-\gamma) \cdot \left(D_{j,nd_{j,t-1}} - \sum_{j' \neq j} D_{j',nd_{j,t-1}+1} \right) & \text{if } (j = i) \wedge (TD \leq \tilde{C}_{nd_{j,t-1}}), \\ -\infty & \text{otherwise} \end{cases} \tag{3.23}$$

$j = 1, \ldots, J, 0 \leq \gamma \leq 1.$

Measure 4

$$r'_{jt}(i) := \begin{cases} r_{jt}(i) \cdot \dfrac{u^{st} - CST}{u^{st}} & \text{if } j \neq i \\ r_{jt}(i) \cdot \left(1 - \dfrac{u^{st} - CST}{u^{st}} \right) & \text{if } j = i \end{cases}, \tag{3.24}$$

$j = 1, \ldots, J,$ where CST is cumulative setup time in process.

Measure 1 is almost the same to Haase's which uses a subtraction to aggregate inventory and setups. In measure 2, instead of subtraction, we take a ratio to estimate.

Moreover, in measure 3, measure 1 is extended to involve a regret of the selection for other candidates. It will provide more information to make a decision of selection, and we expect that it will be better. In order to control the total setup time in process, we introduce measure 4. In measure 4, measures, which are obtained by measure 1, 2 and 3, are multiplied by a ratio which is a proportion of available time of setups. If there is no time remained for setups, then following changeovers will not be allowed.

For roulette wheel selection purpose, we use function (3.25) to normalize measures.

$$\rho_{jt}(i) := \begin{cases} 0 & \text{if } r_{jt}(i) = -\infty \\ \left(r_{jt}(i) - \bar{r}_{kt}(i) + \varepsilon \right)^{\delta} & \text{if } r_{jt}(i) > -\infty \end{cases}, \tag{3.25}$$

where $\bar{r}_{kt}(i) = \min\{ r_{kt}(i) \mid k = 1, \ldots, J \wedge r_{kt}(i) > -\infty \}$,
$j = 1, \ldots, J, \delta \geq 0, \varepsilon > 0$.

This normalization can transform every $r_{jt}(i)$ into $\rho_{jt}(i)$. Maximum measure that is transformed remains the greatest and vice versa. Nevertheless, the measure equals to zero after transformation under infeasible cases ($r_{jt}(i) = -\infty$). Apart from infeasible cases, every $\rho_{jt}(i)$ will be positive because $\varepsilon > 0$. The size of δ determines the scale of the measure. Note that, if all the $\rho_{jt}(i)$ equals zero, then product $j^* = 0$ will be selected, and it implies the machine is idle.

Setup: After deciding product j^* to be the next, if the machine is setup for product i, the setup time $s_{j^*,i}$ is needed to be arranged. If $s_{j^*,i} \geq C_t$ then $s_{j^*,i}$ will be split into α_t and β_{t-1} to cross from period $t-1$ to period t. α_t, β_{t-1} and C_t are updated by equations (3.26), (3.27) and (3.28), respectively. If $s_{j^*,i} < C_t$ then α_t and β_{t-1} are not updated and C_t are updated by equation (3.28). However, if the setup state does not need to be changed, i.e. $s_{j^*,i} = 0$, then nothing will be changed.

$$\alpha_t(j^*, i) := \min\{ C_t, s_{j^*,i} \}, \tag{3.26}$$

$$\beta_{t-1}(\, j^*, i\,) := \max\{\, s_{j^*,i} - C_t, 0\,\}, \tag{3.27}$$

$$C_t(\, j^*, i\,) := \min\{\, C_t - s_{j^*,i}, 0\,\}. \tag{3.28}$$

Production: Product j^* will be scheduled with lot-size $x_{j^*,t} := \min\{\, C_t, D_{j^*,t}\,\}$ until $D_{j^*,t} = 0$, and then j^* will be inserted into the first position of seq_t. Subsequently, $D_{j^*,t}$, TD and $\tilde{C}_t$ are updated by equations (3.17), (3.18) and (3.20), respectively. If all of the demand are satisfied, i.e. $TD = 0$, it is the end of program. Otherwise, it goes back to *selection* to re-select a product to produce.

In the progress of the program, all of setup time we used will be accumulated. If the accumulated setup time exceeds the upper bound, or unsatisfied demands still exist when the scheduling has already moved forward to the starting point, this search will be deemed as a failure. Afterwards, the program will automatically re-start searching from initialization. If the given problem is not that difficult and has a feasible solution, this program will obtain a feasible solution once executed. The results of each execution may be different. To see how the backward method works, a numerical example is given in section 3.5.

A high level code of the procedure of the backward method is given in Appendix I and a flowchart is given in Figure 3.3.

3.4.2 The Improvement Procedure

Even though the problem that we are dealing with is difficult, it is urged to be solved practically. There are no specific methods that can be ensured to get the optimal solution when solving this kind of large problem in practical until now. Therefore, we can assume that any solutions that we have got can be improved in certain ways. The improvement procedure searches better solutions from a feasible initial solution. Whether the solution is solved by the method that we provided before,

the backward method, executives can get a better solution from implementing the improvement procedure.

The implementation procedure includes two steps: Step 1. *Sequencing*: to find the optimal sequence of the produced products in each period such as to minimize the total setup time. Step 2. *Shifting*: To do "right shift" for each production in a feasible solution. The program will go through twice, i.e. step 1- step 2- step 1- step 2, for improving the solution.

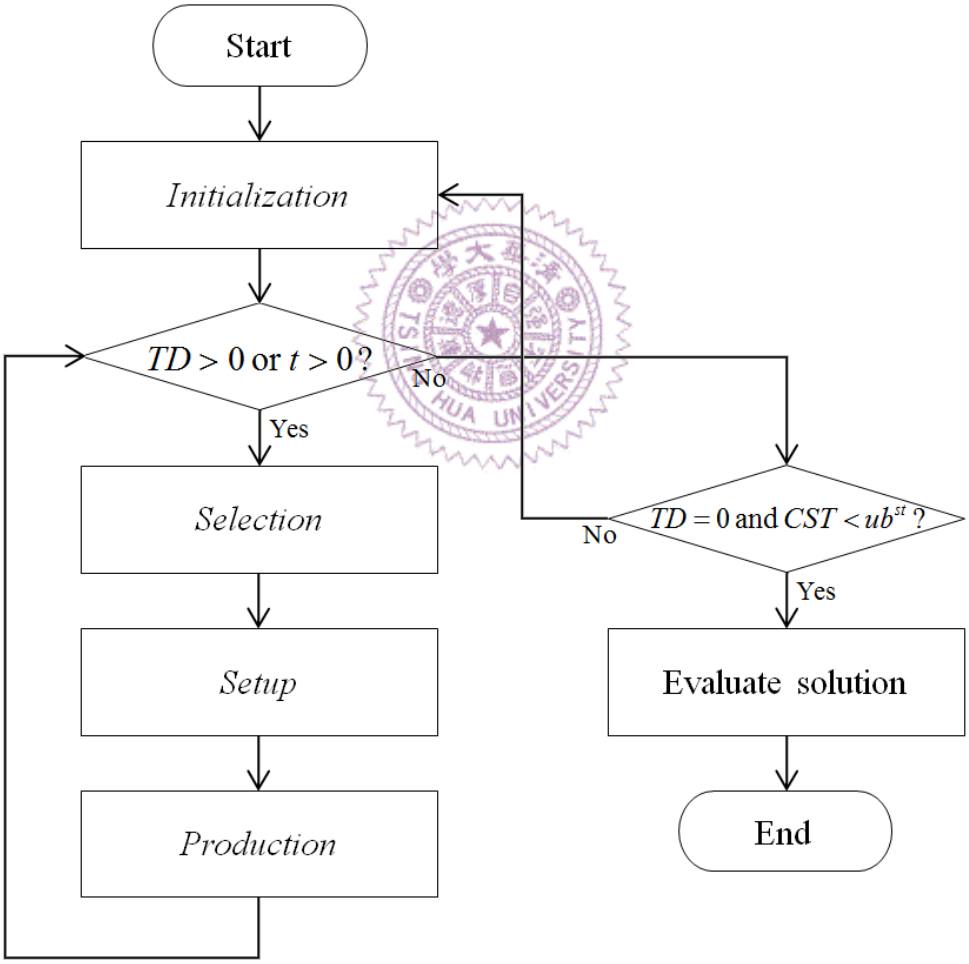


Figure 3.3. The Backward Method

Sequencing: From the known feasible initial solution, the production sequence in each period can be obtained. The aim to this step is to rearrange the production

sequence in order to save setup time. For each sequence, the first product and the last product is fixed to keep the linked relation between sequences. Therefore, this step affects the problem which plans more than three products. Products in the middle of sequence are rearranged to save the total setup time. The setup time we saved will become idle time, and then it is used to saving costs in the next step. We can regard the sequencing problem as a traveling salesman problem. Even though it is a difficult problem, there are no many products being produced in a same period and the size of the problem is small, we can solve it by using dynamic programming. Details are explained in the following:

The sequencing problem is given as $Q = (I, i_F, i_L, S)$ where $I = \{1, 2, \ldots, n\}$ is the set of n products, $i_F \in I$ is the first produced product, $i_L \in I$ is the last produced product and S is the setup time matrix with $\forall i, j \in I, i \neq j, s_{ij} \geq 0$. The problem is to find the production sequence with minimum total setup time, that starting in i_F and goes through all n products only once and stopping in i_L.

Let $T(I, i_F, i_L)$ be the minimum total setup time of production sequence that starts at i_F, produce all products in I and ends at i_L. A recursive definition of the optimal solution is as follows:

$$T(I, i_F, i_L) = \begin{cases} 0 & \text{if } |I| = 1 \\ s_{i_F, i_L} & \text{if } |I| = 2 \\ \min_{u,v \in I - i_F - i_L} \left\{ T(I - i_F - i_L, u, v) + s_{i_F, u} + s_{v, i_L} \right\} & \text{if } |I| \geq 3 \end{cases}. \quad (3.29)$$

A high level code of the *Sequencing* step is given in Appendix II.

Shifting: In this step the inventory cost is expected to be reduced. As long as there is no shortage occurs, the inventory cost appears less when the products are produced in a later time. We assume that there is no setup cost. If there is idle time in the schedule, the products that are produced in the early time can be shifted to the idle time. Therefore, the inventory cost will be saved.

The procedure will check all the productions in the solution to decide how much quantity can be moved later. Products are ranked by holding cost from high to low and the product with the highest holding cost will be checked first. Let t' be the target period which with idle time. The t' is defined as the last period with idle time at first. Productions at period t' and after period t' are not checked. Productions before period t' are checked from period $t'-1$ to period 1. In each iteration, the production at period t is checked for be moved to period t', $t'-1, \ldots, t+1$. For example, period 5 is the last period with idle time and a production at period 2 is checked. The program checks moves from period 2 to period 5, 4 and 3. After checking, under the condition of no shortage, the production quantity that can be moved will be moved later. Meanwhile, it appears a usable new idle time, and then checks former productions.

Suppose now a production of product k at period t is being checked and there is an idle in period t'. The decision of shifting is made by two values q_{kt} and $q'_{kt'}$ which are determined by equations (3.29) and (3.30).

$$q_{kt} := \min\{ x_{kt}, CX_{kt} - CD_{kt} \}, \tag{3.29}$$

$$q'_{kt'} := \begin{cases} \min\{ e_{t'}, CD'_{kt'} - CX'_{kt'} \} & \text{if } x_{kt'} > 0 \\ \min\{ e_{t'} - AS, CD'_{kt'} - CX'_{kt'} \} & \text{if } x_{kt'} = 0 \end{cases}. \tag{3.30}$$

CX_{kt} = the cumulative processing time from period 1 to t.
CD_{kt} = the cumulative demand in terms of time from period 1 to t.
CX'_{kt} = the cumulative processing time from period t to T.
CD'_{kt} = the cumulative demand in terms of time from period t to T.
AS = the additional setup time for inserting the production.

The q_{kt} denotes the available time to do the right shift from period t. It is the minimum value of the processing time in period t and the additional processing time from the beginning to period t. Similarly, the $q'_{kt'}$ denotes the available time that can be shifted into period t' ($>t$). If product k is produced in period t', the target period, $q'_{kt'}$ is the minimum of idle time in period t' and the total unsatisfied demand time from period t' to period T. Otherwise, product k is not produced in period t', the

procedure will check each position and find a position to insert the production of product k such that minimize the additional setup time (AS). Thus, the idle time in period t' should be deducted by the additional setup time. And then $q'_{kt'}$ is determined by the minimum of idle time and total unsatisfied demand.

The right shift will be executed with the lot-size min{ q_{kt}, $q'_{kt'}$ }. The procedure continues checking the former period until all products and productions are checked. A high level code of the *Shifting* step is given in Appendix II.

3.5 Numerical Example

We use an example to illustrate the backward method and the Improvement procedure. There are three products ($J = 3$) and five periods ($T = 5$) being planned. Available production time is 50 for all periods. Lower bound of inventory is zero; upper bound of inventory, shelf life and upper bound of setup time are assumed to be infinite. Demands, setup time and other known parameters are given by Table 3.10. The measure formula we used is *measure 1*. We will show that how to obtain a feasible schedule by the backward method, and then try to refine it via the improvement procedure.

Table 3.10. Parameters

	d_{j1}	d_{j2}	d_{j3}	d_{j4}	d_{j5}	s_{1j}	s_{2j}	s_{3j}	I_{j0}	p_j	h_j
$j = 1$	20	20	0	10	0	0	10	5	0	1	4
$j = 2$	10	10	30	0	20	10	0	10	0	1	3
$j = 3$	0	0	20	10	10	5	10	0	0	1	6

First, for the backward method an initialize operation is executed. The initialized data is shown in Table 3.11 and we set $\gamma = 0.5$, $\delta = 1$ and $\varepsilon = 1$, current setup state $i = 0$ (zero setup time from product '0' to each product).

Table 3.11. Initial Data

	t	0	1	2	3	4	5
Original Problem	d_{1t}		20	20	0	10	0
	d_{2t}		10	10	30	0	20
	d_{3t}		0	0	20	10	10
	C_t	0	50	50	50	50	50
Cumulative Demand	D_{1t}	0	50	30	10	10	0
	D_{2t}	0	70	60	50	20	20
	D_{3t}	0	40	40	40	20	10
Total Demand	TD		160				
Next-Demand Period	nd_{1t}	0	1	2	2	4	4
	nd_{2t}	0	1	2	3	3	5
	nd_{3t}	0	0	0	3	4	5
Cumulative Capacity	$\tilde{C}_t$	0	50	100	150	200	250

Iteration 1

Selection: The measures for each product at period $t := T\,(=5)$ are determined as follows:

$r_{15}(0) = -\infty \quad \rightarrow \quad \rho_{15}(0) = 0,$

$r_{25}(0) = 0.5 \times 20 - 0.5 \times 0 = 10 \quad \rightarrow \quad \rho_{25}(0) = 10 - 5 + 1 = 6,$

$r_{35}(0) = 0.5 \times 10 - 0.5 \times 0 = 5 \quad \rightarrow \quad \rho_{35}(0) = 5 - 5 + 1 = 1.$

Let P_{jt} denotes the probability to select product j at period t. Thus $P_{15} = 0$, $P_{25} = 6/7$ and $P_{35} = 1/7$. Product 2 is selected at first.

Setup: There is no setup time between product 2 and product '0'.

