Synchronized scheduling of air transportation and assembly manufacturing in consumer electronics supply chain

Li, Kunpeng

2005

Li, K. (2005). Synchronized scheduling of air transportation and assembly manufacturing in consumer electronics supply chain. Doctoral thesis, Nanyang Technological University, Singapore.

<https://hdl.handle.net/10356/6548>

<https://doi.org/10.32657/10356/6548>

Nanyang Technological University

Downloaded on 26 Apr 2023 18:32:04 SGT

**SYNCHRONIZED SCHEDULING OF AIR
TRANSPORTATION AND ASSEMBLY MANUFACTURING
IN CONSUMER ELECTRONICS SUPPLY CHAIN**

LI KUNPENG

**SCHOOL OF MECHANICAL & AEROSPACE ENGINEERING
NANYANG TECHNOLOGICAL UNIVERSITY**

2005

**Synchronized Scheduling of Air Transportation and
Assembly Manufacturing in Consumer Electronics
Supply Chain**

Li Kunpeng

School of Mechanical and Aerospace Engineering

A thesis submitted to the Nanyang Technological University

in fulfillment of the requirements for the degree of

Doctor of Philosophy

2005

ABSTRACT

The consumer electronics has been a booming industry in recent years. Motivated by the challenges confronted by a major PC manufacturer in consumer electronics industry, this research studies the problem of synchronizing the assembly manufacturing and the air transportation to achieve accurate delivery with minimized delivery cost. There are two integrated decisions involved in the research problem that determine: (1) proper allocation of orders to various available flights in the planning period (2) an appropriate release time for each order in assembly ensuring it can catch the allocated flight after assembly, and to minimize the waiting time between assembly and air transportation.

A two stage Synchronized Scheduling Methodology (SSM) is proposed. SSM decomposes the overall problem into air transportation allocation problem and the assembly scheduling problem. The two sub-problems are then formulated considering synchronization with each other. The air transportation problem is classified into four categories according to different assumptions on transportation destination and order split for delivery. The four types of air transportation problems are formulated using integer linear programming models separately. The assembly scheduling problem is first formulated as a single machine scheduling problem. Subsequently, the parallel machine assembly scheduling problems are addressed. In addition, the single machine assembly scheduling problem considering stochastic machine breakdowns is formulated. Schedule repair and reschedule algorithms are developed to solve this problem.

Numerical experiments show that SSM can achieve better performance compared to a popular industry practice heuristic in terms of lower air transportation cost and reduced

waiting time before air transportation. A real-life supply chain case is also studied to demonstrate the application of the proposed methodology. Results indicate that the schedule generated by SSM is superior to the current schedule of a global electronics manufacturer in terms of significantly reduced waiting cost before air transportation, capability to avoid transportation backlog, and total air transportation costs reduction.

ACKNOWLEDGEMENTS

I would like to express my deep gratitude to my supervisor, Dr. Appa Iyer Sivakumar, for his guidance, encouragement and inspiration throughout the progress of this research as well as the writing of this thesis.

I am grateful to Dr. M.Mathirajan and Dr. Viswanath Kumar Ganesan, who spent much time in discussion, and gave numerous valuable suggestions during the progress of this research. I also would like to thank Dr. Linet Özdamar, Dr. Rajesh Piplani for their helpful comments and valuable advices to my first year report. I extend my thanks to the anonymous examiners for their thorough review, critical comments and constructive suggestions to improve the presentation of this thesis.

I wish to thank the research students in Center for Supply Chain Management (CSCM) for aspiring discussions, various suggestions and continuous encouragement. I would also like to express my appreciation to Ms Leong Yuit Lin for her support and assistance during my stay in CSCM. I would like to thank my cousin Mr. Liu Fangrui and all my friends in Singapore for their help in one way or another.

Finally, I would like to express my deepest love and appreciation to my fiancée Ms Wang Dan. Without her endless support, this research would have never been possible.

TABLE OF CONTENTS

Abstract.....	I
Acknowledgments	III
Table of Contents	IV
List of Figures.....	VIII
List of Tables	X
List of Abbreviations	XII
List of Variables	XIII
Chapter 1: Introduction	1
1.1 Background.....	1
1.2 Motivation.....	4
1.3 Structure of Consumer Electronics Supply Chain	6
1.4 CESC Synchronization Problem.....	8
1.5 Research Objectives.....	11
1.6 Methodology	12
1.6.1 Synchronization in CESC	13
1.6.2 The Air Transportation Problem.....	13
1.6.3 The Assembly Scheduling Problem.....	14
1.7 Organization of the Thesis	15
Chapter 2: Literature Review.....	19
2.1 Production and Distribution Integration	19
2.1.1 Production/Transportation Integration Problem	20
2.1.2 Production/Inventory/Distribution Integration Problem.....	23
2.2 Outbound Logistics.....	26
2.2.1 Air transportation Scheduling Problem	26
2.2.2 Vehicle Routing Problems	27

2.3 Single and Parallel Machine Scheduling with Earliness and Tardiness Penalties.....	28
2.3.1 Single Machine Scheduling Problem with Earliness and Tardiness.....	29
2.3.2 Parallel Machine Scheduling Problem with Earliness and Tardiness.....	41
2.3.3 Techniques for Solving the Earliness Tardiness Scheduling Problems.....	47
2.4 Summary	50
Chapter 3: The Air Transportation Problems	54
3.1 Introduction and Assumptions	54
3.2 Type 1 Air Transportation Problem.....	56
3.2.1 Problem Formulation	57
3.2.2 Experimental Design.....	61
3.2.3 Computational Complexity Analysis.....	63
3.3 Type 2 Air Transportation Problem.....	66
3.3.1 Problem Formulation	66
3.3.2 Computational Intractability of Type 2 Air Transportation Problem	68
3.3.3 NP-Completeness of Type 2 Air Transportation Problem	69
3.4 Type 3 Air Transportation Problem.....	73
3.4.1 Problem Formulation	73
3.4.2 Experimental Design.....	75
3.4.3 Computational Complexity Analysis.....	77
3.5 Type 4 Air Transportation Problem.....	79
3.5.1 Problem Formulation	79
3.5.2 NP-Completeness of Type 4 Air Transportation Problem	81
3.6 Summary	81
Chapter 4: The Single Machine Assembly Scheduling Problem.....	83
4.1 Introduction and Assumptions	83
4.2 Solution Methodologies	84
4.2.1. Forward Scheduling Algorithm	86
4.2.2 Backward Scheduling Algorithm.....	89
4.3 Detail of the Synchronized Scheduling Methodology	92
4.4 Computational Results	94

4.4.1 Results and Analysis for Synchronizing Type 1 Air Transportation and the Single Machine Assembly	95
4.4.2 Results and Analysis for Synchronizing Type 3 Air Transportation and the Single Machine Assembly	99
4.5 Managerial Implications	104
4.6 Summary	105
Annexure: Demonstration of Synchronized Scheduling Methodology by A Numerical Example	107
Chapter 5: The Parallel Machine Assembly Scheduling Problem	112
5.1 Introduction and Problem Formulation.....	112
5.2 PENDSP with Independent Weights	114
5.2.1 PENDSP with Independent Weights and Split Processing.....	115
5.2.2 PENDSP with Independent Weights and Non-split Processing	116
5.3 PENDSP with Dependent Weights.....	125
5.3.1 PENDSP with Dependent Weights and Split Processing	126
5.3.2 Formulation of PENDSP with Dependent Weights and Non-split Processing .	129
5.3.3 NP-Completeness of PENDSP with Dependent Weights and Non-split Processing	131
5.3.4 Solution Methodologies for PENDSP with Dependent Weights and Non-split Processing	136
5.3.5 Experimental Design.....	151
5.3.6 Computational Results	151
5.4 Summary	160
Annexure: Demonstration of Probabilistic Nature of PSAD	162
Chapter 6: The Single Machine Assembly Scheduling Problem with Breakdowns ...	164
6.1 Problem Description	164
6.2 Problem Formulation	165
6.3 NP-Hardness of the Problem	170
6.4 Solution Methodologies	172
6.4.1 Schedule Repair Method.....	172

6.4.2 Reschedule Method using Simulated Annealing	174
6.5 Experimental Design.....	176
6.6 Computational Results	178
6.6.1 Comparison of the Remaining Optimal Heuristic with Two Alternative Schedule Repair Heuristics.....	178
6.6.2 Comparison of the Remaining Optimal Heuristic with PSAD	185
6.7 Summary	189
 Chapter 7: Application of the Synchronized Scheduling Methodology to A Supply Chain Case.....	190
7.1 Supply Chain of GEM Singapore	190
7.2 Challenges Confronted by GEM Singapore	192
7.3 Application of SSM to the Supply Chain of GEM Singapore.....	194
7.3.1 Input Data	195
7.3.2 Results Analysis.....	197
7.4 Summary	204
Annexure: Input Data of Day 1.....	205
 Chapter 8: Summary, Conclusions and Future Work	211
8.1 Summary and Conclusions	211
8.2 Limitations	214
8.3 Future Work.....	215
 Reference	219
Publications	238

LIST OF FIGURES

Figure 1.1 MTS strategy (I: Inventory)	3
Figure 1.2 MTO strategy (Q: Queue)	4
Figure 1.3 Material flow through CESC.....	6
Figure 1.4 Structure of CESC	7
Figure 1.5 Two stage decision model in CESC	8
Figure 1.6 Synchronization between assembly scheduling and air transportation allocation.....	13
Figure 1.7 Organization of the thesis.....	16
Figure 2.1 Comparison of published literature and assembly scheduling problems in CESC	52
Figure 2.2 Comparison of published literature and air transportation allocation problem in CESC	53
Figure 3.1 Type 1 air transportation problem	56
Figure 3.2 Transportation allocation and the accumulated assembly capacity.....	59
Figure 3.3 Processes from assembly to delivery.....	60
Figure 3.4 Equality of Type 1 air transportation problem to the unbalanced transportation problem.	65
Figure 3.5 Type 2 air transportation problem	66
Figure 3.6 Type 3 air transportation problem	73
Figure 3.7 Type 4 air transportation problem	79
Figure 4.1 The schedule generated by the forward scheduling algorithm.....	88
Figure 4.2 The schedule generated by the backward scheduling algorithm.....	91
Figure 4.3 Flow diagram of SSM	93
Figure 5.1 The assembly flow shop	112
Figure 5.2 A special case of PENDSP with non-split processing	117
Figure 5.3 Demonstration of machine unavailable time.....	121
Figure 5.4 Schedule on machine m within σ	127
Figure 5.5 Search procedure of BSA in the search space.....	140
Figure 5.6 Search procedure of PSA in the search space	140

Figure 5.7 Search procedure of PSAD in the search space	143
Figure 6.1 Procedure to solve the single machine assembly scheduling problem with breakdowns	167
Figure 6.2 The remaining optimal heuristic.....	173
Figure 6.3 The append to last heuristic.....	178
Figure 6.4 The Shift heuristic	179
Figure 7.1 The market distribution in Asia-Pacific region	191
Figure 7.2 Comparison between SSM, GEM's scheduling method and 'MIP+FCFS' ...	199

LIST OF TABLES

Table 3.1 Experimental design used in random problems generation.....	61
Table 3.2 Computational times for tested Type 1 air transportation problems	63
Table 3.3 Computational times for tested Type 2 air transportation problems	68
Table 3.4 Experimental design for modified and newly introduced parameters.....	75
Table 3.5 Computational times for tested Type 3 air transportation problems	78
Table 4.1 Order processing times and air transportation allocation	88
Table 4.2 Flight departure times	88
Table 4.3 Performance of the proposed backward / forward scheduling algorithm in synchronizing Type 1 air transportation and the single machine assembly	96
Table 4.4 Performance of SSM and EDD+FCFS heuristic in synchronizing Type 1 air transportation and the single machine assembly.....	98
Table 4.5 Performance of the proposed backward / forward scheduling algorithm in synchronizing Type 3 air transportation and the single machine assembly	99
Table 4.6 Performance of SSM and EDD+FCFS heuristic in synchronizing Type 3 air transportation and the single machine assembly.....	101
Table 4.7 Cost reduction of the ILP model to EDD+FCFS heuristic	102
Table 5.1 Order processing times and air transportation allocation	120
Table 5.2 Value for algorithm dependent parameters.....	150
Table 5.3 Cost performances of the four algorithms	153
Table 5.4 Performances of the four algorithms in terms of computational time	155
Table 5.5 Computational results comparison of adapted BSA ($\delta=0.01$) with PSAD.....	158
Table 5.6 Computational results comparison of adapted BSA ($\delta=0.1$) with PSAD ^a	158
Table 6.1 Value for algorithmic parameters of PSAD.....	175
Table 6.2a Performances of the schedule repair heuristics with breakdown1	181
Table 6.2b Evaluation of the schedule repair heuristics with breakdown1	181
Table 6.3a Performances of the schedule repair heuristics with breakdown2.....	182
Table 6.3b Evaluation of the schedule repair heuristics with breakdown2	182
Table 6.4a Performances of the schedule repair heuristics with breakdown3.....	183
Table 6.4b Evaluation of the schedule repair heuristics with breakdown3	183

Table 6.5a Performances of the schedule repair heuristics with breakdown4.....	184
Table 6.5b Evaluation of the schedule repair heuristics with breakdown4	184
Table 6.6 Performance comparison of the remaining optimal heuristic and PSAD with breakdown1	185
Table 6.7 Performance comparison of the remaining optimal heuristic and PSAD with breakdown2.....	186
Table 6.8 Performance comparison of the remaining optimal heuristic and PSAD with breakdown3.....	186
Table 6.9 Performance comparison of the remaining optimal heuristic and PSAD with breakdown4.....	187
Table 7.1 Destination cities and the transportation times	195
Table 7.2 Air transportation costs with large available flight capacity	200
Table 7.3 Waiting costs in hub with large available flight capacity.....	200
Table 7.4 Air transportation costs with constrained flight capacity	202
Table 7.5 Waiting costs in hub with constrained flight capacity	203

LIST OF ABBREVIATIONS

ATO: Assemble-to-Order

BSA: Basic Simulated Annealing

CESC: Consumer Electronics Supply Chain

EDD: Earliest Due Date

EDD+FCFS: the existing industry practice heuristic, EDD for assembly and FCFS for air transportation.

E/T: Earliness and Tardiness

FCFS: First Come First Serve

GEM: Global Electronics Manufacturer

ILP: Integer Linear Programming

LPT: Longest Processing Time

LP: Linear Programming

MIP: Mixed Integer Programming

MTO: Make-to-Order

MTS: Make-to-Stock

PENDSP: Parallel machine Earliness Non-common Due date Scheduling Problem

PSA: Population based Simulated Annealing

PSAD: Population based Simulated Annealing with Diversification

PSAID: PSA Iterate by Diversification

SA: Simulated Annealing

SSM: Synchronized Scheduling Methodology

TSP: Traveling Salesman Problem

VRP: Vehicle Routing Problem

LIST OF VARIABLES

d = a common due date

dd_i = delivery due date of order i

d_j = assembly due date of job j

dp = a demand point

dt =downtime, i.e., the time of machine breakdown

$d\alpha_i$ = delivery earliness penalty cost (/unit/hour) of order i

$d\beta_i$ = delivery tardiness penalty cost (/unit/hour) of order i

e = per unit per hour earliness penalty cost in assembly

f = flight index, $f=1,2,\dots,F$

g = destination index, $g=1,2,\dots,L$

i = order index in the air transportation problem, $i = 1,2,\dots,O$

j = job index in the assembly scheduling problem, $j = 1,2,\dots,N$

l = iteration index in the searching of SA

m = machine index in the parallel machine assembly scheduling problem, $m=1,2,\dots,M$

n = a given positive integer

p_i = processing time of order i in the single machine assembly

p_j = processing time of job j in the single machine assembly

p_{jm} = processing time of job j on machine m in assembly

r = the cooling rate

s = index of the idle time spans, $s = 1,2,\dots,S$

sp = a supply point sp

t = the minimum acceptable value of temperature

tr = the tolerance of due date

w_j = earliness penalty weight of job j (per hour) in assembly

γ = the fixed coefficient denoting the relationship between job processing time and its weight of earliness penalty

α_j = the weight of the unit time earliness penalty of job j

α_{lj} = earliness penalty of job j for assembly

β_j = the weight of the unit time tardiness penalty of job j

β_{lj} = per unit transportation cost of job j that transported by commercial flight

ω = the backlog percentage generated by the EDD+FCFS heuristic

θ = the coefficient, which is the results that the average delivery cost of per unit backlogged product divided by the average delivery cost of per unit no backlog product

σ = is a schedule of the parallel machine assembly

σ_a = an allocation for the air transportation problem

λ_m = the proportion of machine m 's production rate to the overall production rate

$\chi_{jmk} = 1$ if job j is processed on machine m , and there are $k-1$ jobs processed before it, 0 otherwise.