Production: The lot-size of product 2 at period 5 is $x_{25} = \min\{50, 20\} = 20$. The capacity is updated: $C_5 := 50 - 20 = 30$. The total demand is updated: $TD := 160 - 20 = 140$. The cumulative demands of product 2 are updated in the following table. Product 2 is added into the production sequence of period 5: $seq_5 := (2)$.

Table 3.12. Cumulative Demands of Product 2

	t	0	1	2	3	4	5
Cumulative Demand	D_{2t}	0	50	40	30	0	0

Figure 3.4. Gantt Chart

Iteration 2

Selection: Based on setup state $i = 2$, the measures for each product at period $t = 5$ are determined as follows:

$r_{15}(2) = -\infty \quad \rightarrow \quad \rho_{15}(2) = 0,$

$r_{25}(2) = -\infty \quad \rightarrow \quad \rho_{25}(2) = 0,$

$r_{35}(2) = 0.5 \times 10 - 0.5 \times 10 = 0 \quad \rightarrow \quad \rho_{35}(2) = 1.$

Product 3 is selected with probability one.

Setup: A setup time $s_{32} = 10$ is scheduled, and the capacity is updated: $C_5 := 30 - 10 = 20$.

Production: The lot-size of product 3 at period 5 is $x_{35} = \min\{ 20, 10 \} = 10$. The capacity is updated: $C_5 := 20 - 10 = 10$. The total demand is updated: $TD := 140 - 10 = 130$. The cumulative demands of product 3 are updated in the following table. Product 3 is added into the head of production sequence of period 5: $seq_5 := (3, 2)$.

Table 3.13. Cumulative Demands of Product 3

	t	0	1	2	3	4	5
Cumulative Demand	D_{3t}	0	30	30	30	10	0

Figure 3.5. Gantt Chart

Iteration 3

Selection: Based on setup state $i = 3$, the measures for each product at period $t = 5$ are determined as follows:

$r_{15}(2) = -\infty \quad \rightarrow \quad \rho_{15}(2) = 0,$

$r_{25}(2) = -\infty \quad \rightarrow \quad \rho_{25}(2) = 0,$

$r_{35}(2) = 0.5 \times 10 - 0.5 \times 0 = 5 \quad \rightarrow \quad \rho_{35}(2) = 1.$

Product 3 is selected with probability one and which force a jump-back to period 4.

Setup: Zero setup time $s_{33} = 0$ is scheduled.

Production: The jump-back is from period 5 to period 4. The idle time is updated: $e_5 := C_5\ (=10)$ and $t := 4$. The lot-size of product 3 at period 4 is $x_{34} = \min\{50, 10\} = 10$. The capacity is updated: $C_4 := 50 - 10 = 40$. The total demand is updated: $TD := 130 - 10 = 120$. The cumulative demands of product 3 are updated in the following table. Product 3 is added into the head of production sequence of period 4: $seq_4 := (3)$.

Table 3.14. Cumulative Demands of Product 3

	t	0	1	2	3	4	5
Cumulative Demand	D_{3t}	0	20	20	20	0	0

Figure 3.6. Gantt Chart

Iteration 4

Selection: Based on setup state $i = 3$, the measures for each product at period $t = 4$ are determined as follows:

$r_{14}(3) = 0.5 \times 10 - 0.5 \times 5 = 2.5 \quad \rightarrow \quad \rho_{14}(3) = 1,$

$r_{24}(3) = -\infty \quad \rightarrow \quad \rho_{24}(3) = 0,$

$r_{34}(3) = 0.5 \times 20 - 0.5 \times 0 = 10 \quad \rightarrow \quad \rho_{34}(3) = 8.5.$

Let P_{jt} denotes the probability to select product j at period t. Thus $P_{14} = 1/9.5$, $P_{24} = 0/9.5$ and $P_{34} = 8.5/9.5$. Product 3 is selected and which force a jump-back to period 3.

Setup: Zero setup time $s_{33} = 0$ is scheduled.

Production: The jump-back is from period 4 to period 3. The idle time is updated: $e_4 := C_4\ (=40)$ and $t := 3$. The lot-size of product 3 at period 3 is $x_{33} = \min\{50, 20\} = 20$. The capacity is updated: $C_3 := 50 - 20 = 30$. The total demand is updated: $TD := 120 - 20 = 100$. The cumulative demands of product 3 are updated in the following table. Product 3 is added into the head of production sequence of period 3: $seq_3 := (3)$.

Table 3.15. Cumulative Demands of Product 3

	t	0	1	2	3	4	5
Cumulative Demand	D_{3t}	0	0	0	0	0	0

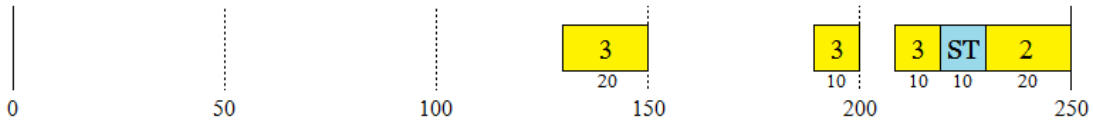


Figure 3.7. Gantt Chart

Following iterations are omitted and a step by step data of the backward method is given in Table 3.16. In summary, at iteration 3 product 3 is selected which forces a jump-back to period 4. At iteration 4 product 3 is selected again, which forces a jump-back to period 3; and so on. Gantt charts for each iteration are given in Figure 3.8 and a feasible solution is obtained finally given in Table 3.17 with total cost 150.

Table 3.16. Step by Step Data of the Backward Method

I	t	i	$r_{1t}(i)$	$r_{2t}(i)$	$r_{3t}(i)$	$\rho_{1t}(i)$	$\rho_{2t}(i)$	$\rho_{3t}(i)$	j^*	C_t	$D_{j^*,t}$	$x_{j^*,t}$	TD	e_t	seq_t
1	5	0	$-\infty$	10	5	0	6	1	2	50	20	20	140		(2)
2	5	2	$-\infty$	$-\infty$	0	0	0	1	3	30	10	10	130		(3,2)
3	5	3	$-\infty$	$-\infty$	5	0	0	1	3					10	
	4								3	50	10	10	120		(3)
4	4	3	2.5	$-\infty$	10	1	0	8.5	3					40	
	3								3	50	20	20	100		(3)
5	3	3	2.5	10	$-\infty$	1	8.5	0	2	30	30	20	80		(2,3)
	2								2	50	20	20	60		(2)
6	2	2	10	$-\infty$	$-\infty$	1	0	0	1	30	40	20	40		(1,2)
	1								1	50	30	30	10		(1)
7	1	1	$-\infty$	0	$-\infty$	0	1	0	2	20	10	10	0		(2,1)

Column I records iterations.

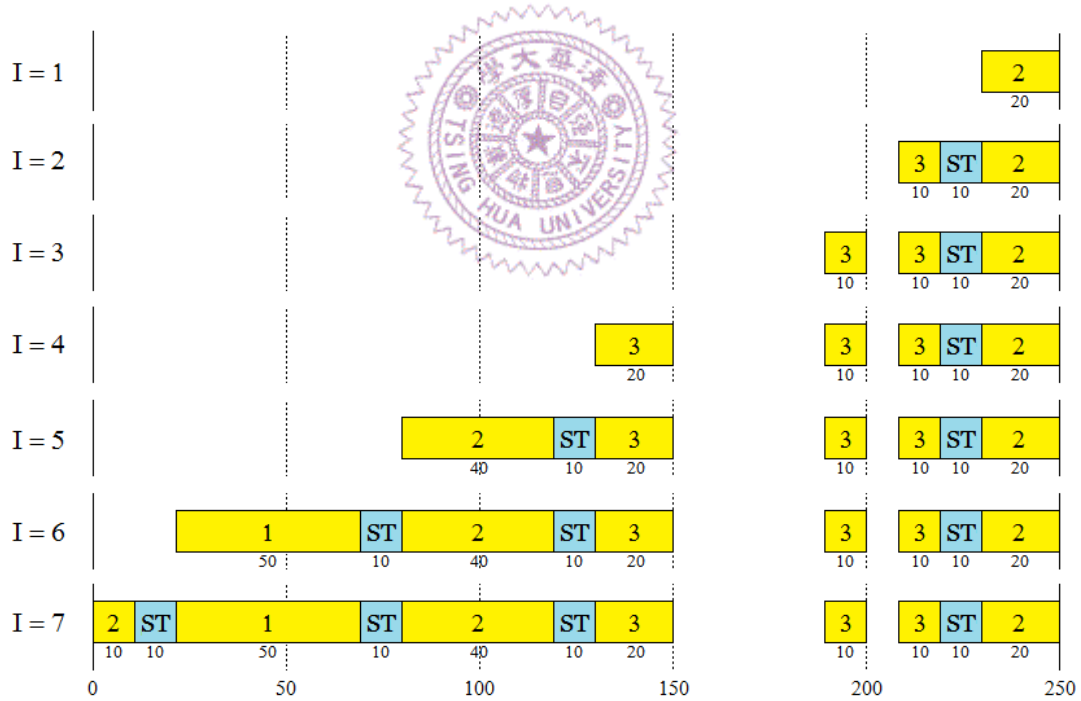


Figure 3.8. Gantt Charts for each Iteration

Table 3.17. A Feasible Solution

	$t=1$	$t=2$	$t=3$	$t=4$	$t=5$
x_{1t}	30	20	0	0	0
x_{2t}	10	20	20	0	20
x_{3t}	0	0	20	10	10
α_t	0	0	0	0	0
β_t	0	0	0	0	0
e_t	0	0	0	40	10
seq_t	(2,1)	(1,2)	(2,3)	(3)	(3,2)

Subsequently, we try to refine the obtained solution by using the improvement procedure.

Sequencing: This step is ignored since there are only three (less than four) products in the problem.

Shifting: Products are checked in the order of non-increasing holding cost: 3-1-2.

Iteration 1

Let t' := 5, which is the last period with idle time. The production of product 3 at period 4 is checked (k = 3, t = 4). q_{34} and q'_{35} are determined as follows:

q_{34} = min{ 0, 0 } = 0,

q'_{35} = min{ 10, 0 } = 0.

The quantity for moving is min{ q_{34}, q'_{35} } = 0, therefore nothing is moved in this iteration.

Iteration 2

Let t' := 5. The production of product 3 at period 3 is checked (k = 3, t = 3). q_{33} and q'_{35} are determined as follows:

q_{33} = min{ 0, 0 } = 0,

q'_{35} = min{ 10, 0 } = 0.

The quantity for moving is min{ q_{33}, q'_{35} } = 0, therefore nothing is moved.

Let $t' := 4$, which is the previous period with idle time. q'_{34} is determined as follows:

$q'_{34} = \min\{ 40, 0 \} = 0.$

The quantity for moving is $\min\{ q_{33}, q'_{34} \} = 0$, therefore nothing is moved.

Iteration 3

Let $t' := 5$. The production of product 1 at period 2 is checked ($k = 1$, $t = 2$). q_{12} and q'_{15} are determined as follows:

$q_{12} = \min\{ 20, 10 \} = 10,$

$q'_{15} = \min\{ 0, 0 \} = 0.$

The quantity for moving is $\min\{ q_{12}, q'_{15} \} = 0$, therefore nothing is moved.

We then set $t' := 4$, which is the previous period with idle time. q'_{15} is determined as follows:

$q'_{14} = \min\{ 30, 10 \} = 10.$

The quantity for moving is $\min\{ q_{12}, q'_{14} \} = 10$, therefore 10 units of product 1 is right shifted from period 2 to period 4.

Following iterations are omitted and a step by step data of the shifting is given in Table 3.18 and corresponding Gantt charts are given in Figure 3.9. In summary, there are two improvements in this execution of shifting. A lot with 10 units of product 1 is right shifted from period 2 to period 4, and then a lot with 10 units of product 1 is shifted from period 1 to period 2. The total cost is reduced to 30 from 150. The second run is also omitted because there is no improvement. This example is completed and the solution is given in Table 3.19.

Table 3.18. Step by Step Data of the Shifting

I	j	t	t'	q_{jt}	$q'_{jt'}$	$\min\{q_{jt}, q'_{jt'}\}$	Movement
1	3	4	5	0	0	0	
2	3	3	5	0	0	0	
3	1	2	5	10	0	0	
	1	2	4	10	10	10	Move 10 unit of product 1 from period 2 to period 4
	1	2	3	0	0	0	
4	1	1	4	10	0	0	
	1	1	3	10	0	0	
	1	1	2	10	10	10	Move 10 unit of product 1 from period 1 to period 2
5	2	3	5	0	0	0	
	2	3	4	0	0	0	
6	2	2	5	10	0	0	
	2	2	4	10	0	0	
	2	2	3	10	0	0	
7	2	1	5	0	0	0	
	2	1	4	0	0	0	
	2	1	3	0	10	0	
	2	1	2	0	0	0	

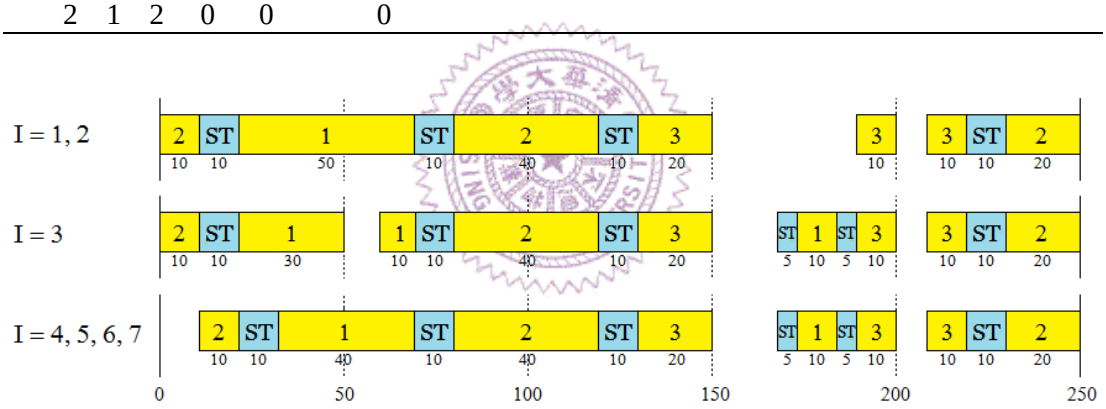


Figure 3.9. Gantt Charts for each Iteration of the Shifting

Table 3.19. The Feasible Solution

	$t=1$	$t=2$	$t=3$	$t=4$	$t=5$
x_{1t}	20	20	0	10	0
x_{2t}	10	20	20	0	20
x_{3t}	0	0	20	10	10
α_t	0	0	0	5	0
β_t	0	0	0	0	0
e_t	10	0	0	25	5
seq_t	(2,1)	(1,2)	(2,3)	(1,3)	(3,2)

3.6 Summary

In this chapter we provide a simple procedure to convert the problem into the one without initial inventory. We then present a mixed-integer programming formulation for the proposed problem. For large-scale problems, we also provide a two-stage heuristic which can find a feasible schedule in a short time.

The program of this thesis is implemented by Excel VBA, which has several advantages such as graphical user interface, easy to use and easy to access. Via Excel, the output of the program is no longer a lot of numbers, but to use understandable figures to represent a production schedule. A Gantt chart is useful way to specify what time to produce and how long of the production. A Bar chart is also used to show the comparison of the quantities of demands, inventories and productions. In manufacturing environment, in spite of a good feasible schedule is obtained, it still requires the management to make some appropriate changes to meet variant situations. In the program, the schedule can be modified by user and then reevaluated in a short time. An output of the program for the example in section 3.5 is given in Figure 3.10 and Figure 3.11.