δ = a parameter that controls the rate of cooling in the SA algorithm presented by Radhakrishnan and Ventura (2000)

ε = a small number

τ = a given positive number

ρ = a given positive number

A_f = arrival time of flight f at the destination

B = a given positive integer

BL = the best number of solutions chosen among the neighborhood population of the current solution

BS = the best solution found from the beginning of searching

C_j = assembly completion time of job j

C_{nm} = assembly completion time of the last job on machine m

D_f = departure time of flight f at the local place where the manufacturing plant locates

Des_i = order i 's destination

Des_f = flight f 's destination

DN = diversification number

DU = duration of breakdown

E_j = length of time that job j is early

$EF_{jf} = 1$ if job j is earlier than flight f 's departure time, i.e., job j can catch flight f .
0 otherwise.

F = total number of flights

FN_f = flight number assigned to a flight among the flights that have the same destination

G = a set of positive integers $\{x_1, x_2, \dots, x_n\}$ with $\sum_{i=1}^n x_i = 2B$,

G' = a job set that corresponds to G

G_I = a subset of G such that $\sum_{x_i \in G_I} x_i = \sum_{x_i \in G - G_I} x_i = B$

G_I' = a job set that corresponds to G_I

$HX_{if} = 1$ if order i allocated to flight f 's normal capacity, 0 otherwise.

$HY_{if} = 1$ if order i allocated to flight f 's special capacity, 0 otherwise.

J_{jm} = a partial job of job j

L = total number of destinations

LN = a large positive number

$|LN|$ = absolute value of LN

MDN = maximum diversification number

MNI = max number of no improve

N = total number of jobs

N_d = number of disturbed jobs by machine breakdown

N_r = number of remaining unprocessed jobs when breakdown occurs

NC_f = transportation cost for per unit product which allocated to normal capacity of flight f

$NCap_f$ = available normal capacity of flight f

NI = iterations of no improvement found in SA

O = order numbers in the air transportation problems

P_i = priority of order i

PA_{jf} = predetermined air transportation allocation by the ILP model, 1 if job j is allocated to flight f , 0 otherwise.

PE_{if} = per unit delivery earliness penalty cost for order i when it is transported by flight f

$PI_{ij} = 1$ if job i precedes job j immediately, 0 otherwise

$PI_{ijm} = 1$ if job i precedes job j immediately on machine m , 0 otherwise.

PL_{if} = per unit delivery tardiness penalty cost for order i when it is transported by flight f

PR = the total production rate of assembly

Q_i = quantity of order i

R_j = release time of job j

R_{1m} = the release time of the first job on machine m

S = idle time spans from time $dt+DU$ in the schedule determined by the backward scheduling algorithm

SC_f = transportation cost for per unit product which allocated to special capacity of flight f

$SCap_f$ = available special capacity of flight f

SP_k = sub-problem k of the parallel machine assembly scheduling problem

T = the temperature in SA algorithms

T_{AL} = the number of tardy jobs obtained by applying append to last heuristic

T_{Shift} = the number of tardy jobs obtained by applying Shift heuristic

T_{RO} = the number of tardy jobs obtained by applying remaining optimal heuristic

TA_i = the time when order i arrivals airport

TC_1 = the total cost generated by the EDD+FCFS heuristic

TC_2 = the total cost generated by the ILP model

TC_{if} = transportation cost matrix which is determined by the ILP model.

TC_{cur} = current solution.

TC_{best} = the best solution.

TDA_f = transportation time of flight f from departure to arrival

TF_g = the total number of flights that have the same destination g

TI = searching iterations at each value of T in SA algorithms

TIS = searching iterations at each searching stage in the SA algorithm presented by Radhakrishnan and Ventura (2000)

TQ = total quantity of all the orders

U_m = unavailable time of machine m

V_{jm} = 1 if job j is assigned to machine m , 0 otherwise

WT_i = waiting time of order i between assembly manufacturing and air transportation

X_{if} = the quantity of the portion of order i allocated to flight f 's normal capacity

Y_{if} = the quantity of the portion of order i allocated to flight f 's special capacity

Z_{if} = quantity of order i allocated to flight f

ΔZ = the difference between the two steps in an uphill move

ΔZ_{min} = the smallest uphill move found in 100,000 moves

ΔZ_{max} = the largest uphill move found in 100,000 moves

Chapter 1: Introduction

1.1 Background

Consumer electronics products, which contribute a major share in many nations' GNP, are playing an important role in our daily life. Almost every home has consumer electronics products such as a videocassette recorder, a compact disk player, or a digital videodisk player (Han et al. 2001). Thirty years ago, electronic calculators were beginning to penetrate mass markets in rich countries. Today, around half a billion people are using consumer electronics products (Ayres and Williams, 2004). Most of consumer electronics products are knowledge-intensive goods with high value-to-weight ratios, which comprise a growing share of global trade (Bowen, 2004). All these indicate that consumer electronics is a booming industry.

Today's consumer electronics manufacturing companies have to face the challenge of volatile demand, shorter product life cycles, product customization, time to market and cost reduction. To meet these challenges, companies are moving toward global manufacturing network. Air cargo services enhance the performance of the global manufacturing networks. Southeast Asia is a region that is distant from the world's principle economies and important as an electronics export platform. Air cargo services are vital to the incorporation of peripheral regions like Southeast Asia into the international economy (Leinbach and Bowen, 2004). Furthermore, due to the high value of consumer electronics products and to meet the promised short delivery time, air

transportation is commonly used in consumer electronics products distribution (Bowen, 2004). Electronics and computers is the largest single air-freight category, comprising nearly 20% of the market (U.S. Bureau of Transportation Statistics, 2000).

Cooperation with all the participants within the manufacturing network and effective utilization of global resources benefit the manufacturers to gain competition advantages. Operations in this global manufacturing network require co-ordination of a global network of manufacturing units, warehouses or distribution points; the optimization of routings and logistics (Azevedo and Sousa, 2000). Within the Consumer Electronics Supply Chain (CESC), synchronization of manufacturing with transportation is especially important for consumer electronics manufacturers to improve performance with lower cost.

Furthermore, the competition among the consumer electronics industries forces a kind of just-in-time manufacturing philosophy to avoid both earliness and tardiness. Costs or penalties are incurred by delivering an order either earlier or later than the due date. The delivery earliness costs could result from the need for storage and insurance. The delivery tardiness cost consists of customer dissatisfaction, contract penalties, loss of sales, and loss of reputation. Orders transferred to the airport ahead of flights departure times incur waiting penalties. The penalties are particularly for handling and storage of the goods in the airport. Unlike the basic assembly and transportation cost of the products, these penalty costs can be minimized by achieving better synchronization in CESC.

Different consumer electronics manufacturers have different strategies to manage the supply chain. In general, there are three strategies: Make-to-Stock (MTS), Make-to-Order (MTO), and a hybrid of MTO and MTS (Williams, 1984; Rajagopalan, 2002). Figure 1.1 illustrates the MTS strategy, which has attracted much attention for many years. Manufacturing is based on forecasting. Finished goods are stored in inventory. When an order comes, products are transferred to distributors, retailers or deliver directly to customers from the finished goods inventory.

Figure 1.1 MTS strategy (I: Inventory)

Generally, the MTS strategy with respect to CESC has significant weaknesses. This is due to the fact that the consumer electronics products' life cycle is short, demand variance is high and forecasting is difficult as the products are mostly customized. In general, finished goods inventory cost is a significant part of the product cost. The time in inventory is also undesired as the product life cycle is short. Furthermore, there exist risk of losing customer demand and risk of finished products perishing in inventory due to the incorrect forecasting.

Recently, researchers and industry practitioners started to pay more attention in developing MTO based supply chain management strategy, particularly in the consumer electronics industry. Figure 1.2 depicts the MTO strategy.

Figure 1.2 MTO strategy (Q: Queue)

Since the production in CESC is mainly assembly, the more appropriate definition is Assemble-to-Order (ATO) for this strategy. A manufacturing strategy where materials and subassemblies are made or acquired according to forecasts, while the final assembly of product is delayed until customer orders have been received is commonly referred to as ATO (Gerchak and Henig 1989; Guerrero, 1991; Maruchek and McClelland, 1992; Fumero and Vercelis, 1994; Harris and Pinder, 1995; Hsu and Wang, 2001). As the assembly manufacturing is based on acceptance of customer orders other than forecasting, and delivery is carried out without finished goods inventory, ATO strategy avoids wrong forecasting, finished goods inventory costs and the risk of finished product perishing, which are the weaknesses of MTS strategy.

1.2 Motivation

Considering the features of relative fixed route, fixed departure time and stable transportation time, air transportation scheduling is different from the vehicle routing scheduling problems classified by Destrochers et al. (1990). Unlike the popular vehicle routing scheduling problems, there has been a relative neglect of air transportation scheduling problems, even though it is a sizable sector of transportation. Meanwhile, the just-in-time production philosophy has led to a growing interest both in production

and transportation scheduling problems considering earliness and tardiness penalties. It is due to the fact of growing pressure of competition in the international markets: firms have to offer a great variety of different and individual products while customers expect ordered goods to be delivered on time. However, research in these two areas is mainly focused independently and there has been a relative neglect of synchronization of these two to achieve the delivery accuracy, but effective solution of this problem is significantly important for a consumer electronics manufacturer to gain competitive advantages.

This research is motivated by observations found in a large MTO based PC assembly manufacturer who faces challenges in synchronizing the supply chain. The company has its major assembly plant in Singapore. The observed industry receive their orders through many sources including email, World Wide Web, fax and phone. Orders arrive randomly, and the company commits the delivery time to the customers. Air transportation is commonly used for the distribution of high value ATO consumer electronics products to global customers. In general, commercial cargo flights are often used. The material flow through the supply chain is depicted in Figure 1.3.

Components manufacturing is outsourced to local and global suppliers. Procured components are stored in inventory before assembly. When an order is received, the components are transferred into assembly flow shop. After four major modules of assembly processes that consist of assembly, burn-in, test and packaging, finished products are transferred from assembly plant to airport by local transportation. Then, the products reach the airport of the destination country through global air

transportation. Finally, destination country local transportation delivers the products to customers. Due to the complexity of the supply chain, synchronization is a challenge.

Figure 1.3 Material flow through CESC

1.3 Structure of Consumer Electronics Supply Chain

As the material flow is depicted in Figure 1.3, the detailed structure of CESC is illustrated in Figure 1.4. In general, the assembly manufacturing of consumer electronics product is accomplished in a flow shop. There are several parallel assembly lines within the assembly flow shop. Each assembly line performs assembly manufacturing by four steps: assembly, burn-in, test and package. It is noted that the burn-in process is a batch processing procedure. Unlike the piece by piece processing for the remaining three steps, burn-in processes more than one piece together in a burn-in batch. After packaging process, products are transferred to the airport and queued to be transported to customers. Customers are located in different places, leading to

multiple destinations of air transportation. Multiple flights are available to each destination. For each flight, it has a single destination.

Figure 1.4 Structure of CESC

Based on the definition of supply chain management given by Simchi-levi et al. (2000), the CESC management in this research aims to release the right order to the assembly at the right time, the assembly manufacturing completed at the right time and the order catch the right flight and be delivered to the right customer at the right time, so as to minimize systemwide costs while satisfying service level requirements.

There are two integrated decisions involved in the CESC management: (1) the decision in air transportation determining the proper allocation of the orders to various available flights during the planning period. (2) The decision in assembly determining an appropriate release time for each order ensuring it can catch the allocated flight after assembly, meanwhile, the waiting time between assembly and transportation is minimized. Note that a planning period is defined as a time length, during which a set of orders are produced and delivered to customers. The two stage decision model in CESC is illustrated in Figure 1.5.

Figure 1.5 Two stage decision model in CESC

1.4 CESC Synchronization Problem

According to the two decision phases in CESC management, the overall synchronization problem can be decomposed into an air transportation allocation problem and an assembly scheduling problem.

For the air transportation problem, the main objective is to select the appropriate flights among the available ones to transport existing orders. It is common that there may be large number of flights available during the planning period. The transportation cost,

available capacity and delivery time of each flight differ. The selection of flights is based on the consideration of the transportation costs, available capacities and delivery times of the flights. For instance, some flights have low transportation cost. But the arrival times do not match well with the promised order delivery time, which will lead to early or late penalty. Therefore, there are two objectives involved in the air transportation problem: the delivery accuracy and transportation cost. Clearly, assembly capacity should be taken into account in the decision making procedure of air transportation allocation. It means that proper transportation allocation should be based on the condition of sufficient assembly capacity, i.e., the assembly of each order could be completed before the departure time of the allocated flight.

Moreover, different situations are considered in the formulation of the air transportation problem. One important issue concerns with order split in delivery. In some cases, customers accept split delivery. In this situation, a delivery to a customer can be split among any number of flights. While in other cases, customers expect the delivery be performed by only one flight. The other issue focuses on the destinations of transportation. If the demands come from a particular customer or the customers are located in a region that can be served from a single airport, there is only one destination of air transportation. The more general case is to consider multiple destinations, where customers are located in different regions. The formulation of the air transportation varies in different situations.

The overall assembly scheduling problem is complex in the context of the assembly flow shop that is described in Figure 1.4. The complexity of the problem is identified

from the parallel assembly lines, multiple stages of assembly process, and the batch processing stage within the flow shop. The assembly of each order should be completed early enough that the order can catch the allocated flight. Meanwhile, the order should not be completed too early. Otherwise, there will be a long waiting time, resulting in high intermediate inventory cost between assembly and transportation. In contrast, if the assembly of an order is completed late so that the order cannot catch its allocated flight, heavy late penalty is also incurred. The late penalty accounts either for hiring a chartered flight with significantly high cost or for delivering the order in the next planning period that leads to long delay time. Therefore, the assembly scheduling seeks to avoid both earliness and lateness penalties.

The assembly scheduling problem is also addressed in different situations. In some cases, in order to complete all the jobs as soon as possible, jobs can be split arbitrarily and processed on more than one assembly line in the flow shop (Xing and Zhang, 2000). For another situation which is commonly considered in the published work addressing machine scheduling problem, one job can only be processed on one assembly line. In addition, two conditions are considered for the intermediate inventory cost between assembly and transportation, which depend on the total number of jobs and total quantity of jobs respectively. The terms job independent/dependent weight represents these two conditions.

Based on this overview of the CESC synchronization problem, the solution methodology is developed in this research. The main objectives of the methodology, which is also the objective of this research, are presented in the following section.