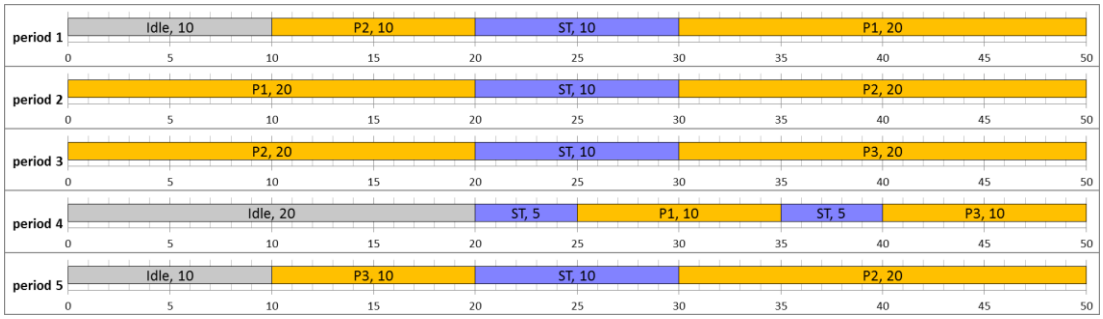


Figure 3.10. Gantt Charts

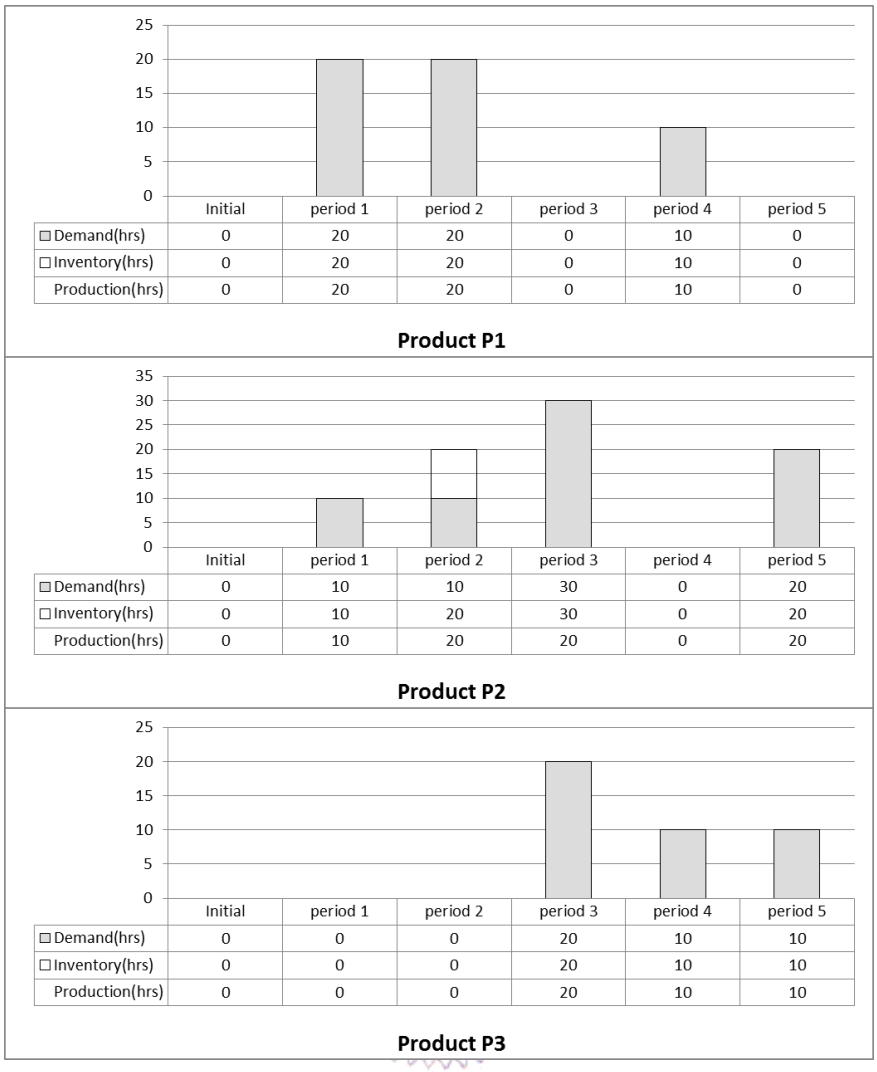


Figure 3.11. Bar Charts

3.7 Computational Experiment

We use an Excel-VBA implementation to test the two-stage heuristic on a PC with Intel Dual-Quad-Core with 2.83 GHz running under Windows 7 SP1.

3.7.1 Test Problems

The difficulty of a lot-sizing and scheduling problem may depend on several features. In order to explore the performance of the proposed method applied to different problems, we generate test problems by using the following settings:

(1) Number of products (J)

Let $J \in \{ 5, 15 \}$, two of number of products will be tested.

(2) Number of periods (T)

Let $T \in \{ 26, 52 \}$, two of number of periods will be tested. Note that a period means a week, so 26 and 52 periods means a half of a year and a whole year.

(3) Processing rate (p_j)

Let p_j be chosen out of the interval [1, 10] with uniform distribution for each product j processed. The unit of processing rate is a quantity unit per hour.

(4) Holding costs (h_j)

Let h_j be chosen out of the interval [1, 10] with uniform distribution for each product j. The unit of holding costs is one dollar per period.

(5) Setup times (s_{ij})

Let s_{ij} be chosen out of the interval [8, 16] with uniform distribution and $s_{ii} = 0$ for each pair of product i and j ($\neq i$). The setup time is in hours.

(6) Capacity (C_t)

For each period t, C_t is determined from a general distribution as follows: C_t = 7, 6, 5 and 4 with probability 0.4, 0.3, 0.2 and 0.1, respectively. The capacity is in 24-hours days.

(7) The upper bound of setup times (u^{st})

Let u^{st} be total capacity × V, where $V \in \{ 0.1, 0.2, 0.3 \}$. Three of upper bounds of setup times will be tested.

(8) Capacity utilization (U)

Let $U \in \{ 0.4, 0.6, 0.8 \}$, three of utilization will be tested. Note that, the utilization of capacity does not include the consuming of setup times. Hence, a value U = 0.6 actually means that on average the utilization of capacity by production and setup actions is greater than 60%.

(9) Fill rate (α)

The fill rate α is a proportion of non-zero values in the demand matrix. Let α be

chosen out of the interval [0.6, 0.8] with uniform distribution.

(10) Demand (d_{jt})

It needs several steps to generate. First, we generate a demand matrix with J products and T periods. Each entry is set to be one with probability α and be zero with probability ($1 - \alpha$). We then assign a uniform value e_{jt} in (0, 1] to each non-zero entry. Subsequently, d_{jt} (in quantity units) is generated as follows:

$$d_{jt} = \frac{e_{jt}}{\sum_i e_{it}} \times U \times C_t \times p_j, \ \ j = 1, \ldots, J, \ t = 1, \ldots, T \ . \tag{6.1}$$

(11) Initial inventory (I_{j0})

For each product j, the initial inventory I_{j0} is set to be J percent of total demand of product j, that is, $I_{j0} = \frac{J}{100} \times \sum_t d_{jt}, \ \ j = 1, \ldots, J$. Note, in proposed problem, the initial inventory is not the additional capacity. It also needs to return in the end of planning horizon.

Let $TP(J, T, U, V ; n)$ be an algorithm which generate n of lot-sizing and scheduling problem with J products, T periods, and the capacity utilization is U and the upper bound of setup time is total capacity × V. Other parameters are generated by using above rules.

For each parameterization of $TP(J, T, U, V ; n)$ we generate n = 5 test problems each. There are 36 sets of parameters and a total of 180 test problems. Let P001 to P180 be the problems. (cf. Table 3.20.)

Table 3.20. Test Problems

Set	Problems	*J*	*T*	*U*	*V*	*n*
1	P001 to P005	5	26	0.4	0.1	5
2	P006 to P010	5	26	0.4	0.2	5
3	P011 to P015	5	26	0.4	0.3	5
4	P016 to P020	5	26	0.6	0.1	5
5	P021 to P025	5	26	0.6	0.2	5
6	P026 to P030	5	26	0.6	0.3	5
7	P031 to P035	5	26	0.8	0.1	5
8	P036 to P040	5	26	0.8	0.2	5
9	P041 to P045	5	26	0.8	0.3	5
10	P046 to P050	5	52	0.4	0.1	5
11	P051 to P055	5	52	0.4	0.2	5
12	P056 to P060	5	52	0.4	0.3	5
13	P061 to P065	5	52	0.6	0.1	5
14	P066 to P070	5	52	0.6	0.2	5
15	P071 to P075	5	52	0.6	0.3	5
16	P076 to P080	5	52	0.8	0.1	5
17	P081 to P085	5	52	0.8	0.2	5
18	P086 to P090	5	52	0.8	0.3	5
19	P091 to P095	15	26	0.4	0.1	5
20	P096 to P100	15	26	0.4	0.2	5
21	P101 to P105	15	26	0.4	0.3	5
22	P106 to P110	15	26	0.6	0.1	5
23	P111 to P115	15	26	0.6	0.2	5
24	P116 to P120	15	26	0.6	0.3	5
25	P121 to P125	15	26	0.8	0.1	5
26	P126 to P130	15	26	0.8	0.2	5
27	P131 to P135	15	26	0.8	0.3	5
28	P136 to P140	15	52	0.4	0.1	5
29	P141 to P145	15	52	0.4	0.2	5
30	P146 to P150	15	52	0.4	0.3	5
31	P151 to P155	15	52	0.6	0.1	5
32	P156 to P160	15	52	0.6	0.2	5
33	P161 to P165	15	52	0.6	0.3	5
34	P166 to P170	15	52	0.8	0.1	5
35	P171 to P175	15	52	0.8	0.2	5
36	P176 to P180	15	52	0.8	0.3	5

3.7.2 Computational Experiments

We will test the propose method with different selection measures. Algorithms with each measure will be compared with an algorithm which randomly selects a produced product in every selection step. The computational results are shown in Table 3.21 and Table 3.22. The control parameter γ = 0.2, δ = 1 and ε =1 are determined by an advance experiment.

Let A0 be the algorithm which selects products randomly. Let A1, A2 and A3 be

the algorithm which uses measure 1, 2 and 3 respectively. Let A4, A5 and A6 be the algorithm which uses measure 1, 2 and 3 combined the measure 4, respectively. The notation in the tables is as defined in the following:

Z_{A}^{N}: Objective function value of the best solution which has been found within N executions of algorithm A. $Z_{\mathrm{A}}^{N} = \infty$ if there is no feasible solution obtained.

$$\Delta Z_{\mathrm{A}}^{N} = \begin{cases} \dfrac{Z_{\mathrm{A0}}^{N} - Z_{\mathrm{A}}^{N}}{Z_{\mathrm{A0}}^{N}} & \text{if } Z_{\mathrm{A}}^{N} < Z_{\mathrm{A0}}^{N} \text{ and } Z_{\mathrm{A0}}^{N} < \infty \\ 0 & \text{otherwise} \end{cases}.$$

$$\Delta Z_{\mathrm{A}} = \begin{cases} \dfrac{Z_{\mathrm{A}}^{1000} - Z_{\mathrm{A4}}^{1000000}}{Z_{\mathrm{A4}}^{1000000}} & \text{if } Z_{\mathrm{A}}^{1000} < \infty \text{ and } Z_{\mathrm{A4}}^{1000000} < \infty \\ 1 & \text{otherwise} \end{cases}.$$

Numbers in Table 3.21 are mean deviations from the random search (Algorithm A0). A high value means a low cost. A zero value means the cost is higher than the one obtained by using the random search or a feasible solution has not yet been obtained. Implications contained in the table are summarized below:

1. The numbers of P001 to P090 are almost greater than the numbers of P091 to P180. We can know that number of products is an important factor for the difficulty of the scheduling problem but the number of periods is not.
2. No feasible solution of problems with J = 15 and V = 0.1 has been found for the reason that setup time allowed may be too few to do changeovers for many products. Feasible solutions have been found for all the problems with V = 0.2 and 0.3.
3. In general, algorithm A4, which combines measure 1 and measure 4, has the best performance of all types of problems and Algorithm A1 is secondary.
4. However, for problem sets 2,3,5,6,8,9,11,12,14,15,17 and 18 (problems

with $J = 5$ and $V \geq 0.2$), algorithm A3 and A6 are better than algorithm A1 and A4. Therefore, if capacity is sufficient, algorithm A3 and A6 are alternative options.

Table 3.21. Average Deviation

Set	Problems	$\Delta\bar{Z}_{A1}^{1000}$	$\Delta\bar{Z}_{A2}^{1000}$	$\Delta\bar{Z}_{A3}^{1000}$	$\Delta\bar{Z}_{A4}^{1000}$	$\Delta\bar{Z}_{A5}^{1000}$	$\Delta\bar{Z}_{A6}^{1000}$
1	P001 to P005	0.43	0	0	0.37	0	0
2	P006 to P010	0.53	0.07	0.68	0.53	0.13	0.69
3	P011 to P015	0.8	0.81	0.81	0.8	0.81	0.81
4	P016 to P020	0.32	0	0	0.36	0	0
5	P021 to P025	0.55	0.24	0.67	0.55	0.14	0.68
6	P026 to P030	0.79	0.8	0.8	0.8	0.8	0.8
7	P031 to P035	0.32	0	0.16	0.29	0	0.17
8	P036 to P040	0.47	0.58	0.53	0.52	0.55	0.54
9	P041 to P045	0.44	0.54	0.54	0.49	0.53	0.52
10	P046 to P050	0.35	0	0	0.35	0	0
11	P051 to P055	0.53	0	0.72	0.55	0.12	0.72
12	P056 to P060	0.8	0.81	0.81	0.8	0.81	0.81
13	P061 to P065	0.34	0	0	0.33	0	0
14	P066 to P070	0.47	0.28	0.7	0.53	0.28	0.69
15	P071 to P075	0.15	0.79	0.79	0.73	0.8	0.79
16	P076 to P080	0.35	0	0.17	0.35	0	0.25
17	P081 to P085	0.48	0.58	0.56	0.51	0.57	0.56
18	P086 to P090	0.47	0.59	0.59	0.56	0.59	0.58
19	P091 to P095	0	0	0	0	0	0
20	P096 to P100	0.1	0	0	0.16	0	0
21	P101 to P105	0.22	0	0.04	0.22	0	0.02
22	P106 to P110	0	0	0	0	0	0
23	P111 to P115	0.22	0	0	0.22	0	0
24	P116 to P120	0.21	0.13	0.2	0.21	0.18	0.22
25	P121 to P125	0	0	0	0	0	0
26	P126 to P130	0.2	0.18	0.19	0.19	0.19	0.21
27	P131 to P135	0.16	0.16	0.16	0.18	0.16	0.15
28	P136 to P140	0	0	0	0	0	0
29	P141 to P145	0.16	0	0	0.21	0	0
30	P146 to P150	0.21	0	0.01	0.21	0	0.03
31	P151 to P155	0	0	0	0	0	0
32	P156 to P160	0.2	0	0	0.18	0	0.04
33	P161 to P165	0.19	0.24	0.22	0.16	0.25	0.23
34	P166 to P170	0	0	0	0	0	0
35	P171 to P175	0.12	0.12	0.11	0.12	0.13	0.12
36	P176 to P180	0.14	0.13	0.14	0.13	0.13	0.14
	Average	0.30	0.20	0.27	0.32	0.20	0.27

Another experiment is to compare the objective value obtained by a million runs of algorithm A4, and the value obtained by one thousand runs of each algorithm. The numbers in Table 3.22 mean the deviations between algorithm A4 and each algorithm.

The value is lower, the deviation is lower. A zero value represents the objective value obtained before the thousandth run of the algorithm is as good as the value obtained by a million runs of algorithm A4.

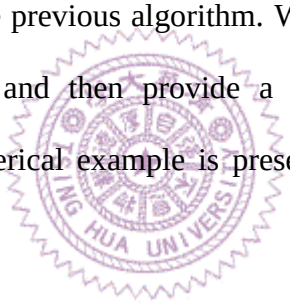
In P001 to P090, zeros appear in A2, A3, A5 and A6, especially A3 and A6. It means A3 and A6 are good for problems with less products. But general speaking, algorithm A4 has the best performance.

Table3.22. Average Deviation

Set	Problems	$\Delta\bar{Z}_{A1}$	$\Delta\bar{Z}_{A2}$	$\Delta\bar{Z}_{A3}$	$\Delta\bar{Z}_{A4}$	$\Delta\bar{Z}_{A5}$	$\Delta\bar{Z}_{A6}$
1	P001 to P005	0.42	1	1	0.56	1	1
2	P006 to P010	0.4	1	0	0.45	1	0
3	P011 to P015	0.15	0.12	0.08	0.14	0.08	0.08
4	P016 to P020	0.41	1	1	0.4	1	1
5	P021 to P025	0.49	1	0	0.37	1	0
6	P026 to P030	0.15	0.07	0.07	0.07	0.07	0.08
7	P031 to P035	0.26	1	0.04	0.29	1	0.02
8	P036 to P040	0.24	0.02	0.07	0.12	0.04	0.06
9	P041 to P045	0.27	0.03	0.08	0.09	0.07	0.09
10	P046 to P050	0.28	1	1	0.31	1	1
11	P051 to P055	0.28	1	0	0.29	1	0
12	P056 to P060	0.12	0.08	0.07	0.12	0.06	0.06
13	P061 to P065	0.23	1	1	0.22	1	1
14	P066 to P070	0.22	1	0	0.09	1	0
15	P071 to P075	1	0	0.03	0.08	0	0.06
16	P076 to P080	0.12	1	0.04	0.15	1	0.6
17	P081 to P085	0.19	1	0	0.1	1	0
18	P086 to P090	0.15	0	0	0.04	0	0
19	P091 to P095	1	1	1	1	1	1
20	P096 to P100	0.11	1	0.22	0.16	1	0.15
21	P101 to P105	0.11	1	1	0.12	1	1
22	P106 to P110	1	1	1	1	1	1
23	P111 to P115	0.14	1	1	0.15	1	0.16
24	P116 to P120	0.09	0.19	0.08	0.09	0.17	0.09
25	P121 to P125	1	1	1	1	1	1
26	P126 to P130	0.05	0.06	0.06	0.05	0.07	0.05
27	P131 to P135	0.04	0.06	0.05	0.04	0.07	0.05
28	P136 to P140	1	1	1	1	1	1
29	P141 to P145	0.32	1	1	0.34	1	1
30	P146 to P150	0.05	1	1	0.05	1	1
31	P151 to P155	1	1	1	1	1	1
32	P156 to P160	0.06	0.32	1	0.05	1	1
33	P161 to P165	0.04	0.01	0.01	0.06	0	0.01
34	P166 to P170	1	1	1	1	1	1
35	P171 to P175	0.03	0.04	0.04	0.03	0.05	0.03
36	P176 to P180	0.05	0.04	0.04	0.03	0.05	0.05
	Average	0.35	0.61	0.47	0.30	0.63	0.44

Chapter 4 Parallel Machine Problems

In the previous chapter we successfully use a two-stage heuristic to solve a single machine problem. Now we attempt to extend the heuristic to the parallel-machine problem which is also a common option in manufacturing environment. In this chapter we face the multi-machine type CLSP with sequence-dependent setup time and initial inventory. Non-identical and parallel machines are considered. Although the multi-machine problem is more difficult because the machine selection for products will make the decision problem more complicated. However, we also adopt the basic ideas of the backward construction of solutions. Therefore, we only need to make slight modification of the previous algorithm. We will describe the problem by using a mathematical model and then provide a revised backward method for multi-machine. Finally, a numerical example is presented to illustrate the proposed solution method.



4.1 Problem Description, Notations and Assumptions

In this case, each product can be produced on any one of given machines. Machines are non-identical in parallel. For each product, the processing rate may be different for machines. Capacities are also dynamic for each machine in periods. Deterministic and dynamic weekly demands should be satisfied without shortage. Sequence-dependent setup time and the initial inventory are also considered. Setup cost is ignored. The goal is to determine the production sequence for each machine and production quantity such that the total inventory cost is minimized.

The assumptions are the same as chapter 3. Parameters and variables are given in Table 4.1 and Table 4.2, respectively.

Table 4.1. Parameters

Parameters	Definition
J	Number of products
T	Number of periods
M	Number of parallel machines
K	Number of products can be produced in a period
d_{jt}	Deterministic demand for product j in period t (quantity unit)
C_{mt}	Available production capacity of machine m in period t (hour)
p_{jm}	Processing rate of product j on machine m (quantity unit/hour)
s_{ijm}	Setup time of a changeover from product i to product j on machine m (hour)
u^{st}	The upper bound of total setup time (hour)
u_j	The upper bound of inventory for product j (unit)
l_j	The lower bound of inventory for product j (unit)
v_j	The shelf life of inventory for product j (period)
h_j	Holding costs for product j (dollar/quantity unit/period)
I_{j0}	Initial inventory for product j (quantity unit)
z_{jm0}	$z_{jm0} = 1$, if the machine is setup for product j on machine m at the beginning. ($z_{jm0} = 0$, otherwise.)

Table 4.2. Variables

Variables	Definition
I_{jt}	Inventory of product j at the end of period t (unit)
x_{jmt}	Processing time of product j on machine m in period t (hour)
y_{ijmt}	$y_{ijmt} \geq 1$, if a setup occurs from product i to product j in machine m in period t ($y_{ijmt} = 0$, otherwise.)
$z_{jmt\tau}$	$z_{jmt\tau} = 1$, if product j is τ^{th} produced product on machine m in period t ($z_{jmt\tau} = 0$, otherwise)
S_{mt}	Setup time used in period t (hour)
α_{mt}	Part of setup time, at the beginning on machine m of period t (hour)
β_{mt}	Part of setup time, at the end on machine m of period t (hour)
e_{mt}	Idle time on machine m in period t (hour)

4.2 Mathematical Model

The propose problem is formulated as a mixed-integer programming as follows:

$$\text{Minimize} \sum_{t=1}^{T}\sum_{j=1}^{J} h_j I_{jt}, \tag{4.1}$$

subject to

$$I_{i,t-1} + \sum_{m=1}^{M} p_{jm} x_{jmt} - I_{jt} = d_{jt}, \qquad j=1,\ldots,J, t=1,\ldots,T, \tag{4.2}$$

$$\sum_{j=1}^{J} x_{jmt} + S_{mt} + e_{mt} = C_{mt}, \qquad m=1,\ldots,M, t=1,\ldots,T, \tag{4.3}$$

$$x_{jmt} \le C_{mt} \sum_{\tau=1}^{K} z_{jmt\tau}, \qquad j=1,\ldots,J, m=1,\ldots,M, t=1,\ldots,T, \tag{4.4}$$

$$\sum_{j=1}^{J} z_{jmt\tau} = 1, \qquad m=1,\ldots,M, t=1,\ldots,T, \tau=1,\ldots,K, \tag{4.5}$$

$$S_{mt} = \sum_{i=1}^{J}\sum_{j=1}^{J} y_{ijmt} s_{ijm} + \alpha_{mt} + \beta_{mt}, \qquad m=1,\ldots,M, t=1,\ldots,T, \tag{4.6}$$

$$s_{ijm}\left(z_{im,t-1,K} + z_{jmt1} - 1\right) \le \alpha_{mt} + \beta_{m,t-1}, \quad i,j=1,\ldots,J, m=1,\ldots,M, t=2,\ldots,T, \tag{4.7}$$

$$z_{imt\tau} + z_{jmt,\tau+1} \le y_{ijmt} + 1, \quad i,j=1,\ldots,J, m=1,\ldots,M, t=1,\ldots,T, \tau=1,\ldots,K-1, \tag{4.8}$$

$$\sum_{m=1}^{M}\sum_{t=1}^{T} S_{mt} \le ub^{st}, \tag{4.9}$$

$$I_{jt} \le u_j, \qquad j=1,\ldots,J, t=1,\ldots,T, \tag{4.10}$$

$$I_{jt} \ge l_j, \qquad j=1,\ldots,J, t=1,\ldots,T, \tag{4.11}$$

$$I_{jt} \le \sum_{\sigma=t}^{t+v_j} d_{j\sigma}, \qquad j=1,\ldots,J, t=1,\ldots,T, \tag{4.12}$$

$$z_{jmt\tau} \in \{0,1\}, \qquad i,j=1,\ldots,J, m=1,\ldots,M, t=1,\ldots,T, \tau=1,\ldots,K, \tag{4.13}$$

$$\alpha_{mt}, \beta_{mt}, S_{mt}, r_{mt} \ge 0, \qquad m=1,\ldots,M, t=1,\ldots,T, \tag{4.14}$$

$$I_{jt} \ge 0, \qquad j=1,\ldots,J, t=1,\ldots,T, \tag{4.15}$$

$$x_{jmt} \ge 0, \qquad j=1,\ldots,J, m=1,\ldots,M, t=1,\ldots,T, \tag{4.16}$$

$y_{ijmt} \geq 0$, $i, j = 1, \ldots, J, m = 1, \ldots, M, t = 1, \ldots, T$. (4.17)

The objective (4.1) is to minimize the total inventory cost without setup costs and constraints (4.2) are the typical inventory balance equations. Note that (4.1) and (4.2) are the same as (3.1) and (3.2) respectively because the weekly inventory and weekly demand have no change while multi-machine case is considered. Constraints (4.3) are the capacity restriction of each machine. Constraints (4.4) guarantee that as long as $x_{jmt} > 0$, product j should be scheduled on machine m in period t. Constraints (4.5) ensure that at each position on machines there is exact one product being scheduled in periods. Setup time for each period and each machine is computed by constraints (4.6). Constraints (4.7) divides setup time into two parts, α and β, which are performed in two adjacent periods, if setup state on the same machine are different in the end of last period and at the beginning of this period. Constraints (4.8) shows that the indicator variable $y_{ijmt} \geq 1$ if product i and product j are produced in sequence on machine m in period t. Constraint (4.9) is the restriction of total setup time for the whole planning horizon. Constraints (4.10), (4.11) and (4.12) are the inventory upper bound, lower bound and the shelf life constraint, respectively, and they are also same to the single machine case. Finally, constraints (4.13) are binary restrictions and constraints (4.14) to (4.17) are non-negative constraints.

The number of real variables, binary variables and constraints are summary in Table 4.3. A small numerical example shown in Example 4.1 is solved by using the above model.

Table 4.3. Complexity of the Parallel Machine Model

Real variables	$J \times T + 4 \times M \times T + J \times M \times T + J^2 \times M \times T$
Binary variables	$J \times M \times T \times K$
Constraints	$1 + 4 \times J \times T + 2 \times M \times T + J \times M \times T + M \times T \times K$ $+ J^2 \times M \times T + J^2 \times M \times T \times K$

Example 4.1. There are two products ($J = 2$) and three periods ($T = 3$) being scheduled on two parallel machines ($M = 2$). We assume that at most $K = 2$ products can be produced on machines in periods. The available production time is 30 for each machine for all periods. Lower bound of inventory is zero; upper bound of inventory, shelf life and upper bound of setup time are assumed to be infinite. We assume that initial inventory is zero ($I_{10} = 0$, $I_{20} = 0$) and two machine are both set up for product one to be produced at the beginning ($z_{110} = 1$, $z_{120} = 1$). Demands, setup time and other known parameters are given by Table 4.4.

Table 4.4. Parameters

	d_{j1}	d_{j2}	d_{j3}	s_{1j1}	s_{2j1}	s_{1j2}	s_{2j2}	p_{j1}	p_{j2}	h_j
$j = 1$	30	0	30	0	10	0	10	1	1	4
$j = 2$	0	65	10	10	0	10	0	1	1	9

We formulated it as a mixed-integer programming problem and solved by using LINGO. The objective value is 45. The optimal solution is shown in Table 4.5 and the corresponding Gantt chart is given in Figure 4.1.

Table 4.5. The Optimal Solution

	$t = 1$	$t = 2$	$t = 3$
(x_{11t}, x_{12t})	(20, 10)	(0, 0)	(20, 10)
(x_{21t}, x_{22t})	(0, 5)	(30, 30)	(0, 10)
(S_{1t}, S_{2t})	(10, 10)	(0, 0)	(10, 10)
$(\alpha_{1t}, \alpha_{2t})$	(0, 0)	(0, 0)	(10, 0)
(β_{1t}, β_{2t})	(10, 0)	(0, 0)	(0, 0)
(e_{1t}, e_{2t})	(0, 5)	(0, 0)	(0, 0)
$(z_{11t1}, z_{11t2}, z_{12t1}, z_{12t2})$	(1, 1, 1, 0)	(0, 0, 0, 0)	(1, 1, 0, 1)
$(z_{21t1}, z_{21t2}, z_{22t1}, z_{22t2})$	(0, 0, 0, 1)	(1, 1, 1, 1)	(0, 0, 1, 0)

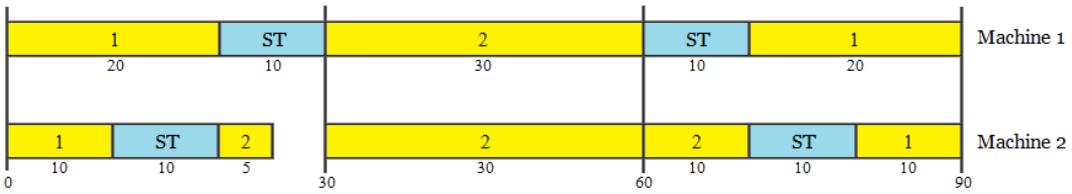


Figure 4.1. Gantt Chart of the Optimal Solution

4.3 Solution Method

The basic idea of the backward method is adopted for the parallel-machine case. But it still has some modifications for products selection and the timing of making decisions. In the following descriptions we only state that different from the solution method for single-machine case.