1.5 Research Objectives

The major general objective of this research is to improve the performance of demand fulfillment in terms of reduced total delivery costs, and enhanced accuracy of delivery, with the help of synchronized scheduling of assembly manufacturing and air transportation within CESC. Based on the outline of the scheduling of assembly manufacturing and air transportation presented in section 1.4, the overall objective of this research is to develop a methodology to synchronize the scheduling of assembly manufacturing and air transportation in the make-to-order consumer electronics supply chain. As mentioned in section 1.4, the overall problem can be decomposed into air transportation problem and assembly scheduling problem. The details of the objectives are as follows:

- To develop an approach to decompose the overall CESC synchronization problem.
- To define and formulate the air transportation problem in the situations of single/multiple destinations and split/non-split deliveries.
- To study the complexities of the air transportation problem in different situations and identify approaches to solve the problem.
- To define and formulate the assembly scheduling problem in the situations of job split/non-split in processing and job dependent/independent weights for the waiting penalty between assembly and air transportation
- To study the complexities of the assembly scheduling problem in different situations and to develop exact and approximate methods to solve the problem.

The assembly scheduling problem studied in this research represents the scheduling problem in an assembly flow shop, which includes parallel assembly lines, batch processor, etc. As the overall assembly scheduling problem is complex, it is studied from the simplest situation of single machine assembly. Subsequently, the more complicated situation involving parallel assembly lines in assembly flow shop is investigated. Moreover, stochastic machine breakdowns are taken into account in the single machine assembly scheduling problem.

1.6 Methodology

The CESC synchronization problem is investigated in this research by adopting a two stage synchronized scheduling methodology namely SSM, which decompose the overall problem into two sub-problems: the air transportation allocation problem and the assembly scheduling problem. These two problems are formulated and solved considering synchronization with each other. The overall synchronization problem is solved by solving the air transportation problem first to obtain the transportation allocation. Then the assembly scheduling problem is solved to obtain the release time of each order.

The air transportation problem is formulated using an Integer Linear Programming (ILP) model. The assembly scheduling problem is modeled as a machine scheduling problem with earliness and tardiness penalties. The synchronization issues that are taken into account in the formulation of each sub-problem are illustrated below.

1.6.1 Synchronization in CESC

Most of published researches consider the production scheduling problem and the air transportation problem separately (Li et al. 2004). The novelty of this research is based on the synchronization of the two sub-problems. Synchronization is incorporated into the ILP model of the air transportation problem by one constraint that matches the flight allocation with assembly capacity. In other words, assembly of the allocated orders should be completed on time by sufficient assembly capacity. On the other hand, the assembly due date of each order is determined based on the departure time of the flight that the order is allocated to, which is in turn determined by the ILP model. The synchronization relationships between these two sub-problems are illustrated in Figure 1.6.

Figure 1.6 Synchronization between assembly scheduling and air transportation allocation

1.6.2 The Air Transportation Problem

In this research, the air transportation problem is classified into four categories according to different assumptions on transportation destination and order split for delivery. The four types of air transportation problems are formulated using ILP models separately. The studies on the computational complexity of these problems indicate that the condition of split delivery makes the problem easy and can be solved optimally by

the available mathematical programming software. In contrast, the condition of non-split delivery leads the air transportation problem to be NP-complete.

1.6.3 The Assembly Scheduling Problem

The assembly problem is first formulated as a single machine scheduling problem with earliness penalties. Backward scheduling algorithm is employed to provide the optimal solution for this problem. The backward scheduling algorithm is based on backward scheduling logic and Longest Processing Time (LPT) sequence rule. The advantage of this method is identified by comparing it with a heuristic widely applied in industry and a forward scheduling algorithm.

Subsequently, the assembly scheduling problem considering parallel machines is addressed. Different formulations are proposed based on different assumptions: job dependent/independent penalty weight and job split/non-split processing. Note that split processing means that a job can be split arbitrarily and processed on more than one machine. It is found that the condition of split processing imposes the backward scheduling algorithm applicable to the parallel machine assembly scheduling problem to obtain the optimal solution. Assuming non-split processing, the parallel machine assembly scheduling problem with job independent penalty weight is formulated as a transportation problem. Considering job dependent weight and non-split processing, the parallel machine assembly scheduling problem is formulated using a Mixed Integer Programming (MIP) model and proved to be NP-complete. Simulated Annealing (SA) based algorithms are employed to obtain near optimal solutions.

The assembly scheduling problem considering stochastic machine breakdowns is also studied in this research. The problem is formulated as a single machine scheduling problem with earliness tardiness penalties. Two methods are proposed for this problem: schedule repair method and reschedule method.

Finally, a real-life case study is conducted to demonstrate the efficiency of the proposed SSM methodology for solving the CESC synchronization problem.

1.7 Organization of the Thesis

Organization of the thesis is shown in Figure 1.7. The thesis consists of seven chapters. Chapter 2 reviews the published literature addressing the problem of integrated scheduling of production and distribution. This review also covers the research that deals with the scheduling issues in air transportation and land transportation. In addition, the literature addressing single and parallel machine scheduling with earliness and tardiness penalties is reviewed.

Chapter 3 proposes the formulations of four air transportation problems according to different assumptions on the destinations and split deliveries. Computational complexities of these problems are studied by both of the numerical experiments and the mathematical analysis.

Chapter 4 formulates the single machine assembly scheduling problem. Two solution methods, namely forward scheduling algorithm and backward scheduling algorithm are presented. Computational results indicate that backward scheduling algorithm performs

better compared to forward scheduling algorithm in reducing waiting time before transportation. The proposed methodology of SSM, which consists of the ILP formulation of the air transportation problem and backward scheduling algorithm for the single machine assembly scheduling problem, is also compared with a heuristic used in industry practice. Advantages of SSM over the industry practice heuristic are identified from the reduced delivery cost, waiting time before transportation and backlog orders.

Figure 1.7 Organization of the thesis

Chapter 5 considers parallel machines in assembly. Four problems are formulated according to the assumptions on split/non-split processing and job dependent/independent penalty weight. Computational complexity of each problem is also identified. Assuming split processing, optimal solution can be obtained by applying the backward scheduling algorithm. Assuming non-split processing and job independent weight, the parallel machine scheduling problem is decomposed into sub-problems. Then, each sub-problem is formulated as a transportation problem. For the problem considering non-split processing and job dependent weight, it is formulated using a MIP model and proved to be NP-complete. SA based algorithms are developed for this problem.

Chapter 6 studies the single machine assembly scheduling problem considering stochastic machine breakdowns. The problem is formulated using a MIP model and shown to be NP-hard. Two approaches are presented to solve this problem: schedule repair and reschedule approach. Problem with different breakdown times and various repair durations are tested by these two approaches. The computational results show that the performance of the reschedule approach is slightly superior to schedule repair approach.

The application of SSM to a real life CESC is discussed in chapter 7. The air transportation considers multiple destination and split deliveries. Parallel machines are considered in assembly. The schedule generated by SSM is compared with the current schedule of a consumer electronics manufacturer and the schedule generated from a methodology named 'MIP+FCFS'. Both situations of large flight capacity and

constrained flight capacity are examined. Results indicate that SSM outperforms the ‘MIP+FCFS’ method and current scheduling method of the consumer electronics manufacturer in terms of waiting cost reduction before transportation, transportation backlog reduction, and total air transportation cost decreasing.

Finally, the conclusions are drawn in the last chapter, which indicate the major contribution of this research. The directions for further work are also summarized.

Chapter 2: Literature Review

This chapter reviews works that are closely in line with the objectives of this research. Firstly, the literature concerned with production distribution integration and production transportation synchronization is discussed. Then, the research addressing transportation scheduling is discussed, which includes both air transportation and land transportation. Finally, the work on the topics of single and parallel machine scheduling problem with earliness and tardiness penalties are discussed. Also, the related research considering machine breakdown is summarized.

2.1 Production and Distribution Integration

During the past decade, the integrated analysis of production and distribution has attracted the attention of the researchers. The objective is to optimize the decision variables of different functions in production and distribution simultaneously in considering coordination of each other. Traditional research carried out the optimization without taken into account of the coordination. However, this coordinated approach has shown significant relevance to industry. Sarmiento and Nagi (1999) and Thomas and Griffin (1996) reviewed the work on production distribution integration. According to the type of decisions in the production and distribution within the supply chain, the published literature addressing integrated production and distribution scheduling can be classified into two categories: production/transportation system, production/inventory/distribution system. The major difference of these two is the

consideration of finished goods inventory between production and distribution, or finished goods inventory is maintained in the distribution facilities, such as the retailer's warehouse. It is noted that the production/transportation system is more related to this research.

2.1.1 Production/Transportation Integration Problem

As there is no finished goods inventory between production and transportation, products are transported right after the completion of production, or waiting for other products to form a batch and be transported. The problems in this category focus on the scheduling of both production and transportation. Particularly, synchronization of the two schedules is considered. The literature can be classified into two categories according to the two types of transportation modes of air and road transportation.

2.1.1.1 Synchronize the Scheduling of Production and Air Transportation

There are generally few work reported in literature addressing the synchronization of production and air transportation. To best of our knowledge, the only work in this area reported is Li et al. (2004). In this work, air transportation allocation problem is formulated by an ILP model and the production floor shop is assumed to be a single machine. A solution methodology named SBS is presented for the synchronized scheduling of both assembly and air transportation. SBS methodology is compared to an industry practice heuristic. The computational results indicate that the SBS methodology performs much better than the industry practice heuristic in reduction of intermediate waiting time and total air transportation cost.

2.1.1.2 Synchronize the Scheduling of Production and Road Transportation

Although there are a large number of papers addressing production and transportation problem separately, handful literature is available on the coordination of production scheduling with transportation scheduling (Chen, 2000).

Blumenfeld et al. (1991) examine the cost-effectiveness to synchronize production and transportation schedules on a production network which consists of one origin and many destinations. The trade-offs between production setup, freight transportation, and holding costs during production to consumption on the network are analyzed and synchronized schedules are developed. This work considers the situation in which different products have different demands, values, setup costs, and freight transportation charges. It also considers a special case, called the homogeneous case, in which these parameters are the same for all products. Costs resulting from the synchronized schedules and the independent production transportation schedules are compared to identify the advantage of the synchronized schedules.

Zuo et al. (1991) consider a production planning problem in a large scale agricultural production and distribution system. A set of production facilities and sales regions are taken into account. The task is to find ways to allocate various corn hybrids to production locations and to determine proper transportation between facilities and sales regions in order to optimize the system operations costs. This single period problem does not include any storage facilities. Heuristics are developed combining a linear programming package and a MIP package to solve the problem.

Chien (1993) consider the problem of producing and transporting a unique product directly from a production plant to a retailer that faces stochastic demands with a known probability distribution. The objective is to maximize the profit by determining a joint optimal production rate and shipping size. Firstly, the expected profit is derived as a function of the production and shipment size for a general demand distribution. Then, an iterative procedure is designed to obtain the optimal production/shipping policies by exploiting the concavity of the expected profit function on the production rate for the uniform demand distribution case. This research extends the classic Newsboy Model by combining the production and logistical decisions into an integrated framework.

Ruiz-Torres and Tyworth (1997) investigate the interaction of production scheduling and routing/transportation on a logistic network by simulation. The results indicate that low manufacturing-logistics cost and a high customer service level can both be maintained by an appropriate combination of scheduling and routing rules.

Chen (2000) addresses the problem of integrating production and transportation scheduling in a MTO environment with the aim of minimizing the total cost which consists of transportation cost, delivery tardiness penalty cost and overtime production cost. An approach is presented to determine the vehicle routing first and the production scheduling secondly. Another approach is proposed to determine production scheduling first and the vehicle routing secondly.

Garcia et al. (2004) studied a no-wait production and delivery-scheduling problem with several production plants. The orders are required to be immediate delivered by a

homogenous vehicle fleet after manufacturing. First, a special case with unlimited production capacity is considered for the problem. The problem which can be solved in polynomial time by algorithm named minimum cost flow algorithm. Then, this method has been used to construct a heuristic for the general case. To validate the efficiency of the heuristic, a graph based exact solution method is also developed to obtain optimal solution for small sized problem.

Finally, the scheduling problem with transportation consideration is studied by Lee and Chen (2001). Both the situation of transporting semi-finished job between machines and delivering finished job to customer are studied. This class of scheduling problems is computationally difficult.

2.1.2 Production/Inventory/Distribution Integration Problem

As there is finished goods inventory between production and distribution, the problem can be easily decomposed. Cohen and Lee (1988) present a hierarchical decomposition approach to decompose the whole supply chain and model each part separately. Ishii et al. (1988) study the three echelon system consists of one production facility, one wholesaler and one retailer. The objective is to minimize the dead stock and shortages by controlling the transportation ordering levels, stock levels and the production ordering level. Haq et al. (1991) also present a multi-stage model for a three echelon system, which consists of one production facility, multiple warehouses and retailers. The objective is to determine the production and distribution batch size and the inventory levels at all the locations thus to minimize total cost.

Considering the capacity planning issues within production and transportation, such as production rate and shipping size, Blumenfeld et al. (1985) and Benjamin (1989) analyze the integrated production inventory and transportation system, which including several locations on each echelon. Hall (1996) examined the decisions of economic order quantity and economic production quantity decisions in the context of integrated production and distribution system. Lee et al. (2002) proposed a hybrid approach to solve the production-distribution capacity planning problem. The approach combines analytical method and simulation method.

Considering synchronized scheduling of production and transportation, Chandra and Fisher (1994) study the system consists of a single production plant and a number of retailers. Finished goods inventory is maintained at the plant. A fleet of trucks is used to deliver products to the retailers. The objective is to minimize the transportation cost and the inventory holding cost. The proposed coordinated approach to determine production scheduling and vehicle routing in one model is shown to be better than the separated approach to solve production scheduling problem and the vehicle routing problem in two models. Fumero and Vercellis (1999) proposed an integrated optimization model for production and distribution planning with the aim of optimally coordinating important and interrelated logistics decisions such as capacity management, inventory allocation, and vehicle routing. Özdamar and Yazgac (1999) model a production-distribution system involving production and transportation decisions in a central factory and several warehouses. The overall system costs are optimized considering factory and warehouse inventory costs and transportation costs. Constraints include production capacity, inventory balance and fleet size integrity. A hierarchical

production-distribution planning approach is proposed to solve the problem optimally. Dhaenens-Filipo and Finke (2001) consider a multi-facility, multi-product and multi-period production-distribution system. The objective is to minimize total cost which consists of production cost, production changeover cost between different products, holding cost and transportation cost, by assign products to the unique production lines.

Hall and Potts (2003) considered a variety of scheduling, batching, and delivery problems that arise in an arborescent supply chain where a supplier makes deliveries to several manufacturers, who also make deliveries to customers. Models and algorithms are presented to corporate the scheduling and batching decisions of supplier and manufacturer to minimize the total system cost. Lee et al. (2003) consider the single-product production and transportation problem with dynamic demand. The model allows heterogeneous vehicles to immediately transport the finished product in the same period it is produced. The objective of this study is to simultaneously determine the optimal production and transportation policy that minimizes the total system cost (production cost, inventory holding cost, and freight cost) to satisfy the dynamic demands over the finite time horizon. Jolayemi and Olorunniwo (2004) developed a deterministic model for planning production and transportation quantities in multi-plant and multi-warehouse environment with extensible capacities. The decision variables are the quantities of each product to be: (1) produced at each plant, (2) transported from each plant to each warehouse, (3) subcontracted at each warehouse, and (4) kept in inventory at each warehouse.

2.2 Outbound Logistics

Quick delivery is desired by the consume electronics products for the short life cycle. It is observed in industry that air transportation is commonly used for international transportation and local land transportation mostly depends on the trucks. Therefore, this section reviews the published literature addressing air transportation and land transportation. Within the category of land transportation problems, the vehicle routing problem (VRP) is extensively studied. The work in line with the objectives of the problem studied in this thesis is reviewed.