Initialization: Several data have to be setup in advance. Lot-sizes x_{jmt} are set to be zero for all product j, machine m and period t in the beginning. Let D_{jt} be the cumulative unsatisfied demand in terms of capacity unit (hour) from period t to period T. The D_{jt} are determined by equations (3.17). Without loss of generality we set D_{j0} = 0 for all j. Let TD be the total unsatisfied demand and computed by equation (3.18). For each product j, nd_{jt} denotes the next but last period which is less than or equal to t with unsatisfied demand. The nd_{jt} are determined by equations (3.19). Let $\tilde{C}_{mt}$ be the cumulative capacity of machine m from period 1 to period t and determined by equations (4.18):

$$\tilde{C}_{mt} := \sum_{\sigma=1}^{t} C_{m\sigma}, \quad m = 1,\ldots,M, t = 1,\ldots,T. \tag{4.18}$$

Moreover, several variables are introduced in order to identify the current period of machine m in progress. Let t_m be the period checked in progress of machine m. The t_m are initialized as equations (4.19):

$$t_m := T, \quad m = 1,\ldots,M. \tag{4.19}$$

Selection: If product j* and machine m* are selected to be the next produced

production, and the machine is setup for the product, then product j^* will be produced with lot-size $x_{j^*,t} := \min\{ C_{m^*,t}, D_{j^*,t} \}$ and then go to next iteration. A "jump-back" may occur if the selected machine is setup for the selected product in period t, but there is no unsatisfied demand of the product after period t.

We hope that all productions are scheduled as later as they can without shortage, so in each selection decision we use the latest checked period of machines as the selected checked period t, i.e.

$$t := \max\{ t_1, t_2, \ldots, t_M \}. \tag{4.20}$$

The program will compute a measure for each combination of products and machines for every iteration. Measures are determined according to incurred setup time and inventory. We will assign a probability to each possible choice according to each measure and choose one of them randomly. The following formulas include two kinds of measures are extended from *measure 1* (equations (3.21)) and *measure 3* (equations (3.23)). The γ is a control parameter used to balance inventory and setup time. Based on the current setup state i_m and the checked period t_m on machine m, the measure of producing product j on machine m in period t is defined as follows:

Measure 5

$$r_{jmt}(i_m, t_m) := \begin{cases} (1-\gamma)\cdot D_{jt} - \gamma \cdot s_{j,i_m,m} & \text{if } D_{jt} > 0 \\ (1-\gamma)\cdot D_{j,nd_{j,t-1}} & \text{if } (j = i_m) \wedge \left(TD \le \tilde{C}_{m,nd_{j,t-1}} + \sum_{m' \ne m} \tilde{C}_{m't_m} \right), \\ -\infty & \text{if } t > t_m \\ -\infty & \text{otherwise} \end{cases} \tag{4.21}$$

$$j = 1, \ldots, J, m = 1, \ldots, M, 0 \le \gamma \le 1.$$

Measure 6

$$r_{jmt}(i_m, t_m) := \begin{cases} (1-\gamma)\cdot \left(D_{jt} - \sum_{j' \ne j} D_{j't} \right) - \gamma \cdot s_{j,i_m,m} & \text{if } D_{jt} > 0 \\ (1-\gamma)\cdot \left(D_{j,nd_{j,t-1}} - \sum_{j' \ne j} D_{j',nd_{j,t-1}+1} \right) & \text{if } (j = i_m) \wedge \left(TD \le \tilde{C}_{m,nd_{j,t-1}} + \sum_{m' \ne m} \tilde{C}_{m't_m} \right), \\ -\infty & \text{if } t > t_m \\ -\infty & \text{otherwise} \end{cases}$$

$$j = 1, \ldots, J, m = 1, \ldots, M, 0 \le \gamma \le 1. \tag{4.22}$$

where i_m is the setup state on machine m.

Function (4.21) and (4.22) both has four cases. The first case occurs if unsatisfied demand of product j exists in period t. The second case occurs if a jump-back to the period $nd_{j,t-1}$ on machine m is feasible. The third case occurs if machine m is not the machine which still has capacity in period t. Otherwise cases belong to the fourth.

For positive value purpose, we use function (4.23) to normalize measures.

$$\rho_{jmt}(i_m,t_m) := \begin{cases} 0 & \text{if } r_{jmt}(i_m,t_m) = -\infty \\ \left(r_{jmt}(i_m,t_m) - \bar{r}_{jmt}(i_m,t_m) + \varepsilon \right)^{\delta} & \text{if } r_{jmt}(i_m,t_m) > -\infty \end{cases}, \quad (4.23)$$

where $\bar{r}_{jmt}(i_m,t_m) = \min\{ r_{kmt}(i_m,t_m) \mid k = 1,\ldots J \wedge r_{kmt}(i_m,t_m) > -\infty \}$,

$j = 1,\ldots,J, m = 1,\ldots,M, \delta \geq 0, \varepsilon > 0$.

Setup: After deciding product j^* and machine m^* to be the next, if machine m^* is setup for product i_{m^*}, the setup time s_{j^*,i_{m^*},m^*} is needed to be arranged to machine m. If $s_{j^*,i_{m^*},m^*} \geq C_{m^*,t}$ then s_{j^*,i_{m^*},m^*} will be split into $\alpha_{m^*,t}$ and $\beta_{m^*,t-1}$ to cross from period $t-1$ to period t. $\alpha_{m^*,t}$, $\beta_{m^*,t-1}$ and $C_{m^*,t}$ are updated by equations (4.24), (4.25) and (4.26) respectively. If $s_{j^*,i_{m^*},m^*} < C_{m^*,t}$ then $\alpha_{m^*,t}$ and $\beta_{m^*,t-1}$ are not updated and $C_{m^*,t}$ are updated by equation (4.26). However, if the setup state does not need to be changed, i.e. $s_{j^*,i_{m^*},m^*} = 0$, then nothing will be changed.

$$\alpha_{m^*t}(j^*,i_{m^*}) := \min\{ C_{mt}, s_{j^*i_{m^*},m^*} \}, \quad (4.24)$$

$$\beta_{m^*t-1}(j^*,i_{m^*}) := \max\{ s_{j^*,i_{m^*},m^*} - C_{mt}, 0 \}, \quad (4.25)$$

$$C_{m^*,t}(j^*,i_{m^*}) := \min\{ C_{mt} - s_{j^*,i_{m^*},m^*}, 0 \}. \quad (4.26)$$

Production: Product j^* will be produced on machine m^* with lot-size $x_{j^*,m^*,t} := \min\{ C_{m^*,t}, D_{j^*,t} \}$, and then j^* will be inserted into the first position of $seq_{m^*,t}$ which represents the production sequence on machine m^* in period t. Subsequently, $D_{j^*,t}$, TD and $\tilde{C}_{m^*,t}$ are updated by equations (3.17), (3.18) and (4.18), respectively. If all of the

demand are satisfied, i.e. $TD = 0$, it is the end of program. Otherwise, it goes back to *selection* to re-select a product to produce.

To see how the backward method works, a numerical example is given in section 4.4. A high level code of the procedure is given in Appendix III:

4.4 Numerical Example

We use an example to illustrate the backward method on parallel machines. There are three products ($J = 3$), two parallel machines ($M = 2$) and five periods ($T = 5$) being planned. The available production time is 25 for all periods for two machines. Lower bound of inventory is zero; upper bound of inventory, shelf life and upper bound of setup time are assumed to be infinite. We assume that two machines are identical. Therefore, processing rate of products and setup time between two products are the same on two machines. There is no initial inventory at first. Demands, setup time and other known parameters are given by Table 4.6. We will use *measure 6* to illustrate this example and we will show that how to obtain a feasible schedule by the backward method.

Table 4.6. Parameter Values

	d_{j1}	d_{j2}	d_{j3}	d_{j4}	d_{j5}	s_{1jm}	s_{2jm}	s_{3jm}	p_{jm}	h_j
$j = 1$	20	20	0	10	0	0	10	5	1	4
$j = 2$	10	10	30	0	10	10	0	10	1	3
$j = 3$	0	0	20	10	20	5	10	0	1	6

First, for the backward method, an initialize operation is executed. The initialized data is shown in Table 4.7 and we set $\gamma = 0.2$, $\delta = 1$ and $\varepsilon = 1$, current setup state $i_1 = i_2 = 0$ for two machines (zero setup time from product '0' to each product).

Table 4.7. Initial Data

	t	0	1	2	3	4	5
Original Problem	d_{1t}		20	20	0	10	0
	d_{2t}		10	10	30	0	10
	d_{3t}		0	0	20	10	20
	C_{1t}	0	25	25	25	25	25
	C_{2t}	0	25	25	25	25	25
Cumulative Demand	D_{1t}	0	50	30	10	10	0
	D_{2t}	0	60	50	40	10	10
	D_{3t}	0	50	50	50	30	20
Total Demand	TD		160				
Next-Demand Period	nd_{1t}	0	1	2	2	4	4
	nd_{2t}	0	1	2	3	3	5
	nd_{3t}	0	0	0	3	4	5
Cumulative Capacity	$\tilde{C}_{1t}$	0	25	50	75	100	125
	$\tilde{C}_{2t}$	0	25	50	75	100	125

Iteration 1

Selection: Let $t_1 := 5$ and $t_2 := 5$ at first, so $t := \max\{ t_1, t_2 \} = 5$. Based on setup state $i_1 = 0$, $i_2 = 0$, the measures for each product at period t are determined as follows:

$$r_{115}(0,5) = 0.8 \times (10 - (10 + 20)) - 0.2 \times 0 = -16 \quad \rightarrow \quad \rho_{115}(0,5) = 1,$$

$$r_{125}(0,5) = 0.8 \times (10 - (10 + 20)) - 0.2 \times 0 = -16 \quad \rightarrow \quad \rho_{125}(0,5) = 1,$$

$$r_{215}(0,5) = 0.8 \times (10 - 20) - 0.2 \times 0 = -8 \quad \rightarrow \quad \rho_{215}(0,5) = 8,$$

$$r_{225}(0,5) = 0.8 \times (10 - 20) - 0.2 \times 0 = -8 \quad \rightarrow \quad \rho_{225}(0,5) = 8,$$

$$r_{315}(0,5) = 0.8 \times (20 - 10) - 0.2 \times 0 = 8 \quad \rightarrow \quad \rho_{315}(0,5) = 24,$$

$$r_{325}(0,5) = 0.8 \times (20 - 10) - 0.2 \times 0 = 8 \quad \rightarrow \quad \rho_{325}(0,5) = 24.$$

Let P_{jmt} denotes the probability to select product j produced on machine m in period t. Thus $P_{115} = 1/66$, $P_{125} = 1/66$, $P_{215} = 8/66$, $P_{225} = 8/66$, $P_{315} = 24/66$ and $P_{325} = 24/66$. The operation of product 3 produced on machine 2 is selected at first.

Setup: There is no setup time between product 3 and product '0' on machine 2.

Production: The lot-size of product 3 on machine 2 at period 5 is x_{325} = min{ 25, 20 } = 20. The capacity is updated: C_{25} := 25 – 20 = 5. The total demand is updated: *TD* := 160 – 20 = 140. The cumulative demands of product 3 are updated in the following table. Product 3 is added into the production sequence of period 5 of machine 2: seq_{25} := (3).

Table 4.8. Cumulative Demands of Product 3

	t	0	1	2	3	4	5
Cumulative Demand	D_{3t}	0	30	30	30	10	0

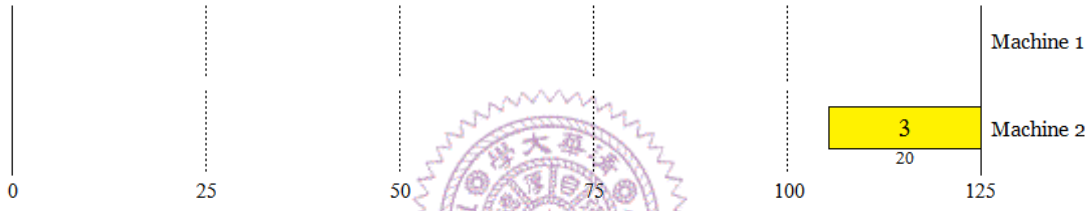


Figure 4.2. Gantt Chart

Iteration 2

Selection: Based on setup state i_1 = 2, i_2 = 3, t_1 = 5 and t_2 = 5, the measures for each product at period t := max{ t_1, t_2 } = 5 are determined as follows:

$r_{115}(0, 5) = 0.8 \times (10 - 10) - 0.2 \times 0 = 0 \quad \rightarrow \quad \rho_{115}(0, 5) = 2,$

$r_{125}(3, 5) = 0.8 \times (10 - 10) - 0.2 \times 5 = -1 \quad \rightarrow \quad \rho_{125}(3, 5) = 1,$

$r_{215}(0, 5) = 0.8 \times (10) - 0.2 \times 0 = 8 \quad \rightarrow \quad \rho_{215}(0, 5) = 9,$

$r_{225}(3, 5) = 0.8 \times (10) - 0.2 \times 10 = 6 \quad \rightarrow \quad \rho_{225}(3, 5) = 8,$

$r_{315}(0, 5) = 0.8 \times (10 - 10) - 0.2 \times 0 = 0 \quad \rightarrow \quad \rho_{315}(0, 5) = 2,$

$r_{325}(3, 5) = 0.8 \times (10 - 10) - 0.2 \times 0 = 0 \quad \rightarrow \quad \rho_{325}(3, 5) = 2.$

Let P_{jmt} denotes the probability to select product j produced on machine m in period t. Thus P_{115} = 2/24, P_{125} = 1/24, P_{215} = 9/24, P_{225} = 8/24, P_{315} = 2/24 and P_{325} = 2/24.

The operation of product 2 produced on machine 1 is selected.

Setup: There is no setup time between product 2 and product '0' on machine 1.

Production: The lot-size of product 2 on machine 1 at period 5 is x_{215} = min{ 25, 10 } = 10. The capacity is updated: C_{15} := 25 – 10 = 15. The total demand is updated: *TD* := 140 – 10 = 130. The cumulative demands of product 2 are updated in the following table. Product 2 is added into the production sequence of period 5 of machine 1: seq_{15} := (2).

Table 4.9. Cumulative Demands of Product 2

	t	0	1	2	3	4	5
Cumulative Demand	D_{2t}	0	50	40	30	0	0

Figure 4.3. Gantt Chart

Iteration 3

Selection: Based on setup state i_1 = 2, i_2 = 3, t_1 = 5 and t_2 = 5, the measures for each product at period *t* := max{ t_1, t_2 } = 5 are determined as follows:

$r_{115}(2, 5) = 0.8 \times (10 - 10) - 0.2 \times 10 = -2 \quad \rightarrow \quad \rho_{115}(2, 5) = 1,$

$r_{125}(3, 5) = 0.8 \times (10 - 10) - 0.2 \times 5 = -1 \quad \rightarrow \quad \rho_{125}(3, 5) = 2,$

$r_{215}(2, 5) = 0.8 \times (30 - (10 + 10)) - 0.2 \times 0 = 8 \rightarrow \quad \rho_{215}(2, 5) = 10,$

$r_{225}(3, 5) = 0.8 \times (30 - (10 + 10)) - 0.2 \times 10 = 6 \rightarrow \quad \rho_{225}(3, 5) = 8,$

$r_{315}(2, 5) = 0.8 \times (10 - 10) - 0.2 \times 10 = -2 \quad \rightarrow \quad \rho_{315}(2, 5) = 1,$

$r_{325}(3, 5) = 0.8 \times (10 - 10) - 0.2 \times 0 = 0 \quad \rightarrow \quad \rho_{325}(3, 5) = 2.$

Let P_{jmt} denotes the probability to select product *j* produced on machine *m* in period *t*.

Thus P_{115} = 1/24, P_{125} = 2/24, P_{215} = 10/24, P_{225} = 8/24, P_{315} = 1/24 and P_{325} = 2/24. The operation of product 1 produced on machine 2 is selected with small probability.