2.2.1 Air transportation Scheduling Problem

With the growing importance of international air freight traffic, air cargo has taken a prominent place among the services from which firms can derive competitive advantages in global markets. Air cargo has become the principal model of international transport for a wide variety of goods, especially knowledge-intensive goods (Leinbach and Bowen, 2004). In addition, Bowen (2004) provided the geography of freighter aircraft operations in the Pacific Basin. Leinbach and Bowen (2004) also identified the significance of air cargo service to the electronics industry in the Southeast Asia. The great contribution of air freight traffic to economics has motivated significant research interest in air transportation. The major current research topics cover network design, cost and capacity issues of airlines, investment on airports, airline alliance, safety and security, etc. (Motevalli and Stough, 2004; Jorge and Rus, 2004, Balfour, 2004; Tsai and Kuo, 2004; Oum and Lee 2002; Crainic, 2000; Barnhart and Schneur, 1996). The operational issues studied in the air cargo freight in literature includes the various stages of aircraft and crew schedule planning; revenue management, including overbooking

and leg-based and network-based seat inventory management; and the planning and operations of aviation infrastructure (Barnhart et al., 2003).

These operations issues are observed from the airline's or airport's viewpoint. There is a relative lack of research from the customer's viewpoint to negotiate with airlines to enhance the efficient utilization of the service provided by the airlines. To best of our knowledge, only the following one reference is found similar to the air transportation allocation problem in this thesis. Tyan et al. (2003) studied the freight consolidation problem of grouping different shipments from suppliers into a larger shipment at the consolidation point. The objective is to maximize the aircraft utilization while simultaneously maintaining the service commitments. Note that the service commitments accounts for the quantity of products to be shipped out each day. Three consolidation policies were developed to minimize the total cost under the constraints of capacity and service requirements. A mathematical programming model is developed to assist the evaluation of consolidation policies. The optimal solution specified in what quantities the shipments should be allocated to alternative flights on each day so that the service requirements are satisfied at a minimal cost. The study indicates that a collaborative consolidation policy can benefit both the carrier and the shipper at the same time. In this work, final delivery earliness and tardiness is not taken into account, which is the major difference from the air transportation problem in this thesis.

2.2.2 Vehicle Routing Problems

The typical VRP has been studied for decades. For a classical VRP problem, a set of nodes with known capacity and location is given. A fleet of trucks from a single depot

with known capacity is used to serve the demand nodes. The objective is to find a set of delivery routes that minimizes the total distance traveled. This problem is deterministic without time constraints. Surveys of the VRP problem are available in Fisher (1995), Fawzan and Al-Sultan (1996), Gendreau et al. (1997), and Chen (2000).

Multiple extension of the VRP problem is also studied in literature. VRP problem with time windows receives much research interest for a long time (Ho and Haugland, 2004; Berger and Barkaoui, 2004; Chaovalitwongse et al., 2003). Surveys can be found in Toth and Vigo (2002) (chapter 7) and Golden and Assad (1998). Capacitated VRP problem is also well studied in the past two decades (Achuthan and Caccetta, 1990; Laporte, 1992). The VRP with backhauls is the extension of the capacitated VRP problem. Toth and Vigo (2002) (chapter 8) reviewed the published algorithms for the backhauls VRP problem. In addition, the literature addressing the VRP problem with pickup and delivery is also reviewed by Toth and Vigo (2002) (chapter 9). The literature on this problem is very scant compared to that of the classical VRP (Nagy and Salhi, 2004).

2.3 Single and Parallel Machine Scheduling with Earliness and Tardiness Penalties

Studies of scheduling problems involving minimization of both earliness and tardiness (E/T) costs have emerged as a major thrust of research in the scheduling field in the past two decades. These problems are largely motivated by the adoption of the just-in-time concept in today's manufacturing industry, which aims to finish jobs exactly at their due dates, not earlier and not later. At the first decades of study on the machine

scheduling problem with E/T, virtually the entire body of literature deals with static scheduling and a majority of the studies focus on single machine models (Baker and Scudder 1990). During recent years, the research on the parallel machine and multi-stage machines has become increasingly popular. In addition, the E/T problems considering setup time, ready time for release, due windows and machine breakdown have led to growing interest.

2.3.1 Single Machine Scheduling Problem with Earliness and Tardiness

In order to describe a generic model of single machine scheduling problem with E/T, let N be the number of jobs. Each job is associated with a processing time p_j , a due date d_j , and a completion time C_j . let α_j and β_j be the weight of the unit time earliness and tardiness penalty of job j respectively. The objective is to minimize:

$$\sum_j \{\alpha_j \text{Max}[0, (d_j - C_j)] + \beta_j \text{Max}[0, (C_j - d_j)]\}$$

Many variations of the single machine E/T scheduling problem are studied in the past two decades. Generally, the work can be classified into two categories according to the due date considered: common due date and distinct due dates. A common due date model can represent a situation where several items constitute a single customer's order, or it might reflect an assembly environment in which the components should all be ready at the same time to avoid staging delays. Distinct due dates represents a more general production environment for various customer's orders. Literature in each category can be further classified according to the weights of E/T. An excellent review

is given by Baker and Scudder (1990). Instead of repeating, the research on the single machine E/T scheduling problem excluded in this review paper is reviewed.

2.3.1.1 Single Machine E/T Scheduling Problem with Common Due Date

Two kinds of due dates, namely, restrictive and unrestrictive due dates, are distinguished in literature. A common due date is unrestrictive if it is large enough and has no influence on the optimal sequence (Kanet, 1981). Note that a given due date which is greater than or equal to the sum of processing times of all jobs is always unrestrictive. On the other hand, if a given common due date may influence the optimal sequence of the jobs, it is called restrictive.

2.3.1.1.1 Single Machine E/T Scheduling Problem with Unrestricted Common Due Date

A generalization of the single machine E/T scheduling problem allows symmetric and job dependent E/T weights, i.e., $\alpha_j = \beta_j$. Hall and Posner (1991) prove that this problem is NP-complete. They present four special cases that optimal solution can be found in polynomial time: weights and processing time are equal, jobs have unit processing times, not job is late and the first job is large. Kovalyov and Kubiak (1999) present a fully polynomial approximation scheme for this problem. The main feature is recursively computation of lower and upper bounds on the value of partial optimal solutions. Therefore, the scheme does not require any prior knowledge of lower and upper bounds on the value of a complete optimal solution.

As symmetric E/T weights is a special case of the asymmetric E/T weights, therefore, the single machine E/T scheduling problem considering asymmetric and job dependent

E/T weights is also NP-complete. James (1997) develop Tabu search algorithms for this problem. Different forms of the Tabu search are tested, including one based on a job sequence solution space and another based on an early/tardy solution space. Results show that a search which uses an early/tardy solution space with a neighborhood scheme which eliminates infeasible areas of the solution space is the most efficient and effective solution method.

2.3.1.1.2 Single Machine E/T Scheduling Problem with Restricted Common Due Date

More recent work has appeared addressing restricted common due date. Considering the case of symmetric and job independent weights, i.e., $\alpha_j = \alpha$, $\beta_j = \beta$, and $\alpha = \beta$, the objective is to minimize the sum of the absolute deviation of each job's completion time from the common due date. Hoogeveen and van de Velde (1991) and Hall et al. (1991) independently proved that this problem is NP-complete. Dynamic programming algorithms are presented. Additionally, special cases of this problem are described, which is polynomial solvable. It is noted that other variations of this problem, which considering asymmetric or job dependent E/T weights, are clearly NP-complete.

A more general representation allows symmetric and job dependent weights, that is $\alpha_j = \beta_j$. Hoogeveen and van de Velde (1991) have found that an optimum schedule possesses a W-shaped structure and, consequently, applied the optimality property to devise a solution algorithm with pseudopolynomial time complexity. De et al. (1991) presented a dynamic programming algorithm for this problem in order to generate an exact solution. Special cases amenable to easier solutions are also presented including identical job processing time and identical weights for each job. Alidaee and Dragan

(1997) considered a special case of the problem where weights of jobs were proportional to processing times and the common due date was any given number. They showed that Longest Processing Time (LPT) sequence rule leads to optimal solution and an efficient algorithm for generating an optimal schedule is presented.

Considering asymmetric and job dependent weights, De et al. (1993) present dynamic programming algorithms for this problem. Lee and Kim (1995) propose parallel genetic algorithm. Hoogeveen et al. (1994) develop Branch and Bound algorithm based on Lagrangian lower and upper bounds. Szwarc (1996) extended the algorithm presented by Hoogeveen and van de Velde (1991). The Tabu search method presented by James (1997) can also be applied on this problem. Biskup and Feldmann (2001) present numerous benchmark problems together with some upper bounds on the optimal objective function value. Feldmann and Biskup (2003) study this problem by applying meta-heuristics, namely evolutionary strategies, simulated annealing and threshold accepting.

Considering asymmetric and job independent weights, Ventruea and Wen (1995) improved the dynamic programming algorithm presented by Hall et al. (1991). Mondal and Sen (2001) have suggested an algorithm using graph search space to optimally solve large instances of the problem. It has been found to run faster than dynamic programming and Branch and Bound formulations and can solve much larger instances of the problem in reasonable time. New upper and lower bounds have also been proposed.

2.3.1.2 Single Machine E/T Scheduling Problem with Distinct Due Dates

A large number of papers have appeared during the past decade addressing the single machine E/T scheduling problem with distinct due dates. Generally, the research in this area belongs to two categories: with inserted idle time and without inserted idle time in the schedule.

The problem without inserted idle time was studied by Abdul-Razaq and Potts (1988), Chand and Schneeberger (1988), Ow and Morton (1989), Li (1997), and Liaw (1999), etc. For this problem considering job independent E/T weights, Abdul-Razaq and Potts (1988) developed a Branch and Bound algorithm whose lower bound is obtained by performing recursions on a relaxed state space. Considering job dependent weights, Chand and Schneeberger (1988) study the problem where the objective is to minimize total weighted earliness without tardy jobs. The problem is proved to be NP-hard and dynamic algorithm is presented. Considering both earliness and tardiness, Ow and Morton (1989) developed several heuristics and altered beam search algorithm, which, however, cannot guarantee the optimality of the solutions. For the problem considering asymmetric and job dependent weights, Li (1997) and Liaw (1999) proposed Lagrangian relaxation based Branch and Bound algorithms that guarantee the optimality of the solution but require a great deal of computational effort.

The general cases of the single machine E/T scheduling problem with distinct due date are those that allow inserted idle times. As Baker and Scudder (1990) pointed out, the assumption of no inserted idle times is inconsistent with the E/T criterion because earliness is an irregular performance measure. It is noted that irregular performance

measure is decreasing in job completion times. Because of the hardness of this problem, Branch and Bound algorithm and search based procedures become two typical approaches for these problems. Typically, methods found in the literature solve this problem in two phases. In the first phase, a sequence is generated by some heuristic method. Then, in the second phase, deliberate idle times are inserted in the given sequence to form a schedule (Sridharan and Zhou1996).

Considering symmetric and job independent E/T weights, the objective is to minimize the sum of absolute deviations of the completion time from the due date of each job. Chang (1999) presents a bounding scheme for the calculation of different lower bounds based on the overlap elimination procedure on a just-in-time schedule. Properties and theorems of the overlap elimination procedure are also provided. Ventura and Radhakrishnan (2003) derived properties of an optimal schedule for this problem. A Lagrangian relaxation procedure that utilizes the subgradient algorithm has been implemented to attain tight lower bounds.

Taken into account of asymmetric and job independent E/T weights, Chen and Lin (2002) utilize dominance rules to develop a relationship matrix, which allows a Branch and Bound algorithm to eliminate a high percentage of infeasible solutions. A procedure combining this matrix and a branching strategy is then presented to solve the problem.

Addressing the general problem with asymmetric and job dependent E/T weights, Yano and Kim (1991) and Kim and Yano (1994) consider the problem where weights are

proportional to processing times. Using a simple sorting process, a sequence is generated firstly. Then a dynamic programming procedure is applied to insert idle time in to the job sequence optimally. Davis and Kanet (1993) proposed a TIMETABLE procedure for adjusting idle time by shifting job starting times in a given sequence to minimize the sum of E/T costs, and used this to establish a lower bound for Branch and Bound methods. Szwarc and Mukhopadhyay (1995), and Lee and Choi (1995) independently presented optimal timing algorithms for this problem. These algorithms focus on the weight comparison and were used in Branch and Bound algorithms and local search algorithms. Lee and Choi (1995) also adopted a genetic algorithm for this problem. James and Buchanan (1997) present a Tabu search algorithm and it is improved by James and Buchanan (1998). Almeida and Centeno (1998) present a composite heuristics that combines steepest decent, simulated annealing and Tabu search that does not rely on particular properties of the optimal solution. Verma and Dessouky (1998) address the problem with identical processing time and point out that if the penalties meet a certain criterion called Dominance Condition, the problem is polynomial solvable. Symmetric E/T penalty weights problem satisfies this criterion. Sourd and Kedad-Sidhoum (2003) presented a Branch and Bound algorithm to the problem with distinct deadlines and distinct earliness and tardiness penalties. The lower bound is based on the decomposition of the tasks into unary execution time operations so that it can be computed in pseudo-polynomial time.

Pathumnakul and Egbelu (2005) address the single machine scheduling problem with the objective of minimizing the weighted earliness penalty without tardy job. Each job has a distinct due date and job dependent earliness weight. The work identifies several

local optimality conditions for sequencing of adjacent jobs. A heuristic is developed based on these local optimality conditions. The heuristic starts by scheduling each job such that its completion time coincides with its due date. Probably, there are machine conflicts between jobs, which can be eliminated by applying the local optimality conditions. The approach to eliminate machine conflict starts from the latest conflict. Then, the approach moves backward until all the conflicts are eliminated. Finally, the heuristic searches for improved solution by inserting the imbedded idle times in the job sequence.

2.3.1.3 Single Machine E/T Scheduling Problem considering Ready Time for Release

Earliness tardiness problem with distinct arrival time or available time of each job is complicated even for single machine case. There are only a few studies dealing with this problem. Most of published research concerns on the distinct due dates problem. For symmetric E/T weights problem, Nandkeolyar et al. (1993) propose a modular solution approach which consists of a main theme and an optional front end and/or balancing routine. Considering asymmetric and job dependent E/T weights, Sridharan and Zhou (1996) propose a decision theory based single-pass heuristic for constructing the schedule. Mazzini and Armentano (2001) develop a constructive heuristic, which determines the sequence of jobs and simultaneously inserts idle times. For the problem subject to unrestricted common due date, asymmetric and job independent E/T weights, Ventura et al. (2002) show that the problem is NP-hard in the ordinary sense. An exact dynamic programming algorithm for small and medium size problems is developed. A heuristic algorithm for large-scale problems is also proposed.

2.3.1.4 Single Machine E/T Scheduling Problem considering Setup Time

In most of the E/T literature, it has been assumed that no setup time is required or the setup is included in the processing time. In many realistic situations, however, setup times are needed and are sequence-dependent. In general, scheduling problems with sequence-dependent setup times are similar to the Traveling Salesman Problem (TSP), that it is NP-hard (French, 1982).

Considering distinct due dates and job-dependent penalties, Coleman (1992) presented a 0/1 MIP model, which can find optimal solutions for problems with up to eight jobs. Chen (1997) addressed the single machine E/T scheduling problem with batch sequence-dependent setup times. He showed that the problem with unequal penalties is NP-hard even when there are only two batches of jobs and two due dates that are unrestrictively large. Webster et al. (1998) propose a genetic algorithm for scheduling jobs about an unrestricted common due date with family dependent setup times. A family setup time is required when a job follows a job belonging to some other family. The E/T penalties weights are asymmetric and job dependent. The problem is shown to be NP-complete. Rabadi et al. (2004) addressed the problem with unrestricted common due-date and job independent symmetric E/T weights. They show that this problem is NP-hard. A Branch and Bound algorithm is presented and can solve the problem up to 25 jobs in reasonable time.

2.3.1.5 Single Machine E/T Scheduling Problem considering Due Windows

In many practical situations, jobs have distinct due windows rather than due dates because of uncertainty and tolerance. However, relatively few research consider due

windows, among which most of the research focuses only on common due window $[d - tr, d + tr]$, rather than distinct due windows. Note that d denotes a due date and tr denotes the tolerance.

Cheng (1988) studied a common due window problem where the length of the due window is strictly less than the smallest job's processing time. Any job that completes outside the window is assessed a penalty equal to the absolute deviation from the center of the window. Some dominant properties are proposed for determining optimal solutions. Kramer and Lee (1993) consider a generalization of this model where each job has the same earliness weight and the same tardiness weight, but the two weights are allowed to asymmetric. This problem is known to be easily solved when the due window is an instant in time (Kubiak et al. 1990). Kramer and Lee (1993) generalize properties to the case where the due window is an interval of time and propose a polynomial time algorithm.

Weng and Ventura (1994) provided a simple equation to calculate a critical d^* . If $d > d^*$, the problem is called unrestricted, and restricted if $d < d^*$. Weng and Ventura (1994) identify a polynomial time algorithm for special case of the unrestricted problem with unit E/T weights. For restricted common due window, Ventura and Weng (1996) consider to minimize mean absolute deviations of completion times from the common due window. The problem is shown to be NP-hard and a Lagrangian relaxation procedure is proposed. Weng and Ventura (1996) address the problem to minimize the mean absolute deviation of job completion times about a given common due date with different sizes of tolerance. Some optimality conditions are described and the problem

is shown to be NP-complete. A pseudo-polynomial dynamic programming algorithm is proposed to optimally solve the problem. For the problem with unrestricted start time of the common due window and arbitrary E/T penalty weights, Azizoglu and Webster (1997) present properties that characterize the structure of an optimal schedule, together with a lower bound and a two-step Branch and Bound algorithm.

Considering distinct due windows, Koulamas (1996) studied the problem with equal E/T penalty weights. He defined the earliness as the difference between the beginning of the due window and the starting time of a job, which is different from normal definition of earliness. An optimal timing algorithm and several sequencing heuristics are developed. While considering asymmetric and job dependent E/T weights, Wan and Yen (2002) present an optimal timing algorithm to determine completion times for each job in a given job sequence. A Tabu search procedure is employed to generate job sequences and final schedules.

2.3.1.6 Single Machine E/T Scheduling Problem considering Machine Breakdown

Stochastic E/T scheduling problems have only received limited attention recently, although many practical problems involve essentially randomness, such as random processing times, random deadlines, and stochastic machine breakdowns. In practice, most of the uncertainty centers around the availability of the machines (Federgruen and Mosheiov, 1997).

Mannur and Addagatla (1993) consider fixed idle times over the horizon, such as shift changes and periods of machine maintenance, and propose two heuristics based on the ordering of the values of the job release time R_j and job due date d_j .

Mittenthal and Raghavachari (1993) address the problem considering random breakdowns, subject to unrestricted common due date, asymmetric and job independent E/T weights. A simple recourse model is considered when the penalty function is the squared deviation of job completion times from the common due date, and a deterministic equivalent objective function is developed. Characterizations of optimal schedules for this quadratic objective function are established. Cai and Tu (1996) also examine the similar problem. The processing time of each job is a random variable, which follows an arbitrary distribution with a known mean and a known variance. The objective is to minimize the expected sum of squared deviations of job completion times from the due date. A general form of the deterministic equivalent of the stochastic scheduling problem is obtained when the counting process related to the machine uptime distribution is a generalized Poisson process. A sufficient condition is derived under which optimal sequences are V-shaped with respect to mean processing times. Based on the optimality properties, algorithms with pseudo polynomial time complexity are proposed to solve both versions of the problem.

Federgruen and Mosheiov (1997) consider the single machine scheduling problem with arbitrary machine breakdown and repair processes. The objective is to minimize the sum of the τ th power of the deviations of the jobs' completion times from the due-date, for some $\tau \geq 1$. The problem is shown equivalent to a deterministic one without

breakdowns and repairs and with an equivalent cost function of a job's completion time. A V-shaped schedule without idle time is shown to be optimal, if this equivalent cost function is quasi-convex. The general conditions are derived for the E/T cost structure and repair and breakdown processes under which the equivalent cost function is quasi-convex.

Cai and Zhou (2000) address the single machine scheduling problem with machine down for distinct due date and job dependent E/T penalty weights. Both symmetric and asymmetric weights are examined. The processing times of the jobs are exponentially distributed random variables and the machine is subject to stochastic breakdowns governed by a Poisson process. The problem is shown to be NP-complete. Optimality properties are identified and dynamic programming algorithms are proposed.

2.3.2 Parallel Machine Scheduling Problem with Earliness and Tardiness

Recently, research on parallel machine scheduling problem with E/T penalties has increased. However, compared to the published work on the single machine E/T problem, there are only a handful of results published for the parallel machine problem. The published literature is classified into two categories according to identical machines or unrelated machines considered. Also, there are variations of the parallel machine E/T scheduling problem within each category, considering setup time, symmetric or asymmetric E/T penalty weights, common due date or distinct due dates, etc. The detailed review is provided in the following sections.

2.3.2.1 Identical Machines

Clearly, the identical machine refers to the identical value of machine related parameters, such as the processing time of a job on each machine. As the setup time is commonly considered, research addressing identical machine scheduling with E/T penalties are further classified into two categories subject to setup time be taken into account or not.

2.3.2.1.1 Considering Setup Time

In manufacturing system, it is common that a setup time is required when the machine change to process a different product. Generally, there are three types of setup times considered in literature:

- Sequence dependent setup time: setup time depends on the preceding and succeeding relationship of two adjacent jobs (Zhu and Heady, 2000).
- Family setup time: setup time incurs when a machine changes from processing a job in one job family to a job in another job family (Yi and Wang, 2003; Webster, 1997).
- Job machine dependent setup time: the setup time of a job varies according to the machine it allocated to (Balakrishnan et al., 1999).

It is noted that besides setup time issues, difference of the literature can also be identified based on the common due date or distinct due dates.

For the common due date problem, again, there exists the difference between restricted and unrestricted common due date. The unrestricted due date is defined by Hall (1986)

that it is sufficiently large to have no impact on the optimal assignment of jobs to machines and their sequences on these machines. Webster (1997) also provides distinction between unrestricted and restricted common due date problems. If the optimal cost cannot decrease with increases in the common due date, then the problem is an unrestricted common due date problem; otherwise it is a restricted common due date problem. Webster (1997) considers the parallel machine scheduling problem with family setup time, symmetric and job dependent E/T weights. The work summarizes known complexity results and introduces new complexity results that the E/T scheduling problem is unary NP-hard when the number of machines and families are arbitrary. For the same type of problem, Yi and Wang (2003) present an algorithm named soft computing, which is a fuzzy logic embedded genetic algorithm.

For the distinct due dates problem, Balakrishnan et al. (1999) consider job arrival time in a dynamic albeit deterministic manner, sequence dependent and job machine dependent setup time, and arbitrary E/T penalty weights. They present a mixed integer formulation for the problem to solve small sized problems. Inserted idle time is taken into account to obtain optimal completion time of each job. A primal-dual-based procedure is discussed for solving large sized problem. Radhakrishnan and Ventura (2000) consider sequence dependent setup time to minimize sum of the absolute deviations of job completion times from their corresponding due dates. The problem is formulated as a 0-1 MIP model and a simulated annealing algorithm is presented to solve this problem.

2.3.2.1.2 No Setup Time

It is assumed that setup time is included in the processing time of the job. Thus, setup time issues are avoided in the problem formulation. Again, there are two types of problems considering common due date and distinct due dates.

For the common due date problem, Sundaraagavan and Ahmed (1984) presented a polynomial algorithm to minimize the sum of absolute deviations of job completion time from the unrestricted common due date. Laguna and Velarde (1991) study the problem to minimize the weighted earliness problem without tardiness. This problem is shown to be NP-complete. A search heuristic is presented that combining elements of the greedy randomized adaptive search procedure and Tabu search. A Branch and Bound post processor is also used to optimize the sequence of the jobs assigned to each machine individually. Federgruen and Mosheiov (1996) presented a lower bound and a simple heuristic for the unrestricted due date problem to minimize sum of absolute deviation of the completion time from the common due date. Both of them have polynomial computational time. The heuristic is also generalized to setting with restrictive due date and general asymmetric earliness and tardiness cost function. Chen and Powell (1999) address the problem with unrestricted due date and arbitrary E/T weights. An algorithm named column generation based exact decomposition algorithm is presented. Sun and Wang (2003) consider a special case that the weight of each job is proportional to processing time of the job. Furthermore, the E/T penalty weights are symmetric and job dependent. The problem is proved to be NP-complete by reducing it to a partition problem. A dynamic programming algorithm is presented.

For the distinct due dates problem, Sivrikaya-Serifoglu and Ulusoy (1999) consider the problem constrained by job ready time, asymmetric and job independent E/T weights. Two genetic algorithms are employed to attack this problem; one with a crossover operator and the other with no crossover operator. They conclude that both of the two algorithms are efficient for solving this problem. Ventura and Kim (2003) consider the problem with job ready time, identical processing time and additional resources beside the machine required by each job with the objective of minimizing sum of deviation of completion time from the corresponding due date. The problem is formulated by ILP model and the Lagrangian relaxation approach is used to obtain tight lower bounds.

2.3.2.2 Unrelated Machines

A review of the literature indicates that only a few of the published E/T articles address the problem of scheduling jobs on multiple, non-identical machines to minimize the total penalty costs for earliness and tardiness (Zhu and Heady, 2000).

2.3.2.2.1 Considering Setup Time

To best of our knowledge, there is no research has been published in this category addressing the common due date problem. For the distinct due dates problem, Heady and Zhu (1998) consider sequence dependent setup time and arbitrary E/T penalty weights. This problem is shown to be NP-hard. A heuristic is developed to provide approximate solution for large sized problems. Zhu and Heady (2000) give a MIP formulation for the same type of problem. The model can provide optimal solution to problems involving about nine jobs and three machines.

2.3.2.2.2 No Setup Time

When setup time is assumed to be included in the processing time, the common due date problem has attracted attention almost twenty year ago. However, there is only handful published work addressing this problem. Emmons (1987) extended the algorithm presented by Hall (1986) to address the case with asymmetric earliness and tardiness and settings with unrelated machines. Kubiak et al. (1990) address the problem with unrestricted common due date to minimize mean absolute deviation of job completion time from the common due date. The problem is reduced into a transportation problem, thus is shown to be solvable in polynomial computational time. Bank and Werner (2001) consider the problem with distinct ready times and arbitrary E/T penalty weights. Some structural properties are derived, which is useful in connection with the search for an approximate solution. Various constructive and iterative heuristic algorithms are presented, which are compared on problems with up to 500 jobs and 20 machines. They concluded that there is no one heuristic is superior to the remaining ones.

For distinct due dates problem, a special cases is addressed by Della Croce and Trubian (2002). In this problem, the assignment of jobs to machines and the sequence on each machine is known. Constrained by the preceding and succeeding relationship among jobs, idle time is to be inserted between jobs to minimize the sum of absolute deviation of completion time to the corresponding due date. An exact procedure is proposed based on repeated solutions of maximum cardinality matching problems on a bipartite graph with polynomial computing time.

2.3.3 Techniques for Solving the Earliness Tardiness Scheduling Problems

2.3.3.1 Modeling

The first work addressing single machine scheduling problem with E/T penalties is by Sindney (1977), who formulate the problem using a Linear Programming (LP) model. Baker and Scudder (1990) present a generic LP model for the single machine E/T scheduling problems. Also, they list seven variations considering different objectives for the single machine E/T scheduling problem. Each problem is formulated by an LP model. MIP model is also used to formulate the single machine E/T scheduling problem with sequence dependent setup times (Coleman, 1992). Most of the parallel machine E/T scheduling problems are formulated as ILP problems (Kubiak, 1990; Chen and Powell, 1999; Ventura and Kim, 2003) or MIP problems (Radhakrishnan and Ventura, 2000; Zhu and Heady, 2000). According to the review in section 2.3.1 and 2.3.2, most of the single and parallel machine E/T scheduling problems are NP-complete. Exact and approximate methods are proposed in the published works.

2.3.3.2 Exact methods

Algorithms are proposed in literature in order to obtain the optimal solution for small sized problem in acceptable computational time. Three popular methods are applied in the E/T scheduling problems: Branch and Bound algorithm (Hoogeveen et al., 1994; Mondal and Sen, 2001; Chen and Lin, 2002), Lagrangian relaxation method to obtain lower bound (Ventura and Weng, 1996; Liaw, 1999; Ventura and Kim, 2003), and dynamic programming algorithm (Hoogeveen and van de Velde, 1991; Hall et al., 1991; Kim and Yano, 1994; Ventura et al., 2002). Branch and Bound algorithm finds the optimal solution by efficiently enumerating the points in a sub-problem's feasible

region. Lagrangian relaxation is a general solution strategy for solving ILP problems. The procedure is used to decompose a hard problem into several easy sub-problems by exploiting its special structure. By solving the sub-problems, the lower bound on the optimal solution of the original problem is obtained and can be utilized in a Branch and Bound algorithm (Fisher, 1981). In most applications, dynamic programming algorithm obtains solutions by working backward from the end of a problem toward the beginning, thus breaking up a large, unwieldy problem into a series of smaller, more tractable problems (Winston, 1994).

2.3.3.3 Approximate algorithms

For large sized E/T scheduling problem, approximate algorithms are employed. Local search methods are widely applied in literature, which are based on genetic algorithm (Lee and Kim, 1995; Webster et al., 1998; Yi and Wang, 2003), Tabu search (James, 1997; James and Buchanan, 1997; Wan and Yen, 2002) and simulated annealing (Radhakrishnan and Ventura, 2000).

Genetic algorithms were developed by Holland (1975) as artificial adaptive systems to simulate natural evolution. Genetic algorithms use a population of candidate solutions to conduct a robust search of the search space. The candidate solutions are often called chromosomes. Initially, a set of chromosome is generated randomly. Each chromosome is then evaluated by assessing its objective value. The algorithm then enters a loop. Each iteration of the loop is called a generation. A generation consists of two steps: (1) *selection* and (2) *recombination*. The first step is done probabilistically and the population size does not change. Good chromosomes get more copies in the new

population. The recombination step applies one or more recombination operators to the new population. The two most common operators are *crossover* and *mutation*. After the recombination operators are applied, the population for the next generation is obtained. The loop continues until a terminating condition is met. Genetic algorithms are widely applied for solving the NP-complete problem for their effectiveness and efficiency performances in complex searching spaces.

Tabu search was developed by Glover (1989, 1990). The ideas of Tabu search can be briefly illustrated as follows. At first, the solution space and the neighborhood for the search are identified. The neighborhood defines all possible moves from the current solution. Tabu search iteratively moves from the current solution to its best neighborhood. The move then becomes ‘tabu’ and cannot be reversed until either a predefined number of iterations have passed or the resulting solution obtained by reversing this move meets some ‘aspiration criteria’ which is common that the solution obtained by the move is better than the best solution found.

Introduced by Kirkpatrick et al. (1983) and Cerny (1985), simulated annealing is a local search algorithm using probabilistically accepting worse solutions to escape local optimal. For a simple local search algorithm, an initial solution is chosen randomly. For the first iteration, the initial solution is taken as current solution. From current solution, a neighborhood solution is generated based on some specific mechanism. If the neighborhood solution is better than current solution, it is accepted and replaces current solution. Otherwise, the current solution is kept unchanged. This procedure is repeated until no improvement can be obtained in the neighborhood of current solution. The

major drawback of this algorithm is that it is easily trapped in a local optimal solution. In order to escape the trap, simulated annealing accepts some inferior solution in the neighborhood by a probability. The probability is set by $\exp(\Delta Z/T)$, where ΔZ is the difference between the neighborhood solution and the current solution. T is the control parameter and decreased by a cooling rate r , which corresponds to the temperature in the analogy with the physical annealing process. The value of $\exp(\Delta Z/T)$ is compared with a random number which is drawn from uniform distribution on $(0, 1)$. If the value is greater than the random number, the inferior neighbor is accepted. Otherwise, the inferior neighbor is rejected. The initial T is set large enough that almost all no-improve neighbors are accepted at early stage of the searching. This ensures that the searching is not trapped in a local optimum at the early stage. As T is near to a minimum acceptable value t , almost all no-improve neighbors are rejected. Furthermore, SA searches predefined number of iterations at each value of T in order to find better solution. Throughout the searching procedure, the parameter BS is used to record the best solution ever found. SA has received considerable attention in the recent past and has been widely used to solve difficult combinational optimization problems.