Setup: A setup time s_{132} = 5 is scheduled. Since $s_{132} \geq C_{52}$ = 5, $\alpha_{25} := \min\{ 5, 5 \}$ = 5, $\beta_{24} := \min\{ 5-5, 0 \} = 0$, and the capacity is updated: $C_{52} := 5 - 5 = 0$. Let $t_2 := 4$ and $t := 4$.

Production: The lot-size of product 1 on machine 2 at period 4 is $x_{124} = \min\{ 25, 10 \}$ = 10. The capacity is updated: $C_{24} := 25 - 10 = 15$. The total demand is updated: $TD := 130 - 10 = 120$. The cumulative demands of product 1 are updated in the following table. Product 1 is added into the production sequence of period 4 of machine 2: $seq_{24} := (1)$.

Table 4.10. Cumulative Demands of Product 1

	t	0	1	2	3	4	5
Cumulative Demand	D_{2t}	0	40	20	0	0	0

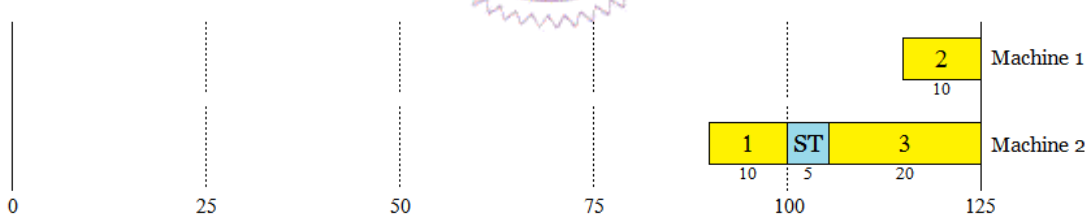


Figure 4.4. Gantt Chart

Iteration 4

Selection: Based on setup state $i_1 = 2$, $i_2 = 1$, $t_1 = 5$ and $t_2 = 4$, the measures for each product at period $t := \max\{ t_1, t_2 \} = 5$ are determined as follows:

$r_{115}(2, 5) = -\infty$ → $\rho_{115}(2, 5) = 0$,

$r_{125}(1, 5) = -\infty$ → $\rho_{125}(1, 5) = 0$,

$r_{215}(2, 5) = 0.8 \times (30 - 10) - 0.2 \times 0 = 16$ → $\rho_{215}(2, 5) = 11$,

$$r_{225}(1,5) = 0.8 \times (30 - 10) - 0.2 \times 10 = 14 \quad \rightarrow \quad \rho_{225}(1,5) = 9,$$

$$r_{315}(2,5) = 0.8 \times (10) - 0.2 \times 10 = 6 \quad \rightarrow \quad \rho_{315}(2,5) = 1,$$

$$r_{325}(1,5) = 0.8 \times (10) - 0.2 \times 5 = 7 \quad \rightarrow \quad \rho_{325}(1,5) = 2.$$

Let P_{jmt} denotes the probability to select product j produced on machine m in period t. Thus $P_{115} = 0$, $P_{125} = 0$, $P_{215} = 11/23$, $P_{225} = 9/23$, $P_{315} = 1/23$ and $P_{325} = 2/23$. The operation of product 2 produced on machine 1 is selected and it forces a jump-back to period 3.

Setup: Zero setup time $s_{221} = 0$ is scheduled.

Production: The jump-back is from period 5 to period 3. The idle time is updated: $e_{15} := C_{15}\ (=15)$, $e_{14} := C_{14}\ (=25)$ and $t := 3$. The lot-size of product 2 on machine 1 at period 3 is $x_{213} = \min\{25, 25\} = 25$. The capacity is updated: $C_{13} := 25 - 25 = 0$ and $t_1 := 2$. The total demand is updated: $TD := 120 - 25 = 95$. The cumulative demands of product 2 are updated in the following table. Product 2 is added into the production sequence of period 3 of machine 1: $seq_{13} := (2)$.

Table 4.11. Cumulative Demands of Product 2

	t	0	1	2	3	4	5
Cumulative Demand	D_{2t}	0	25	15	5	0	0

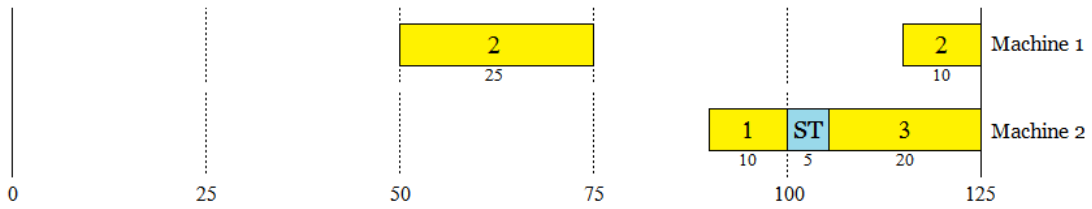
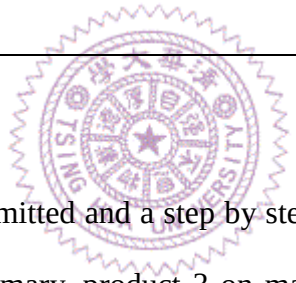


Figure 4.5. Gantt Chart

Table 4.12. Step by Step Data of the Backward Method

I	t	t_1	t_2	i_1	i_2	$r_{11t}(i_1,t_1)$	$r_{12t}(i_1,t_1)$	$r_{21t}(i_1,t_1)$	$r_{22t}(i_1,t_1)$	$r_{31t}(i_1,t_1)$	$r_{32t}(i_1,t_1)$	j^*	m^*	$x_{j^*,m^*,t}$	TD	
1	5	5	5	0	0	–16	–16	–8	–8	8	8	3	2	20	140	seq_{25}:=(3)
2	5	5	5	0	3	0	–1	8	6	0	0	2	1	10	130	seq_{15}:=(2)
3	5	5	5	2	3	-2	–1	8	6	–2	0	1	2			setup
	4		4									1	2	10	120	seq_{24}:=(1)
4	5	5	4	2	1	–∞	–∞	16	14	6	7	2	1			jump-back
	3	3										2	1	25	95	seq_{13}:=(2)
5	4	2	4	2	1	–∞	–13	–∞	–6	–∞	7	3	2	10	85	seq_{24}:=(3,1)
6	3	2	3	2	3	–∞	–5	–∞	–20	–∞	12	3	2	20	65	seq_{23}:=(3)
7	3	2	3	2	3	–∞	11	–∞	2	–∞	–∞	1	2			setup
	2		2									1	2	20	45	seq_{22}:=(1)
8	2	2	2	2	1	2	4	12	10	–∞	–∞	2	1	15	30	seq_{12}:=(2)
9	2	2	2	2	1	6	8	–8	–10	–∞	–∞	1	2			jump-back
	1		1									1	2	20	10	seq_{21}:=(1)
10	2	2	1	2	1	–∞	–∞	8	6	–∞	–∞	2	1			jump-back
	1	1										2	1	10	0	seq_{11}:=(2)

Column I records iterations.



Following iterations are omitted and a step by step data of the backward method is given in Table 4.12. In summary, product 3 on machine 2 is selected as first. At iteration 3, a setup time is schedule between period 4 and period 5. At iteration 4, product 2 on machine 1 is selected and it forces a jump-back to period 3. At iteration 7, a setup time is schedule between period 2 and period 3. At iteration 9, product 1 on machine 2 is selected and it forces a jump-back to period 1. At iteration 10, product 2 on machine 1 is selected and it forces a jump-back to period 1. Gantt charts for each iteration are given in Figure 4.6 and a feasible solution is obtained finally given in Table 4.13 with total cost 15. Note that, if there is any remaining capacity in this period, an idle will appear in the head of this period, since the schedule is constructed backward.

Table 4.13. A feasible Solution

	$t=1$	$t=2$	$t=3$	$t=4$	$t=5$
x_{11t}	0	0	0	0	0
x_{12t}	20	20	0	10	0
x_{21t}	10	15	25	0	0
x_{22t}	0	0	0	0	10
x_{31t}	0	0	0	0	0
x_{32t}	0	0	20	10	20
α_{1t}	0	0	0	0	0
α_{2t}	0	0	5	0	5
β_{1t}	0	0	0	0	0
β_{2t}	0	0	0	0	0
e_{1t}	15	10	0	25	15
e_{2t}	5	5	0	0	0
seq_{1t}	(2)	(2)	(2)	( )	(2)
seq_{2t}	(1)	(1)	(3)	(3,1)	(3)

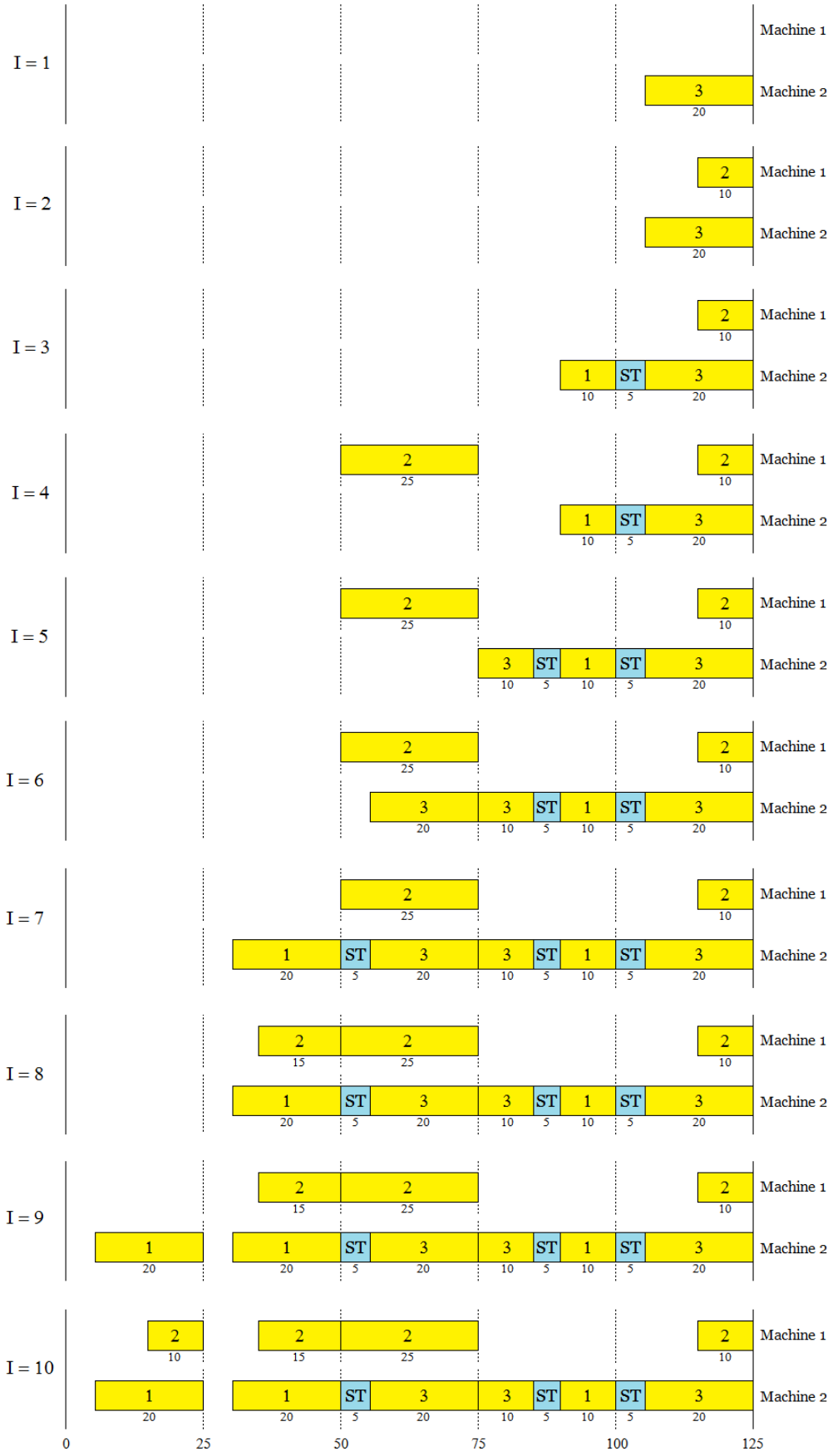


Figure 4.6. Gantt Charts for Each Iteration

Chapter 5 Flow Shop Problems

In this chapter, we extend the proposed problem to a flow shop case. We solve the problem by using the backward method.

5.1 Problem Description, Notations and Assumptions

In this case, products must go through all of the machines in the same order. A product is called a semi-finished product of level *m* when the product is finished on machine *m*. For each product, the processing rate and setup time are given depended on machines. The production sequence of each machine is assumed to be the same. When the product is finished processing on machine *k*, it moves to the next machine *k*+1 only if its finished production quantity is greater than or equal to a specific minimum lot-size. Dynamic and deterministic weekly demands should be satisfied without shortage. Capacities for machines are also dynamic in periods. Sequence-dependent setup time and initial inventory are considered. Setup cost is also ignored. The task is to decide the production sequence and the lot-size for each production such that demands are satisfied and the total cost is minimized. An example is given as follows.

Example 5.1. There are two products and three periods being scheduled on two machines with a flow shop structure. For each machine the available production time is 100 for three periods. Demands and other parameters are given in Table 5.1.

A feasible schedule with total cost 125 is given in Figure 5.1. In the schedule, each lot produced on the first machine must be greater than or equal to a given minimum lot-size then these semi-finished products will be moved to the second machine.

Table 5.1. Parameter Values

	demands			processing rate for machine 1	processing rate for machine 2	setup time for each machine	holding cost of level 1	holding cost of level 2	minimum lot-size
	period 1	period 2	period 3						
product 1	40	20	40	1	2	5	5	10	20
product 2	0	90	60	2	1	10	5	10	30

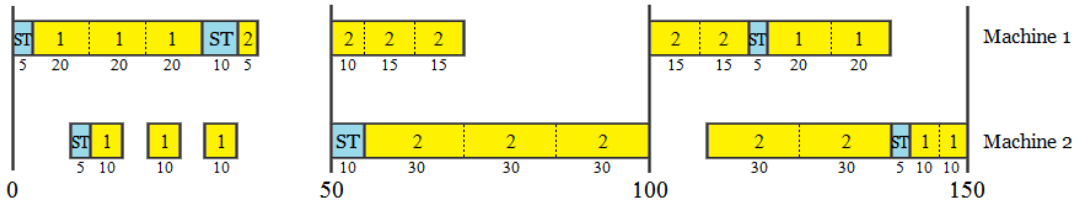


Figure 5.1. Gantt Chart of the Schedule

Several notations are defined in Table 5.2.

Table 5.2. Notations

Notations	Definition
J	Number of products
T	Number of periods
M	Number of machines
d_{jt}	Deterministic demand for product j in period t (quantity unit)
D_{jt}	Cumulative demand for product j from period t to period T
C_t	Available production capacity in period t (hour)
p_{jm}	Processing rate of product j on machine m (quantity unit/hour)
s_{ijm}	Setup time of a changeover from product i to product j on machine m (hour)
h_{jm}	Holding costs for semi-finished product j on machine m (dollar/quantity unit/period)
LS_j	Minimum lot-sizes for product j (quantity unit)
I_{jmt}	Inventory of semi-finished product j on machine m at the end of period t (quantity unit)
x_{jmt}	Processing time of semi-finished product j on machine m in period t (hour)

Following assumptions are given for the flow shop problem and other assumptions are the same as chapter 3:

1. Products are produced on serially arranged machines in a flow shop structure.
2. Production sequences on each machine are the same.
3. A minimum lot-size is given for each product. Each lot must be greater than or equal to the minimum lot-size then it could be moved to the next machine.