2.4 Summary

In this chapter, research addressing the production distribution integrating problems is reviewed. As the problem studied in this research is decomposed into two sub-problems and be formulated as the machine scheduling problem with earliness tardiness penalties and the air transportation problem respectively, research related to these two problems is also summarized. In addition, the commonly applied techniques for solving the machine earliness tardiness scheduling problems are reviewed. From the above review

of the related work, the following important observations can be obtained and these would highlight the significance of the research problem addressed in this thesis:

- The research on synchronizing the production and distribution is mainly carried out in the production/inventory/distribution systems (Sarmiento and Nagi, 1999).
- For the distribution of products, majority of the published work focuses on land transportation and the issue related to outbound logistics was formulated as VRP (Chen 2000). There is a relative lack of literature addressing air transportation operations from the customer point of view (Li et al. 2004).
- Although the single machine E/T scheduling problem is well studied, there is a handful of results published for the parallel machine E/T scheduling problems (Sun and Wang, 2003).

Specifically, focusing on each sub-problem studied in this research respectively, the following figures highlight the scope that each sub-problem falls into. Also, the contribution of this research can be identified. The assembly scheduling problem falls into the scope of machine E/T scheduling problem. Three problems are studied in this research: single machine assembly scheduling, parallel machine assembly scheduling and single machine assembly scheduling with stochastic machine breakdowns. The first assembly problem considers distinct due dates, symmetric and job independent weights. Chang (1999) stated that this problem is NP-complete. However, this research investigates a special case of the problem: only earliness incurred, more than one job sharing due dates with one another, which is polynomially solvable. The second assembly problem considers unrelated parallel machine, distinct due dates, job

dependent earliness penalties. To the best of our knowledge, there is no work published in this area. The last assembly scheduling problem considers distinct due dates, asymmetric and job dependent E/T penalty weights. This type of problem is well known to be NP-complete. This problem has its unique complex non-linear objective function subject to the complex structure of CESC, which is illustrated in Figure 1.4. Furthermore, stochastic machine breakdowns are considered in this problem. Figure 2.1 details the differences between the assembly scheduling problems addressed in this research and the published literature in E/T machine scheduling.

Figure 2.1 Comparison of published literature and assembly scheduling problems in

CESC

Figure 2.2 provides an overall picture of operations research applied in transportation industry, especially in air transportation, and highlights the contribution of this research as the first effort in investigating flight allocation problems.

Figure 2.2 Comparison of published literature and air transportation allocation problem in CESC

Chapter 3: The Air Transportation Problems

In chapter 1, a two stage methodology is presented to solve the synchronization problem within Consumer Electronics Supply Chain (CESC), which decomposes the overall synchronization problem into air transportation problem and assembly scheduling problem. In this chapter, the air transportation problem is studied, which is the first approach of the synchronized scheduling methodology. The problem is investigated in various situations of single/multiple destinations of transportation and split/non-split deliveries. Generally, the formulation and the complexity of the air transportation problem differ significantly according to different situations in CESC.

3.1 Introduction and Assumptions

The important factors which are taken into account in the air transportation problem formulation are: (a) the number of available flights to different destinations in the planning period, (b) the departure and arrival times of the flights, (c) the designated flight capacity and the corresponding transportation cost.

Generally, there are two categories of air transportation problems according to the assumptions on single destination or multiple destinations. There is also an important assumption to be considered, which refers to split or non-split deliveries. Split deliveries means order can be split on conditional reason and allocated into more than one flight and delivered separately. Some sub-orders may be delivered earlier than the

due date, while others may be delivered later. Thus, both earliness and tardiness penalties can be imposed to the original order. In contrast, in the situation of non-split deliveries, one order can only be allocated to one flight. Therefore, the air transportation problem can also be classified into two categories according to the split/non-split deliveries assumptions.

Therefore, combining the assumptions on transportation destination and order split in delivery, there are four types of air transportation problems to be studied within the CESC synchronization problem: single destination with split deliveries (**Type 1**), single destination with non-split deliveries (**Type 2**), multiple destination with split deliveries (**Type 3**), multiple destination with non-split deliveries (**Type 4**). In this chapter, the four problems are formulated and the computational complexities are investigated.

The general assumptions, based on which the four air transportation problems are formulated, are presented as follows:

- Decisions of air transportation allocation and assembly scheduling are applied for the orders accepted in the previous planning period.
- Order fulfillment is considered to be achieved when the order reach the destination airport on time.
- All the packed products have same or similar dimensions.
- There are multiple flights in the planning period with different transportation specifications such as cost, capacity, departure time and arrival time, etc.
- Each flight has a normal capacity with normal transportation cost and a special capacity with higher transportation cost for orders that exceed normal capacity.

Normal capacity can be considered as the forecasted capacity, while special capacity can be taken as maximum extension from the forecasted capacity in the planning period.

- Outbound business processing time and cost, together with loading time and loading cost for each flight are included in the air transportation time and cost.
- Local land transportation time and cost are identical for all orders. They are assumed to be included in assembly time and cost.
- Orders released into assembly flow shop in a planning period are delivered within the same planning period, which means that there is no assembly backlog.
- The assembly flow shop is treated as a single machine.
- The capacity of assembly is represented by its production rate.
- Assembly manufacturing time of an order is proportional to the quantity of the order. The per unit product processing time is the reciprocal of production rate.

3.2 Type 1 Air Transportation Problem

Besides the general assumptions, the Type 1 air transportation problem is also based on the assumptions of single destination and split deliveries, which are demonstrated in Figure 3.1.

Figure 3.1 Type 1 air transportation problem

In this section, the Type 1 air transportation problem is formulated using an ILP model, which allocates orders to available transportation capacities with minimum cost. Subsequently, the computational complexity of this problem is analyzed to assist in determining the appropriate method to solve this problem. Results of computational complexity analysis show that this problem can be solved optimally by available ILP solver. For the remaining three air transportation problems, the formulations and the computational complexities are presented in the succeeding sections respectively.

3.2.1 Problem Formulation

Before proposing the ILP model of the Type 1 air transportation problem, the following notation is defined:

i	order index, $i=1, 2, \dots, O$;	f	flight index, $f=1, 2, \dots, F$;
f'	flight index, $f'=1, 2, \dots, F$		
D_f	departure time of flight f at the local place where assembly plant is located		
A_f	arrival time of flight f at the destination		
NC_f	transportation cost for per unit product that is allocated to normal capacity of flight f		
SC_f	transportation cost for per unit product that is allocated to special capacity of flight f		
$NCap_f$	available normal capacity of flight f		
$SCap_f$	available special capacity of flight f		
Q_i	quantity of order i		
$d\alpha_i$	delivery earliness penalty cost (/unit/hour) of order i		
$d\beta_i$	delivery tardiness penalty cost (/unit/hour) of order i		

dd_i delivery due date of order i

WT_i waiting time of order i between assembly and transportation

PE_{if} per unit delivery earliness penalty cost for order i when it is transported by flight f

$$PE_{if} = \text{Max}(0, dd_i - A_f) * d\alpha_i \quad (3.1)$$

PL_{if} per unit delivery tardiness penalty cost for order i when it is transported by flight f

$$PL_{if} = \text{Max}(0, A_f - dd_i) * d\beta_i \quad (3.2)$$

PR production rate of assembly manufacturing

The following decision variables are defined:

Z_{if} quantity of order i allocated to flight f

X_{if} quantity of order i allocated to flight f 's normal capacity

Y_{if} quantity of order i allocated to flight f 's special capacity

The model is expressed and explained as follows:

$$\text{Min} \quad \sum_i \sum_f NC_f X_{if} + \sum_i \sum_f SC_f Y_{if} + \sum_i \sum_f PE_{if} Z_{if} + \sum_i \sum_f PL_{if} Z_{if} \quad (3.3)$$

Subject to:

$$X_{if} + Y_{if} = Z_{if} \quad \text{for all } i, f \quad (3.4)$$

$$\sum_f (X_{if} + Y_{if}) = Q_i \quad \text{for all } i \quad (3.5)$$

$$\sum_i X_{if} \leq NCap_f \quad \text{for all } f \quad (3.6)$$

$$\sum_i Y_{if} \leq SCap_f \quad \text{for all } f \quad (3.7)$$

$$\sum_{f'=1}^f \sum_i (X_{if'} + Y_{if'}) \leq D_f PR \quad \text{for all } f \quad (3.8)$$

The decision variables are: X_{if} , Y_{if} , and Z_{if} . All decision variables are non-negative integer variables. The objective is to minimize total cost which consists of total transportation cost for orders allocated into normal flight capacity, total transportation cost for orders allocated into special flight capacity, total delivery earliness penalty cost and total delivery tardiness penalty cost. Constraint (3.4) ensures that the quantity of the proportion of order i allocated into flight f consists of quantities of the proportion of order i allocated into normal capacity of flight f and the proportion of order i allocated to special capacity of flight f . Constraint (3.5) ensures that order i is completely allocated. Constraint (3.6) ensures that the normal capacity of flight f is not exceeded. Constraint (3.7) ensures that the special capacity of flight f is not exceeded. Constraint (3.8) ensures that allocated orders do not exceed assembly capacity. It ensures that allocated quantity can be supplied by sufficient assembly capacity. Note that the assembly capacity is the accumulated capacity which is indicated in Figure 3.2.

Figure 3.2 Transportation allocation and the accumulated assembly capacity

In Figure 3.2, x axis denotes the time and y axis denotes the volume of products. D_1 to D_5 denote the departure times of five flights. The allocated volume for each flight is denoted by a rectangle. The accumulated allocation till each departure time is specified by broken lined rectangle. Constraint (3.8) ensures that for each flight's departure time D_f , the accumulative allocated volume should not exceed the accumulated assembly capacity until D_f . Otherwise, the allocation is not a feasible solution to the ILP model. Clearly, Figure 3.2 denotes an infeasible solution because the accumulated allocation exceeds the assembly capacity at D_4 . The output of the ILP model is the order allocation to flights.

The assembly due date of an order is determined based on the departure time of the flight that the order is allocated to. For an order, the processing time of each process from release in assembly to final delivery is indicated in Figure 3.3. TA_i denotes the time when order i arrivals air port. As mentioned in section 3.1, business processing time is assumed to be included in the transportation time. To catch the allocated flight, the order must reach the airport before the departure time of the flight that the order is allocated to. Furthermore, since local transportation time is assumed to be included in the assembly processing time, assembly of the order must be completed before the departure time of the flight. Thus, in this research, the assembly due date of each order is set to the departure time of the flight that the order is allocated to.

Figure 3.3 Processes from assembly to delivery

3.2.2 Experimental Design

In order to study the computational complexity of the Type 1 air transportation problem, a series of computational experiments were conducted. As stated by Rardin and Uzsoy (2001), the power of a few available real data sources can be expended by testing on random variations. Thus, in this research, the test problems are generated with the assistance of the representative data from industry practice. For most of the parameters in this experimental design scheme, the values are generated from uniform distributions with the ranges determined based on the industry representative data. Table 3.1 shows the experimental design.

Table 3.1 Experimental design used in random problems generation

Problem Parameter	No. of classes	Values
Number of orders (O)	10	10, 20, 30, 40, 50, 60, 70, 80, 90, 100
Number of flights (F)	10	3, 4, 6, 8, 10, 12, 14, 16, 18, 20
Flight departure time (D_f)		Fixed interval of $24/F$ between departure times
Normal capacity ($NCap_f$)	1	Uniform[200,800]
Special capacity ($SCap_f$)	1	Uniform[100,200]
Transportation cost of per unit product allocated to normal capacity (NC_p)	1	Uniform [50,70]
Transportation cost of per unit product allocated to special capacity (SC_p)	1	Uniform [80,100]
Order quantity (Q_i)	1	Uniform[50,200]
Order due date (dd_i)	1	Uniform[2,26]
Order delivery earliness penalty cost ($d\alpha_i$)	1	Uniform [3,5]
Order delivery tardiness penalty cost ($d\beta_i$)	1	Uniform [5,8]
Production rate (PR)	1	$TQ^a * \text{Uniform} [1,1.5] / 24$
Instance/problem	5	

^a TQ means total quantity which equals to the sum of all the order's quantity.

The number of orders O ranges from 10 to 100, with the number of flights F ranges from 3 to 20. The problem size is determined by the number of orders and the corresponding number of flights. Generally, O equals to $5F$ in each problem except the first problem with 10 orders and 3 flights. Thus, there are total of 10 problems generated. Five instances are generated for each problem, which gives a total of 50 test problems.

The planning period is set to 24 hours for this problem. For each generated problem, the first flight's departure time is set to $24/F$. There is fixed interval of $24/F$ between the departure times of every two adjacent flights. Thus the remaining flights' departure times are determined. Transportation time of each flight TDA_f from departure to arrival is set to 2 hours (as only one destination is considered) based on the information obtained from the industry, i.e., most of the surrounding important cities can be reached from Singapore within 2 hour's air transportation time. Order due date is drawn from a uniform distribution between 2nd and 26th hour. PR is an important parameter in the ILP model, which constraints the allocation of air transportation. Once the quantity of each order for a given instance is generated, the value of PR is generated from a discrete uniform distribution as: $PR = TQ * \text{Uniform}[1, 1.5]/24$, where TQ is the total quantity which equals to the sum of all the order quantities. It is noted that only the instances with PR and other parameters satisfying constraint (3.8) are feasible instances. Hereafter, instance refers to the feasible ones. Lingo 8.0 software is used to construct the mathematical programming models. All the tested problems are solved on a Pentium 4, 2.4 GHz computer with 256 MB RAM.

3.2.3 Computational Complexity Analysis

A set of test problems are generated following the experiment design presented in previous section. The computational time to obtain the optimal solution of each instance for each problem is collected and presented in Table 3.2. For each problem, the average computational time is also shown in the table. In this section, the terminology ‘order’ can also be denoted by ‘job’. The notation of $OjFf$ is proposed to denote each test problem, where O is the number of jobs and F is the number of flights in the problem. For example, there are 10 jobs and 3 flights in the first problem and it is denoted by 10j3f.

Table 3.2 Computational times for tested Type 1 air transportation problems

Problem	Computational time (second)					
	Instance 1	Instance 2	Instance 3	Instance 4	Instance 5	Average
10j3f	0 ^a	0	0	0	0	0
20j4f	0	0	0	0	0	0
30j6f	1	1	1	1	1	1
40j8f	2	2	2	2	2	2
50j10f	6	2	3	2	2	3
60j12f	4	4	4	4	3	3.6
70j14f	7	7	7	6	7	6.8
80j16f	10	9	10	11	10	10
90j18f	13	18	19	12	13	15
100j20f	18	18	18	18	27	19.8

^a as shown in status penal of Lingo solver, the solving time is 00:00:00, which means the computational time is far less than 1 second.

It is observed that the computational time is within 30 seconds even for the largest test problem 100j20f. As the problem size of 100j20f is large enough in industry practice, it

can be stated that the computational time is short for the Type 1 air transportation problem.

The results of average computational time of each problem provide a rough impression about the computational complexity of the Type 1 air transportation problem. More specifically, the equality of the Type 1 air transportation problem with the unbalanced transportation problem is established. As the unbalanced transportation problem is solvable in polynomial time (Winston, 1994; Alidaee and Ahmadian, 1996), the Type 1 air transportation problem is also solvable in polynomial time.

In general, a transportation problem is specified by three elements: a set of supply points, a set of demand points and cost of shipping each unit from a supply point to a demand point. If demand is larger than supply or supply is larger than demand, it is called an unbalanced transportation problem (Winston, 1994). For the Type 1 air transportation problem, each order is taken as a supply point and each flight's capacity is taken as a demand point. It is noted that the normal capacity and special capacity of each flight are considered as two demand points because of different transportation cost. The unit transportation cost from a supply point sp to a demand point dp is the sum of the unit transportation cost of order sp that is allocated to the flight dp and the delivery earliness or tardiness penalty cost of order sp transported by flight dp . As total quantity of all orders is less than total capacity of all flights, it is an unbalanced transportation problem. The equality establishment is also demonstrated in Figure 3.4.

Figure 3.4 Equality of Type 1 air transportation problem to the unbalanced transportation problem.

Considering the constraint of assembly capacity on the allocation that represented by constraint (3.8), the usable normal and special capacities of each flight are less than the available capacities, which are the input data for the ILP model. For instance, the available normal capacity and special capacity on flight f are 200 and 100 respectively. Flight f departs at 5 am. However, according to assembly capacity, there are only 250 units of products can be produced. Thus, only 250 units of capacity on flight f is usable. Furthermore, since transportation cost in special capacity is larger than normal capacity, there are 200 units of normal capacity and 50 units of special capacity are usable. Therefore, take into account the assembly capacity constraint, the air transportation problem still can be taken as an unbalanced transportation problem.

3.3 Type 2 Air Transportation Problem

Besides the general assumptions listed in section 3.1, the Type 2 air transportation problem is also based on the assumptions of single destination and non-split deliveries, which are demonstrated in Figure 3.5.

Figure 3.5 Type 2 air transportation problem

In this problem, one order can only be allocated to one flight. Furthermore, the order can only be allocated to one type of capacity of the flight, either the normal capacity or the special capacity. In this section, the formulation of the Type 2 air transportation problem is proposed. The computational complexity of the problem is also investigated by computational experiments. The computational results show that the Type 2 air transportation problem is computationally intractable. The NP-completeness of this problem is also established in this section.

3.3.1 Problem Formulation

Prior to present the formulation, the following decision variables are defined, which are not defined in section 3.2.1.

HX_{if} 1 if order i is allocated to flight f 's normal capacity, 0 otherwise.

HY_{if} 1 if order i is allocated to flight f 's special capacity, 0 otherwise.

The model is expressed as follows:

Min

$$\begin{aligned} \sum_i \sum_f NC_f HX_{if} Q_i + \sum_i \sum_f SC_f HY_{if} Q_i \\ + \sum_i \sum_f PE_{if} (HX_{if} + HY_{if}) Q_i + \sum_i \sum_f PL_{if} (HX_{if} + HY_{if}) Q_i \end{aligned} \quad (3.9)$$

Subject to:

$$\sum_f (HX_{if} + HY_{if}) = 1 \quad \text{for all } i \quad (3.10)$$

$$\sum_i HX_{if} Q_i \leq NCap_f \quad \text{for all } f \quad (3.11)$$

$$\sum_i HY_{if} Q_i \leq SCap_f \quad \text{for all } f \quad (3.12)$$

$$\sum_{f'=1}^f \sum_i (HX_{if'} + HY_{if'}) Q_i \leq D_f PR \quad \text{for all } f \quad (3.13)$$

$$HX_{if}, HY_{if} \in \{0, 1\} \quad \text{for all } i, f \quad (3.14)$$

The decision variables are: HX_{ij} , HY_{ij} . All decision variables are 0-1 integer variables, which are indicated by constraint (3.14). The objective is to minimize total cost which consists of total transportation cost for orders allocated into normal flight capacity, total transportation cost for orders allocated into special flight capacity, total delivery earliness penalty cost and total delivery tardiness penalty cost. Constraint (3.10) confirms that one order is allocated to exactly one flight, furthermore, to only one type of capacity on the flight. The implications of constraint (3.11) to (3.13) are similar to

that of the constraint (3.6) to (3.8). More detailed implications of these three constraints can be found in section 3.2.1.

3.3.2 Computational Intractability of Type 2 Air Transportation Problem

Firstly, the computational complexity of the Type 2 air transportation problem is investigated by computational experiments. The test data are generated following the experiment design proposed in section 3.2.2. Five instances are generated for each test problem. Table 3.3 shows the computational time to obtain the optimal solution for each instance of the first three problems by Lingo 8.0.

Table 3.3 Computational times for tested Type 2 air transportation problems

Problem	Computational time					
	Instance 1	Instance 2	Instance 3	Instance 4	Instance 5	Average
10j3f	0 ^a	0	1s	1s	0	0.4s
20j4f	1s	0	0	0	1s	0.4s
30j6f	6m46s	9h19m9s	>12h	>12h	>12h	^b

^a as shown in Lingo solver status penal, the solving time is 00:00:00

^b average computational time can not be obtained as the computational times of the last three instances are unknown.

It is observed that for small sized problems, i.e., the first two problems, the computational time to obtain the optimal solution is trivial. However, the computational time increases significantly for most of the instances of problem 30j6f. For problem 30j6f, instance 1 has the smallest computational time of 6 minutes and 46 seconds according to Table 3.3. Then, the computational time increases significantly to 9 hours for instance 2. Furthermore, for each of the three remaining instances, the computational time is more than 12 hours. After 12 hours, the computation of Lingo is

terminated due to the unacceptably long computational time, as a production shift is normally less than 12 hours. These results clearly demonstrate the computational intractability of the Type 2 air transportation problem.

3.3.3 NP-Completeness of Type 2 Air Transportation Problem

Through computational experiments, the problem is shown to be computationally intractable. This section presents the NP-completeness proof of the Type 2 air transportation problem. The problem can also be stated as: *Is there an allocation schedule to allocate all the orders into the available flights with sum of total delivery earliness, tardiness and transportation cost no larger than some value of ρ ?* To show that this problem is NP-complete, we provide a reduction of the Type 2 air transportation problem to a partition problem (Garey and Johnson, 1979), which is proved to be NP-complete.

Partition: Given positive integers n , B and a set of positive integers $G=\{x_1, x_2, \dots, x_n\}$ with $\sum_{i=1}^n x_i = 2B$, is there a subset $G_1 \subseteq G$ such that $\sum_{x_i \in G_1} x_i = \sum_{x_i \in G - G_1} x_i = B$?

The reduction of the Type 2 air transportation problem means that additional restrictions be placed on the instances of the problem so that the resulting restricted problem will be identical to the partition problem.

Theorem 3.1 The Type 2 air transportation problem is NP-complete.

Proof. We now show how to reduce the Type 2 air transportation problem to the partition problem in detail. In the Type 2 air transportation problem, each order is characterized by $(Q_i, dd_i, d\alpha_i, d\beta_i)$ which are order quantity, order due date, delivery earliness penalty cost, and delivery tardiness penalty cost respectively. Each flight is characterized by $(D_f, A_f, NCap_f, SCap_f, NC_f, SC_f)$ which are departure time, arrival time, normal capacity, special capacity, cost to carry per unit product allocated to normal capacity and cost to carry per unit product allocated to special capacity respectively. The input data consists of O orders and F flights:

1. Order set $OS = \{O_i | O_i = (Q_i, dd_i, d\alpha_i, d\beta_i) \text{ for } i=1, 2, \dots, O\}$
2. Flight set $FS = \{F_f | F_f = (D_f, A_f, NCap_f, SCap_f, NC_f, SC_f) \text{ for } f=1, 2, \dots, F\}$

Suppose there are two identical flights to transport all the orders, which means identical value of above mentioned parameters for the two flights. Therefore, the parameters of the two flights can be denoted by $(D, A, NCap, SCap, NC, SC)$. Thus, each order to be transported by any of the two flights, if allocated to the same capacity (such as normal capacity), will lead to identical total cost, which consists of transportation cost, delivery earliness cost and delivery tardiness cost. The following notation is defined:

PE_i per unit delivery earliness penalty cost for order i when it is transported by any flight

$$PE_i = \text{Max}(0, dd_i - A) * d\alpha_i \quad (3.15)$$

PL_i per unit delivery tardiness penalty cost for order i when it is transported by any flight

$$PL_i = \text{Max}(0, A - dd_i) * d\beta_i \quad (3.16)$$

NTC_i the total cost of order i that is allocated to the normal capacity of a flight

$$NTC_i = (NC + PE_i + PL_i) * Q_i \quad (3.17)$$

STC_i the total cost of order i that is allocated to the special capacity of a flight

$$STC_i = (SC + PE_i + PL_i) * Q_i \quad (3.18)$$

In order to reduce the Type 2 air transportation problem, the following constraints are introduced:

$$PR \times D \geq \sum_i Q_i \quad (3.19)$$

$$Q_i = x_i \quad \text{for all } i \quad (3.20)$$

$$y = \sum_i NTC_i / 2 \quad (3.21)$$

$$NCap = \sum_i Q_i / 2 \quad (3.22)$$

Constraint (3.19) ensures that production rate is large enough for the allocation so that PR does not constraint the allocation in air transportation any more. Constraint (3.20) indicates that for each element in G , there is an order corresponds to it, where the quantity of the order equals to the value of the element. Let G' denotes the corresponding order set. Constraint (3.21) defines a variable y , which will be used in the following illustration. Constraint (3.22) denotes that total normal capacity of the two flights equals to total quantity of all the orders. The notation **T2R** is defined to denote this special instance of the Type 2 air transportation problem.

It is shown in the following that there exists a solution for **T2R** with total cost less or equal to $2y$ if and only if there exists a desired solution for **Partition**.

If Part: let G_I be a desired solution for **Partition**. According to Constraint (3.20), the order set G' can be separated into two partitions based on G_I . Let G_I' be the corresponding partition order set. Clearly, the two order partitions of G_I' and $G' - G_I'$ can be allocated to normal capacity of each flight. The total cost is $\sum_i NTC_i$, which equals to $2y$.

Only if part: Let σ_a be an air transportation allocation such that total cost is less than or equal to $2y$. It implies that all orders are allocated to normal capacity. If any order is allocated to special capacity, the total cost must exceed $2y$. As the normal capacity of the two flights are identical, together with the constraint of total normal capacity being equal to total order quantity that indicated by constraint (3.22), it is derived that the order set G' is divided into two partitions and are allocated to each flight's normal capacity respectively. As the normal capacity of each flight is identical, the total quantities of the two order partitions should be identical too. Therefore, based on constraint (3.20), the partition problem has a desired solution if the allocation problem has a solution with total cost less than or equal to $2y$.

It is observed from the above illustration that there exists a solution for **T2R** with total cost less than or equal to $2y$ if and only if there exists a desired solution for **Partition**.

As **Partition** is NP-complete, **T2R** is also NP-complete. Thus, the Type 2 air transportation problem is NP-complete. $\square$

3.4 Type 3 Air Transportation Problem

Type 3 air transportation problem is assumed to have multiple destinations. For each destination, there are several flights available with different transportation cost, capacity and departure time. Split deliveries are permitted. All the other assumptions follow the assumptions listed in section 3.1. Figure 3.6 demonstrates the Type 3 transportation problem.

Figure 3.6 Type 3 air transportation problem

3.4.1 Problem Formulation

The Type 3 air transportation problem is formulated using an ILP model and presented based on the following notation, which are not defined in the previous sections.

g destination index, $g=1,2,\dots,L$

Des_i order i 's destination

Des_f	flight f 's destination
LN	a large positive number
$ LN $	absolute value of LN

The objective function and majority of the constraints of this ILP model are identical to the ILP model of the Type 1 air transportation problem. This ILP model can also be expressed by objective function (3.3) and constraints (3.4) to (3.8), together with two more constraints.

$$\text{Min} \quad \sum_i \sum_f NC_f X_{if} + \sum_i \sum_f SC_f Y_{if} + \sum_i \sum_f PE_{if} Z_{if} + \sum_i \sum_f PL_{if} Z_{if} \quad (3.3)$$

Subject to:

$$X_{if} + Y_{if} = Z_{if} \quad \text{for all } i, f \quad (3.4)$$

$$\sum_f (X_{if} + Y_{if}) = Q_i \quad \text{for all } i \quad (3.5)$$

$$\sum_i X_{if} \leq NCap_f \quad \text{for all } f \quad (3.6)$$

$$\sum_i Y_{if} \leq SCap_f \quad \text{for all } f \quad (3.7)$$

$$\sum_{f'=1}^f \sum_i (X_{if'} + Y_{if'}) \leq D_f PR \quad \text{for all } f \quad (3.8)$$

$$LN * X_{if} * |Des_i - Des_f| < 1 \quad \text{for all } i, f \quad (3.23)$$

$$LN * Y_{if} * |Des_i - Des_f| < 1 \quad \text{for all } i, f \quad (3.24)$$

Constraints (3.23) and (3.24) ensure that order i can not be allocated to flight f if order i and flight f have different destination. For detailed implications of the objective function and other constraints, see section 3.2.1.

3.4.2 Experimental Design

In order to investigate the computational complexity of the Type 3 air transportation problem, a series of computational experiments were carried out. The experiment design in section 3.2.2 is reused. Specifically, there are modifications for the configurations of some parameters in Table 3.1. Also, two parameters are introduced: order destination and flight destination. Table 3.4 shows the experimental design for the modified parameters and the newly introduced parameters. Section 3.2.2 provides more detailed configuration of other parameters.

Table 3.4 Experimental design for modified and newly introduced parameters

Problem Parameter	No. of classes	Values
Number of orders (O)	9	20, 30, 40, 50, 60, 70, 80, 90, 100
Number of flights (F)	9	4, 6, 8, 10, 12, 14, 16, 18, 20
Number of destinations (L)	4	2, 2, 3, 3, 3, 4, 4, 4, 5
Order due date (dd_i)	1	Uniform[1, 6]*($p_i + TDA_j$)
Order destination (Des_i)	1	Uniform [1, L]
Flight destination (Des_j)	1	Uniform [1, L]
Flight departure time (D_f)		Uniform [$24*(FN_f-1)/TF_g$, $24*FN_f/TF_g$]
Transportation cost of per unit product allocated to normal capacity (NC_f)	1	Uniform [30+20g, 50+20g]
Transportation cost of per unit product allocated to special capacity (SC_f)	1	Uniform [60+20g, 80+20g]
Production rate (PR)	1	$TQ^a * \text{Uniform} [1.2, 2] / 24$
Instance/problem	5	

^a TQ means total quantity which equals to the sum of all the order's quantity.

The number of order O ranges from 20 to 100, while the number of flights F ranges from 3 to 20 and the number of destinations L ranges from 2 to 5. The problem size is determined by the number of orders, the corresponding number of flights and the number of destinations. It is noted that as problem with 10 orders is relatively a small sized problem to have multiple destination, the size of tested problem starts from 20 orders. Generally, O equals to $5F$ in each problem. The destination for each order and each flight is generated from uniform distribution between 1 and the number of destinations of the corresponding problem configuration. There are 9 test problems in this experiment design. Five instances are generated for each of the 9 test problems, which give a total of 45 test problems.

For each problem, the departure time of each flight is generated from a uniform distribution subject to its destination. The total number of flights that have the same destination g is denoted by TF_g . The corresponding flights are assigned a flight number FN_f , which starts from 1 to TF_g . Each flight's departure time is then generated using uniform distribution between $[24*(FN_f-1)/TF_g, 24*FN_f/TF_g]$. The flight transportation time to each destination is generated subject to the specific destination. As air transportation time to a specific destination is normally stable, and to simplify the experiment design further, each flight's transportation time TDA_f is given and set to the value of its destination number. For example, if a flight's destination is destination 1, the transportation time is set to 1 hour. The transportation times are generated based on the industry practice information. Normally, transportation cost is different subject to different destination. The normal transportation cost is drawn from uniform $[30+20g, 50+20g]$ and the special transportation cost is drawn from uniform

$[60+20g, 80+20g]$, where g is the destination number of the flight. Larger value of g indicates longer transportation time, which results in higher transportation cost.