5.2 Solution Method

The solution is constructed by two steps. First, the flow shop problem is regarded as a single-machine problem which only has the last machine, and the problem is solve by the backward method presented in chapter 3. It will obtain a partial schedule for the last machine. Next, a complete schedule will be constructed according to the partial schedule.

According to the well-known Johnson's rule, a method of scheduling a number of jobs on two successive work centers. The main objective of Johnson's rule is to find an optimal sequence of jobs such that minimize makespan (the total amount of time it takes to complete all jobs). It also reduces the number of idle time between the two work centers. The intuition behind Johnson's rule is that a job should be started earlier when it has a smaller processing time on machine 1 and should be started later when it has a smaller processing time on machine 2.

Using the ideas of Johnson's rule, we create a weight w_i for each product i. The weight is computed by following equations:

$$w_j = \sum_{m=1}^{M} p_{jm} \cdot m, \qquad j = 1, \ldots, J, \tag{5.1}$$

where p_{jm} is the processing rate of product j on machine m. The high value of the weight means that the product tends to be processed quickly on rear machines. In

every selection step of the backward method, we hope the selected product is urgent and it does not incur setups. In the flow shop case, there is also a key point is that we have to choose products which has higher processing rate on rear machines. Otherwise it may exceed capacity on preceding machines and leads to an infeasible solution. Therefore, a new selection measure given in equations (5.2) is used for flow shop problems. Note that, the 'jump back' mechanism is no longer in use because it will tend to a result in a more possibility for infeasible solutions in this complicated problem.

Measure 7

$$r_{jt}(i) := \begin{cases} (1-\gamma)\cdot D_{jt}\cdot w_i - \gamma \cdot s_{jiM} & \text{if } D_{jt} > 0 \\ -\infty & \text{otherwise} \end{cases}, \quad j = 1,\ldots,J, 0 \le \gamma \le 1. \qquad (5.2)$$

After obtaining a partial schedule for the last machine *M*, we can complete the schedule from machine $M-1$ to machine 1 in accordance with the production sequence of the machine *M*. In the scheduling for preceding machines, each lot must be greater than its minimum lot-size and should be scheduled as later as it can. If there is no feasible schedule under the partial schedule, then a new partial schedule is generated again by the backward method, until a complete schedule is obtained. Note that, if there is any remaining time in the period, an idle will appear in the head of the period, since the schedule is constructed backward.

5.3 Numerical Example

There are three products ($J = 3$) and three periods ($T = 3$) being planned in on a two-machine flow shop system ($M = 2$). Capacity is 50 for all periods for two machines. Products are produced on machine one and then on machine two. Upper bound of setup time is assumed to be infinite. There is no initial inventory. Demands, setup time and other known parameters are given by Table 5.3. Note that, two sets of minimum lot-sizes are used to illustrate.

Table 5.3. Parameter Values

	d_{it}			s_{ij1} , s_{ij2}								
	$t = 1$	$t = 2$	$t = 3$	$j = 1$	$j = 2$	$j = 3$	p_{i1}	p_{i2}	h_{i1}	h_{i2}	LS_i	LS'_i
$i = 1$	20	20	0	0	10	5	1	1	4	4	10	5
$i = 2$	10	10	30	10	0	10	2	1	3	3	10	5
$i = 3$	0	0	20	5	10	0	1	2	6	6	10	5

Table 5.4. Initial Data

	t	0	1	2	3
Original Problem	d_{1t}		20	20	0
	d_{2t}		10	10	30
	d_{3t}		0	0	20
Cumulative Demand	D_{1t}	0	40	20	0
(hours)	D_{2t}	0	50	40	30
	D_{3t}	0	10	10	10
Total Demand	TD		100		
Cumulative Capacity	$\tilde{C}_t$	0	50	100	150

First, for the backward method, an initialize operation is executed. The initialized data is shown in Table 5.4 and we set $\gamma = 0.5$, $\delta = 1$ and $\varepsilon = 1$, current setup state $i = 0$ for machine two (zero setup time from product '0' to each product). The weight w_i are determined as follows: $w_1 = 1 \times 1 + 1 \times 2 = 3$, $w_2 = 2 \times 1 + 1 \times 2 = 4$, $w_3 = 1 \times 1 + 2 \times 2 = 5$.

Iteration 1

Selection: The measures for each product at period $t := T\,(=3)$ are determined as follows:

$r_{13}(0) = -\infty \quad \rightarrow \quad \rho_{13}(0) = 0,$

$r_{23}(0) = 0.5 \times 30 \times 4 - 0.5 \times 0 = 60 \quad \rightarrow \quad \rho_{23}(0) = 26,$

$r_{33}(0) = 0.5 \times 10 \times 5 - 0.5 \times 0 = 25 \quad \rightarrow \quad \rho_{33}(0) = 1,$

Let P_{jt} denotes the probability to select product j in period t. Thus $P_{13} = 0$, $P_{23} = 26/27$ and $P_{33} = 1/27$. Product 2 is selected at first.

Setup: There is no setup time between product 2 and product '0'.

Production: The lot-size of product 2 on machine 2 at period 3 is x_{223} = min{ 50, 30 } = 30. The capacity is updated: C_{23} := 50 – 20 = 30. The total demand is updated: TD := 100 – 30 = 70. The cumulative demands of product 2 are updated in the following table. Product 2 is added into the production sequence of period 3: seq_3 := (2).

Table 5.5. Cumulative Demands of Product 2

	t	0	1	2	3
Cumulative Demand	D_{2t}	0	20	10	0

Figure 5.2. Gantt Chart

Iteration 2

Selection: Based on setup state i = 2, the measures for each product at period t = 3 are determined as follows:

$r_{13}(2) = -\infty$ → $\rho_{13}(2) = 0$,

$r_{23}(2) = -\infty$ → $\rho_{23}(2) = 0$,

$r_{33}(2) = 0.5 \times 10 \times 5 - 0.5 \times 10 = 20$ → $\rho_{33}(2) = 1$,

Let P_{jt} denotes the probability to select product j in period t. Thus P_{13} = 0, P_{23} = 0 and P_{33} = 1. Product 3 is selected.

Setup: A setup time s_{322} = 10 is scheduled, and the capacity is updated: C_{23} := 20 – 10 = 10.

Production: The lot-size of product 3 on machine 2 at period 3 is x_{323} = min{ 10,

10 } = 10. The capacity is updated: C_{23} := 10 – 10 = 10 and t := 2. The total demand is updated: TD := 70 – 10 = 60. The cumulative demands of product 3 are updated in the following table. Product 3 is added into the production sequence of period 3: seq_3 := (3, 2).

Table 5.6. Cumulative Demands of Product 3

	t	0	1	2	3
Cumulative Demand	D_{3t}	0	0	0	0

Figure 5.3. Gantt Chart

Iteration 3

Selection: Based on setup state i = 3, the measures for each product at period t = 2 are determined as follows:

$r_{12}(3) = 0.5 \times 20 \times 3 - 0.5 \times 5 = 27.5 \quad \rightarrow \quad \rho_{12}(3) = 13.5,$

$r_{22}(3) = 0.5 \times 10 \times 4 - 0.5 \times 10 = 15 \quad \rightarrow \quad \rho_{22}(3) = 1,$

$r_{32}(3) = -\infty \quad \rightarrow \quad \rho_{32}(3) = 0,$

Let P_{jt} denotes the probability to select product j in period t. Thus P_{13} = 13.5/14.5, P_{23} = 1/14.5 and P_{33} = 0. Product 1 is selected.

Setup: A setup time s_{132} = 5 is scheduled, and the capacity is updated: C_{22} := 50 – 5 = 45.

Production: The lot-size of product 1 on machine 2 at period 2 is x_{122} = min{ 45, 20 } = 20. The capacity is updated: C_{22} := 45 – 20 = 25. The total demand is updated: TD := 60 – 20 = 40. The cumulative demands of product 1 are updated in the

following table. Product 1 is added into the production sequence of period 2: seq_2 := (1).

Table 5.7. Cumulative Demands of Product 1

	t	0	1	2	3
Cumulative Demand	D_{1t}	0	20	0	0

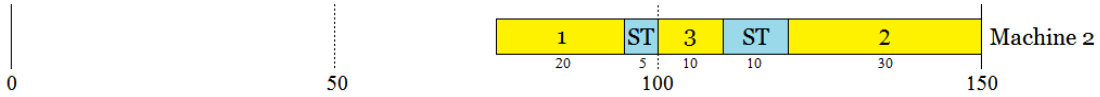


Figure 5.4. Gantt Chart

Following iterations are omitted and a step by step data of the shifting is given in Table 5.8 and corresponding Gantt charts are given in Figure 5.5. At the end of the backward method, a feasible schedule is obtained for machine two.

Table 5.8. Step by Step Data of the Backward Method

I	t	i	$r_{1t}(i)$	$r_{2t}(i)$	$r_{3t}(i)$	$\rho_{1t}(i)$	$\rho_{2t}(i)$	$\rho_{3t}(i)$	j^*	C_t	$D_{j^*,t}$	$x_{j^*,2,t}$	TD	seq_t
1	3	0	–∞	60	50	0	11	1	2	50	30	30	80	(2)
2	3	2	–∞	–∞	45	0	0	1	3	20	20	20	60	(3,2)
3	2	3	27.5	15	–∞	13.5	1	0	1	50	20	20	40	(1)
4	2	1	–∞	15	–∞	0	1	0	2	25	10	10	30	(2,1)
5	1	2	25	20	–∞	6	1	0	1	50	20	20	10	(1)
6	1	1	–∞	15	–∞	0	1	0	2	25	10	10	0	(2,1)

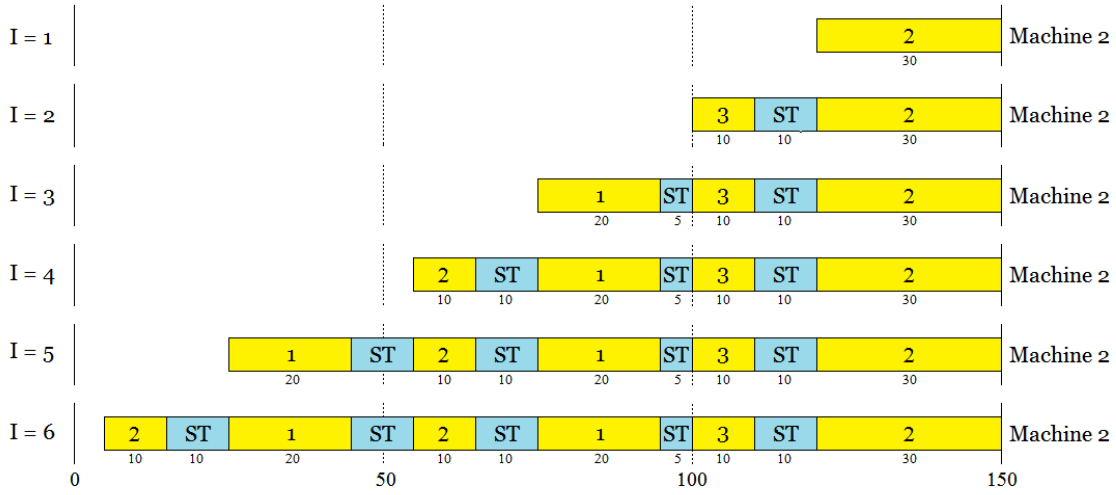


Figure 5.5. Gantt Charts for Each Iteration

The next step is to construct the schedule of machine one according to the schedule of machine two as follows. Figure 5.6 shows the complete schedule with minimum lot-sizes $(LS_1, LS_2, LS_3) = (10, 10, 10)$. The schedule is tight since there is no idle time in the first period on machine one. Figure 5.7 shows the complete schedule with minimum lot-sizes $(LS'_1, LS'_2, LS'_3) = (5, 5, 5)$. Two feasible schedules with total cost 120 and 90 respectively.

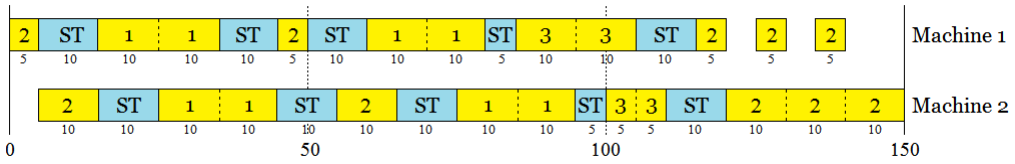


Figure 5.6. Gantt Chart of the Schedule with $(LS_1, LS_2, LS_3) = (10, 10, 10)$

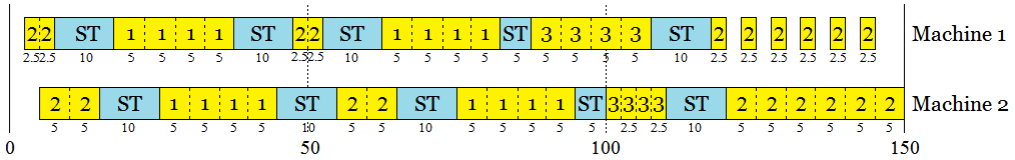


Figure 5.7. Gantt Chart of the Schedule with $(LS'_1, LS'_2, LS'_3) = (5, 5, 5)$

Chapter 6 Conclusions and Future Research

In this thesis we consider capacitated lot-sizing and scheduling problems with initial inventory and sequence-dependent setup time. Instead of assuming zero initial inventories, a simple procedure is provided to convert the problem into the one without initial inventory. We construct a mixed-integer programming formulation which generalizes the former models in the literature. It includes cross-period setup time, safety stock, storage limit and the shelf life of inventory. For large-scale problems we provide a heuristic method which can generate a feasible schedule in reasonable time. The heuristic algorithm comprises two stages. In stage one a backward method is used to generate a good feasible schedule and the schedule is refined in stage two. Moreover, the proposed problem is extended to parallel-machine and flow shop cases and heuristic algorithms are designed for solving them.

In flow shop problems, the assumption that each machine has the same production sequence leads to a permutation scheduling problem. Once the assumption is relaxed, much better solutions will be discovered but the problem is also more complicated. In multi-machine problems, general job shop and open shop are also common structures in the manufacturing system. This will be the focus of our future research. Furthermore, most of literatures focus on problems without shortages or backlogs which usually exist in the real world. This is also an issue for future study.

Appendix I The Backward Method

Procedure_Backward_Method

Begin

Initialization:

$x_{jt} := 0, \forall j, t$; //lot-sizes

$\alpha_t := 0, \beta_t := 0, e_t := 0, \forall t$; //cross-period setup time and idle time

$CST := 0$; //cumulated setup time

$i := 0$; //initial setup state

$seq_t := \phi, \forall t$; //producing sequence

$t := T$; //start at the planning horizon

Compute $D_{jt}, \forall j, t$; //cumulated demand

Compute $nd_{jt}, \forall j, t$; //next-demand period

Compute TD ; //total demand

Compute $\tilde{C}_t, \forall t$; //cumulated capacity

Do While $TD > 0$ and $t > 0$

If $TD > \tilde{C}_t$ or $CST > u^{st}$ then **GoTo** *Initialization*;

Selection:

Determine $\rho_{jt}(i), \forall j$;

If $\sum_j \rho_{jt}(i) = 0$ **then**

$e_t := C_t$; // let idle time be the remaining available time

$t := t - 1$;

GoTo *Selection*;

Else

Choose j^* at random proportional to $\rho_{jt}(i)$;

End

Setup:

If $s_{j^*,i} < C_t$ **then**

$C_t := C_t - s_{j^*,i}$; // update the capacity

If $seq_t = \phi$ **then** $\beta_t := s_{j^*,i}$;

Else

$\alpha_t := \min\{ C_t, s_{j^*,i} \}$;

$\beta_t := \min\{ s_{j^*,i} - C_t , 0 \}$;

$t := t - 1$;

End

Production:

If $j^* = i$ **then** // check whether "jump-back" is required

$r_\sigma := C_\sigma$, $\sigma = t, t-1, \ldots , nd_{j^*,t} + 1$;

$t := nd_{j^*,t}$; // jump-back

End

Do While $D_{j^*,t} > 0$

$x_{j^*,t} := \min\{ C_t , D_{j^*,t} \}$; // assign the production quantity

$C_t := C_t - x_{j^*,t}$; // update capacity

$\tilde{C}_t := \tilde{C}_t - x_{j^*,t}$; // update cumulative capacity

$D_{j^*,\sigma} := D_{j^*,\sigma} - x_{j^*,t}$, $\sigma = 1, 2, \ldots , t$; // update cumulative demands

$TD := TD - x_{j^*,t}$; // update the total demand

$seq_t := (j^* , seq_t)$; // insert j^* into the head of sequence

If $C_t = 0$ **then** $t := t - 1$;

End

$i := j^*$;

End

If $TD > \tilde{C}_t$ or $CST > u^{st}$ then **GoTo** *Initialization*;

Do While $C_t > 0$

$e_t := C_t$; // let idle time be the remaining available time

$t := t - 1$;

End

Evaluate schedule ;

End

Appendix II The Improvement Procedure

Step 1. *Sequencing*

Function_Sequensing(Q, n)

Begin

For each $i \in I$ **do**

$T(\{ i \}, i, i) := 0$; // zero setup time for one-product sequence

End

For each $i, j \in I, i \neq j$ **do**

$T(\{ i, j \}, i, j) := s_{ij}$; // a constant setup time for two-product sequence

End

For $k := 3, 4, \ldots, n$ **do** // sequences with more than two products

For each $I' \subseteq I, |I'| = k$ **do**

For each $i, j \in I', i \neq j$ **do**

$$T(I', i, j) := \min_{u,v \in I'-i-j} \{ T(I'-i-j, u, v) + s_{i,u} + s_{v,j} \} ;$$

End

End

End

$opt := T(I, i_F, i_L)$;

Return opt ; // return the minimum setup time

End

Step 2. *Shifting*

Procedure_Shifting

Begin

If $r_t = 0, \forall t > 1$ **then Exit Procedure** // check whether idle time exists

Sort products by holding costs from high to low ;

For each product i **do**

$t' := \max\{ t \mid e_t > 0 \}$;

$t := t - 1$;

Do While $t \geq 1$

Do While $t' > t$

Determine q_{it} ;

If $q_{it} = 0$ **then Exit Do**

If $x_{it'} = 0$ **then**

Find the best position for the insertion ;

Determine *AST* ; // additional setup time for the movement

End

determine $q'_{it'}$;

$shift := \min\{ q_{it}, q'_{it'} \}$;

If the movement does not violate the setup time constraint **then**

$x_{it} := x_{it} - shift$; // update production quantity

$x_{it'} := x_{it'} + shift$; // update production quantity

$e_t := e_t + shift$; // update idle time

$e_{t'} := e_{t'} - shift$; // update idle time

Update the schedule by inserting or removing setup times;

End

$t' := \max\{ t < t' \mid e_t > 0 \}$;

End

$t := t - 1$;

End

End

End

Appendix III The Backward Method for Parallel Machine Problems

Procedure_Backward_Method_Parallel

Begin

Initialization:

$x_{jmt} := 0, \forall j, m, t$; //lot-sizes

$\alpha_{mt} := 0, \beta_{mt} := 0, e_{mt} := 0, \forall m, t$; //cross-period setup time and idle time

$CST := 0$; //cumulated setup time

$i_m := 0, \forall m$; //initial setup state

$seq_{mt} := \phi, \forall m, t$; //producing sequence

$t_m := T, \forall m$; //start at the planning horizon

Compute $D_{jt}, \forall j, t$; //cumulated demand

Compute $nd_{jt}, \forall j, t$; //next-demand period

Compute TD ; //total demand

Compute $\tilde{C}_{mt}, \forall m, t$; //cumulated capacity

Do While $TD > 0$ and $t > 0$

If $TD > \tilde{C}_t$ or $CST > u^{st}$ then **GoTo** *Initialization*;

Selection:

$t := \min\{ t_1, t_2, \ldots, t_m \}$;

Determine $\rho_{jmt}(i_m, t_m), \forall j, m$;

If $\sum_j \sum_m \rho_{jmt}(i_m, t_m) = 0$ **then**

$e_{mt} := C_{mt}, \forall m$; // let idle time be the remaining available time

$t_m := t_m - 1, \forall m$; // update time index

GoTo *Selection*;

Else

Choose j^* and m^* at random proportional to $\rho_{jmt}(i_m, t_m)$;

End

Setup:

If $s_{j^*,i_{m^*},m^*} < C_{m^*,t}$ **then**

$C_{m^*,t} := C_{m^*,t} - s_{j^*,i_{m^*},m^*}$; // update the capacity

If $seq_{m^*,t} = \phi$ **then** $\beta_{m^*,t} := s_{j^*,i_{m^*},m^*}$;

Else

$\alpha_{m^*,t} := \min\{ C_{m^*,t} , s_{j^*,i_{m^*},m^*} \}$;

$\beta_{m^*,t} := \min\{ s_{j^*,i_{m^*},m^*} - C_{m^*,t} , 0 \}$;

$t_{m^*} := t_{m^*} - 1$;

End

Production:

If $j^* = i_{m^*}$ **then** // check whether "jump-back" is required

$e_{m^*,\sigma} := C_{m^*,\sigma}$, $\sigma = t_{m^*}, t_{m^*} - 1, \ldots , nd_{j^*,t_{m^*}} + 1$;

$t_{m^*} := nd_{j^*,t_{m^*}}$; // jump-back

End

$t := t_{m^*}$;

Do While $D_{j^*,t} > 0$

$x_{j^*,t} := \min\{ C_{m^*,t} , D_{j^*,t} \}$; // assign the production quantity

$C_{m^*,t} := C_{m^*,t} - x_{j^*,t}$; // update capacity

$\tilde{C}_{m^*,t} := \tilde{C}_{m^*,t} - x_{j^*,t}$; // update cumulative capacity

$D_{j^*,\sigma} := D_{j^*,\sigma} - x_{j^*,t}$, $\sigma = 1, 2, \ldots , t$; // update cumulative demands

$TD := TD - x_{j^*,t}$; // update the total demand

$seq_{m^*,t} := (j^* , seq_{m^*,t})$; // insert j^* into the head of sequence

If $C_{m^*,t} = 0$ **then** $t_{m^*} := t_{m^*} 1$;

End

$i_{m^*} := j^*$;

End

If If $TD > \sum_m \tilde{C}_{m,t_m}$ or $CST > u^{st}$ then **GoTo** *Initialization*;

For $m = 1, 2, \ldots , M$

Do While $C_{m,t_m} > 0$

$e_{t_m} := C_{m,t_m}$; // let idle time be the remaining available time

$t_m := t_m - 1$;

End

Evaluate schedule ;

End

References

Almada-Lobo, B., Oliveira, J. and Carravilla, M.A. (2008), A note on "the capacitated lot-sizing and scheduling problem with sequence-dependent setup costs and setup times", *Computers and Operations Research*, 35, 1374-1376.

Barany, I., Van Roy, T.J. and Wolsey, L.A. (1984), Strong formulations for multi-item capacitated lot-sizing, *Management Science*, 30(10), 1255-61.

Bitran, G.R. and Matsuo, H. (1986), Approximation formulations for the single-product capacitated lot-size problem, *Operations Research*, 34, 63-74.

Bitran, G.R. and Yanasse, H.H. (1982), Computational complexity of the capacitated lot-size problem, *Management Science*, 28(10), 1174-1186.

Billington, P.J., McClain, J.O. and Thomas, L.J. (1983), Mathematical programming approaches to capacity constrained MRP systems: review, formulation and problem reduction, *Management Science*, 29, 1126-1141.

Buschkühl, L., Sahling, F., Helber, S. and Tempelmeier, H. (2010), Dynamic capacitated lotsizing problems: a classification and review of solution approaches, *OR Spectrum*, 32 (2), 231-261.

Carreno, J.J., (1990), Economic lot scheduling for multiple products on parallel identical processors, *Management Science*, 36, 348-358.

Chen, W.H. and Thizy, J.M. (1990), Analysis of relaxations for the multi-item capacitated lot-sizing problem, *Annals of Operations Research*, 26, 29-72.

Drexl, A. and Haase, K. (1995), Proportional lotsizing and scheduling, *International Journal of Production Economics*, 40, 73-87.

Drexl, A. and Haase, K. (1996), Sequential-analysis based randomized regret-methods for lot-sizing and scheduling, *Journal of the Operational Research Society*, 47, 251-265.

Drexl, A. and Kimms, A. (1997), Lot-sizing and scheduling – Survey and extensions, *European Journal of Operational Research*, 99, 221-235.

Eisenhut, P.S. (1975), A dynamic lot-sizing algorithm with capacity constraints, *IIE Transactions*, 7(2), 170–6.

Eppen, G.D. and Martin, R.K. (1987), Solving multi-item capacitated lot-sizing problems using variable redefinition, *Operations Research*, 35(6), 832-48.

Fandel, G. and Stammen-Hegene, C. (2006), Simultaneous lot-sizing and scheduling for multi-product multi-level production, *International Journal of Production Economics*, 104(2), 308-316.

Fleischmann, B. (1990), The discrete lot-sizing and scheduling problem, *European Journal of Operational Research*, 44(3), 337-348.

Florian, M., Lenstra, J.K. and Rinnooy Kan, A.H.G. (1980), Deterministic production planning algorithms and complexity, *Management Science*, 26(7), 669-679.

Gupta, D. and Magnusson, T. (2005), The capacitated lot-sizing and scheduling problem with sequence-dependent setup costs and setup times, *Computers and Operations Research*, 32, 727-747.

Haase, K. (1994), *Lotsizing and Scheduling for Production Planning*, Lecture Notes in Economics and Mathematical Systems, vol. 408, Springer, Berlin.

Haase, K. (1996), Capacitated lot-sizing with linked production quantities of adjacent periods, *Technical report*, Working Paper 334, Institut für Betriebswirtschaftslehre, Christian-Albrechts-Universität zu Kiel, Germany.

Haase, K. and Kimms, A. (2000), Lot-sizing and scheduling with sequence-dependent setup costs and times and efficient rescheduling opportunities, *International Journal of Production Economics*, 66, 159-169.

Kaczmarczyk, W. (2009), Practical Tips for Modelling Lot-Sizing and Scheduling Problems, *Decision Making in Manufacturing and Sciences*, 3, 37-48.

Karimi, B., Fatemi Ghomi, S.M.T. and Wilson, J.M. (2003), The capacitated lot-sizing

problem a review of models and algorithms, *Omega*, 31, 365-378.

Karmarkar, U.S., Kekre, S. and Kekre, S. (1987), The dynamic lotsizing problem with startup and reservation costs, *Operations Research*, 35(3), 389-398.

Kimms, A. (1996), Multi-level, single-machine lot-sizing and scheduling (with initial inventory), *European Journal of Operational Research*, 89, 86-99.

Kirca, O. and Kokten, M. (1994), A new heuristic approach for the multi-item dynamic lot-sizing problem, *European Journal of Operational Research*, 75(2), 332-41.

Kovács, A., Brown, K.N. and Tarim, A. (2009), An efficient MIP model for the capacitated lot-sizing and scheduling problem with sequence-dependent setups, *International Journal of Production Economics*, 118, 282-291.

Lasdon, L.S. and Terjung, R.C. (1971), An efficient algorithm for multi-item scheduling, *Operations Research*, 19(4), 946-969.

Leung, J.M.Y., Magnanti, T.L. and Vachani, R. (1989), Facets and algorithms for capacitated lot-sizing, *Mathematical Programming*, 45, 331–59.

Maes, J. and Van Wassenhove, L. (1988), Multi-item single-level capacitated dynamic lot-sizing heuristics a general review, *Journal of the Operational Research Society*, 39(11), 991–1004.

Maes, J. and Van Wassenhove, L.N. (1986), A simple heuristic for the multi-item single level capacitated lot-sizing problem, *Operations Research Letters*, 4(6), 265-73.

Maes, J., McClain, J.O. and Van Wassenhove, L.N. (1991), Multilevel capacitated lotsizing complexity and LP-based heuristics, *European Journal of Operational Research*, 53, 131-148.

Meyr, H. (2002), Simultaneous lotsizing and scheduling on parallel machines, *European Journal of Operational Research*, 139, 277-292.

Mohammadi, M., Fatemi Ghomi, S.M.T., Karimi, B. and Torabi, S.A. (2010),

MIP-based heuristics for lotsizing in capacitated pure flow shop with sequence-dependent setups, *International Journal of Production Research*, 48(10),. 2957-2973.

Salomon, M., Kroon, L.G., Kuik, R. and van Wassenhove, L.N. (1991), Some extensions of the discrete lotsizing and scheduling problem, *Management Science*, 37, 801-812.

Sarker, B.R. and Yu, J. (1996), Lot-sizing and cyclic scheduling for multiple products in a flow shop, *Computers and Industrial Engineering*, 30(4), 799-808.

Sox, C.R. and Gao, Y. (1999), The capacitated lot-sizing problem with setup carry-over, *IIE Transactions*, 31(2), 173-181.

Staggemeier, A.T. and Clark, A.R. (2001), A Survey of lot-sizing and scheduling models, *23rd Annual Symposium of the Brazilian Operational Research Society*, Campos do Jordão, Brazil.

Suerie, C. and Stadtler, H. (2003), The capacitated lot-sizing problem with linked sizes, *Management Science*, 49(8), 1039-1054.

Sung, C.S. (1986), A single-product parallel-facilities production-planning model, *International Journal of Systems Science*, 17, 983-989.

Süral, H., Denizel, M., and Van Wassenhove, L.N. (2009), Lagrangean relaxation based heuristics for lot-sizing with setup times, *European Journal of Operational Research*, 194, 51-63.

Thizy, J.M. and Van Wassenhove, L.N. (1985), Lagrangean relaxation for the multi-item capacitated lot-sizing problem a heuristic implementation, *IIE Transactions*, 17(4), 308-313.

Toledo, F.M.B. and Armentano, V.A. (2006), A Lagrangian-based heuristic for the capacitated lot-sizing problem in parallel machines, *European Journal of Operational Research*, 175, 1070-1083.

Trigeiro, W.W. (1989), A simple heuristic for lot-sizing with setup times, *Decision Science*, 20, 294-303.

Trigeiro, W.W., Thomas, L.J. and McClain, J.O. (1989), Capacitated lot-sizing with setup times, *Management Science*, 35(3), 353-366.