The planning period is set to 24 hours for each problem. Order due date is drawn from uniform distribution considering assembly and transportation lead time. The earliest delivery time of an order is the sum of assembly processing time and transportation time. Therefore, the range for order due date is between $p_i + TDA_f$ and $6(p_i + TDA_f)$, where p_i is the order's assembly processing time and TDA_f is the flight transportation time to its destination. Once the quantity of each order for a given instance is generated, the value of production rate is generated from a discrete uniform distribution as: $PR = TQ * \text{Uniform}[1.2, 2]/24$, where TQ is total quantity which equals to the sum of all the orders' quantity. Lingo 8.0 software is used to construct the ILP models. All the tested problems are solved on a Pentium 4, 2.4 GHz computer with 256 MB RAM.

3.4.3 Computational Complexity Analysis

The computational times to obtain optimal solution of each instance for each problem are collected and reported in Table 3.5. The problem is denoted by $OjFfLd$ where O is the number of jobs (or orders), F is the number of flights and L is the number of destinations in the problem. For example, there are 20 jobs, 4 flights and 2 destinations in the first problem and it can be denoted by 20j4f2d. For each problem, the average computational time is also shown in Table 3.5.

Table 3.5 Computational times for tested Type 3 air transportation problems

Problem	Computational time (s)					
	Instance 1	Instance 2	Instance 3	Instance 4	Instance 5	Average
20j4f2d	0 ^a	0	0	0	0	0
30j6f2d	1	0	0	0	0	0.2
40j8f3d	0	1	0	1	1	0.6
50j10f3d	1	0	1	0	0	0.4
60j12f3d	1	1	1	2	1	1.2
70j14f4d	1	2	1	1	1	1.2
80j16f4d	1	1	2	2	2	1.6
90j18f4d	2	3	3	3	3	2.8
100j20f5d	3	3	3	3	3	3

^aas shown in Lingo solver status penal, the solving time is 00:00:00.

It is observed that the computational time is within 3 seconds even for the largest test problem of 100j20f5d. As the problem size of 100j20f5d is close to the problem size in industry practice, it can be stated that the computational time is very short for Type 3 air transportation problem.

Comparing the results in Table 3.2 and Table 3.5, it is also observed that the average computational time of the Type 3 air transportation problem is less than the Type 1 air transportation problem, which is investigated in section 3.2. This is due to the reduction of decision variables in Type 3 air transportation problem compared to Type 1 air transportation problem. In Type 3 air transportation problem, one order can only be allocated to the flights with the same destination as the order. The number of decision variables X_{if} , Y_{if} is significantly reduced. While in Type 1 air transportation problem, one order can be allocated to any flight. Each pair of if leads to a decision variable. As

the decision variables are reduced in Type 3 air transportation problem, the computational time is also reduced.

3.5 Type 4 Air Transportation Problem

Besides the general assumptions presented in section 3.1, Type 4 air transportation problem is also assumed to have multiple destination and non-split deliveries. The following figure demonstrates this problem.

Figure 3.7 Type 4 air transportation problem

Similar to the Type 2 air transportation problem, which is formulated in section 3.3, one order can only be allocated to one flight in Type 4 air transportation problem. In addition, the order can only be allocated one type of capacity of the flight, either the normal capacity or the special capacity.

3.5.1 Problem Formulation

The Type 4 air transportation problem can also be formulated using an ILP model. The objective function and most of the constraints of this ILP model are identical to the ILP

model of the Type 2 air transportation problem. This ILP model can also expressed by objective function (3.9) and constraints (3.10) to (3.14), together with two new constraints.

Min

$$\begin{aligned} \sum_i \sum_f NC_f HX_{if} Q_i + \sum_i \sum_f SC_f HY_{if} Q_i \\ + \sum_i \sum_f PE_{if} (HX_{if} + HY_{if}) Q_i + \sum_i \sum_f PL_{if} (HX_{if} + HY_{if}) Q_i \end{aligned} \quad (3.9)$$

Subject to:

$$\sum_f (HX_{if} + HY_{if}) = 1 \quad \text{for all } i \quad (3.10)$$

$$\sum_i HX_{if} Q_i \leq NCap_f \quad \text{for all } f \quad (3.11)$$

$$\sum_i HY_{if} Q_i \leq SCap_f \quad \text{for all } f \quad (3.12)$$

$$\sum_{f'=1}^f \sum_i (HX_{if'} + HY_{if'}) Q_i \leq D_f PR \quad \text{for all } f \quad (3.13)$$

$$HX_{if}, HY_{if} \in \{0, 1\} \quad \text{for all } i, f \quad (3.14)$$

$$LN * HX_{if} * |Des_i - Des_f| < 1 \quad (3.25)$$

$$LN * HY_{if} * |Des_i - Des_f| < 1 \quad (3.26)$$

Constraints (3.25) and (3.26) ensure that if order i and flight f have different destination, order i can not be allocated to flight f . For detailed implications of the objective function and other constraints, see section 3.3.1.

3.5.2 NP-Completeness of Type 4 Air Transportation Problem

The NP-completeness of Type 4 air transportation problem can be established based on the NP-completeness of Type 2 air transportation problem. Consider a special case of the Type 4 air transportation problem with two destinations. To each destination, there are two identical flights. Furthermore, similar to the NP-complete proof of the Type 2 air transportation problem, the following constraint is introduced, which relaxes the production rate constraint.

$$PR \times D_f \geq \sum_i Q_i \quad \text{for all } f \quad (3.27)$$

Clearly, the overall problem can be decomposed into two sub-problems. Each sub-problem is a single destination air transportation problem. Moreover, each sub-problem is a **T2R** problem, which is a special case of the Type 2 air transportation problem. As **T2R** is proved to be NP-complete in section 3.3.3, this special case of the Type 4 air transportation problem is also NP-complete. Therefore, Type 4 air transportation problem is NP-complete.

3.6 Summary

In this chapter, four air transportation problems are studied based on different assumptions on the destinations and order split in delivery. ILP model for each problem is proposed. The computational complexities of these problems are also investigated through both computational experiments and mathematical analysis. The results showed that Type 1 and Type 3 air transportation problems can be solved optimally using available ILP solver, while Type 2 and Type 4 air transportation problems are NP-complete. Optimal algorithms can be developed for small sized Type 2 and Type 4 air

transportation problems, such as Branch and Bound algorithms. Heuristics can be developed for large sized problems.

The assembly scheduling problem, which is the second part of the CESC synchronization problem, is studied in the following chapters.

Chapter 4: The Single Machine Assembly Scheduling Problem

This chapter addresses the assembly scheduling problem, which is the second sub-problem of the overall CESC synchronization problem. Specifically, the single machine assembly scheduling problem is studied.

4.1 Introduction and Assumptions

In the formulation of the air transportation problems, the assembly production rate constraints the allocation. In other words, the assembly of each order can be completed before the departure time of the flight that the order is allocated to. Thus, the assembly scheduling problem considered in this chapter is a single machine scheduling problem to minimize total earliness without tardiness. The ultimate result of the schedule is a proper release time for each order.

The assumptions, based on which the assembly scheduling problem is formulated, are presented as follows:

- Setup time is not considered.
- Job data are known and deterministic. Assembly processing time of a job is directly proportional to the job's quantity. The per unit product processing time is the reciprocal of production rate.
- Each machine processes only one job at a time.

- Pre-emption is not allowed, which means once a job's processing is started, it cannot be interrupted by another job.
- The capacity of assembly is represented by assembly production rate. It is assumed to be constant.

The above assumptions simplify the real assembly scheduling problem. The study in this chapter aims to gain some basic knowledge of the assembly scheduling problem. The more complicated situations of assembly that are closer to real practice, such as parallel machine or considering machine breakdown, will be investigated later based on the work done in this chapter.

4.2 Solution Methodologies

In the assembly scheduling problem, a release time is determined for each order to minimize the average waiting time before air transportation. Average waiting time is defined as the mean of sum of the waiting times for all orders between assembly and air transportation. Thus, the earliness weight is job independent. By solving the ILP model of the air transportation problem, the allocation results provide the inputs for the assembly problem which include the orders, quantity and transportation departure time. Orders may be split and allocated to different flights and delivered independently. The split orders will be treated as independent orders in assembly. For each order, the assembly due date is set to the departure time of the flight that the order is allocated to, which is indicated in section 3.2.1. Generally, jobs allocated to different flights are assigned different assembly due dates and vice versa. Therefore, the assembly due dates are distinct. Meanwhile, jobs allocated to the same flight share a common assembly due

date. Thus, the overall due dates are expressed by the defined term of ‘distinct common due dates’ in this chapter.

As mentioned in section 3.2.1, one important constraint of the transportation ILP model is the production rate of assembly, i.e., enough assembly capacity to supply the products to be allocated in air transportation. For a set of orders sharing a common due date, the due date should be larger or equal to the sum of the processing times of all these orders. Otherwise, orders completed later than their assembly due dates will miss the flight. Therefore, each common due date is unrestricted according to the definition of unrestricted common due date in Kanet (1981).

Chand and Schneeberger (1988) studied the single machine scheduling problem where the objective is to minimize total weighted earliness without tardy jobs. The due dates are distinct and the earliness weights are job dependent. There is no inserted idle time in the job sequence. The problem is proved to be NP-hard. Pathumnakul and Egbelu (2005) studied the same problem, while considering inserted idle time in the job sequence. As the problem is NP-hard, heuristics are developed to solve this problem. The single machine assembly scheduling problem studied in this chapter is a special case of the problem addressed by Chand and Schneeberger (1988) and Pathumnakul and Egbelu (2005) due to the features of job independent earliness weights and the distinct common due dates. In this section, a forward scheduling algorithm and a backward scheduling algorithm are proposed to solve this problem.

4.2.1. Forward Scheduling Algorithm

Normally in practice, the schedules are generated using dispatching rules and following a forward scheduling method. Forward scheduling algorithm schedules the jobs in a given sequence one by one, starting from the beginning of the planning period, to achieve feasible and compact schedules. The approach usually generates non-delay schedules. A non-delay schedule is one in which no machine is kept idle at any time when at least one job is waiting for processing. Longest-Processing-Time (LPT) is selected as the dispatching rule in this forward scheduling algorithm. It is proved by Panwalkar et al. (1982) that LPT rule minimizes total earliness for the single machine assembly scheduling in the situation of no tardiness. Since there may be split orders in allocation of air transportation, the sequence determined by LPT rule may be adjusted to combine the split orders to facilitate assembly processing without resulting in any delay of allocation in air transportation.

The general scheme for the forward scheduling algorithm is described by the following steps:

Step 1: For each flight, place all the orders that are allocated to this flight into an order group. Thus, the number of order groups corresponds to the number of flights. Sequence the order groups by the sequence of earliest flight departure time first.

Step 2: For each order group, use LPT rule to sequence the orders within this group.

Step 3: Assembly batching of split orders: This step is used to combine the split orders in a batch for assembly so that split orders in transportation can be treated as a whole order in assembly. This step is applied only in the situation when an order

is split and allocated to two adjacent flights. If the first proportion of the split order is sequenced to be the last one in the order sequence of the first flight, the next proportion of the split order which is allocated to the second flight is adjusted to be the first one in the second flight's order sequence. Similarly, if the second proportion of the split order is sequenced to be the first one in the order sequence of the second flight, the first proportion of the split order which is allocated to the first flight is adjusted to be the last one in the first flight's order sequence. This is to facilitate the assembly processing of an order.

Step 4: Calculate each order's release time by forward scheduling logic starting from the first order to the last order in the whole order sequence without inserted idle time. Note that the whole order sequence consists of the sequence of each order group. If order i is the first order in the whole order sequence, then its assembly release time equals to zero. Else, its assembly release time equals to the release time of order $i-1$ plus assembly processing time of order $i-1$.

Step 5: Compute the average waiting time between assembly and transportation.

The order groups are determined by transportation allocation model. As mentioned in step 1 of this forward scheduling algorithm, the number of groups in the planning period equals to the number of flights that transport orders to customer destinations. Each order group consists of a set of orders and they have same assembly due-date. To determine order release time by forward scheduling logic, the whole order sequence is important regardless of which order is allocated to which flight. To illustrate the

forward scheduling algorithm, consider a problem having four orders and three flights that defined in Table 4.1.

Table 4.1 Order processing times and air transportation allocation

Order	Assembly Processing time	Allocated to flight
1	2	1
2	2	2
3(1)	1	2
3(2)	1.5	3
4	3	3

Order 3(1) and 3(2) in the first column of Table 4.1 indicate that order 3 is split into two sub-orders. The transportation allocation of each order is reported in the third column of Table 4.1. Each flight's departure time can be obtained from Table 4.2.

Table 4.2 Flight departure times

Flight	Departure time
1	4
2	8
3	12

Figure 4.1 displays the resulting assembly schedule by applying the forward scheduling algorithm in a Gantt chart. The x axis denotes time. This Gantt chart shows the schedule from the perspective of the time each order is scheduled for assembly processing.

Figure 4.1 The schedule generated by the forward scheduling algorithm

By applying the first two steps of forward scheduling algorithm, the sequence in the last group should be 4-3(2). However, after the step 3 applied, the two orders are interchanged. It is because that 3(1) is sequenced as the last order in the second order group. Therefore, by adjusting 3(2) to be the first order in the third order group, 3(1) and 3(2) can be processed one after another which means order 3 can be processed without interruption in assembly. The results generated by the forward scheduling algorithm, i.e., the value of average waiting time before air transportation is 3.4.

4.2.2 Backward Scheduling Algorithm

Forward scheduling algorithm is a straightforward method that schedules jobs one by one from the beginning of the planning period. Its main objective is to ensure that each order can meet its assembly due date. However, the forward scheduling algorithm does not minimize average waiting time effectively. As stated by Baker and Scudder (1990), for a single machine earliness scheduling problem with distinct due dates, the search for an optimal solution consists of two steps: finding a good job sequence and inserting idle times in the job sequence. Forward scheduling algorithm creates a good job sequence in the first three steps. However, there is no idle time inserted in the job sequence when the release time of each job is determined in step 4. The drawback of the forward scheduling algorithm is overcome by adopting a backward scheduling algorithm that inserts idle times among order groups. Steps of the backward scheduling algorithm are the same as the forward scheduling algorithm except for step 4. Backward scheduling logic is employed in step 4 of the backward scheduling algorithm instead of forward scheduling logic. Therefore, only the step 4 of the backward scheduling algorithm is given below:

Step 4: Calculate each order's release time by backward calculation using the backward scheduling logic starting from the last order of the last order group.

The following pseudo-code describes the backward scheduling logic in the backward scheduling algorithm:

```

If (job  $i$  is the last job in flight  $f$ ) then
    If (flight  $f$  is the last flight) then
        Release time(job  $i$ , flight  $f$ ) = Departure time(flight  $f$ ) –
        Processing time(job  $i$ , flight  $f$ )
    Else
        If (Release time (the first job, flight  $f+1$ ) is earlier than
        Departure time(flight  $f$ )) then
            Release time (job  $i$ , flight  $f$ ) = Release time (the first job,
            flight  $f+1$ ) – Processing time(job  $i$ , flight  $f$ )
        Else
            Release time (job  $i$ , flight  $f$ ) = Departure time (flight  $f$ ) –
            Processing time (job  $i$ , flight  $f$ )
        End if
    End if
Else
    Release time (job  $i$ , flight  $f$ ) = Release time (job  $i+1$ , flight  $f$ ) –
    Processing time (job  $i$ , flight  $f$ )
End if

```

To illustrate the backward scheduling algorithm, consider the problem defined in Table 4.1 and Table 4.2. Figure 4.2 displays the resulting schedule obtained by applying the backward scheduling algorithm in a machine Gantt chart. It is observed from Figure 4.2 that each job is scheduled closer to its due date compared to the schedule in Figure 4.1. Also, there is inserted idle time between job 1 and 2. The result of average waiting time is 1, which is only 29.4% of the average waiting time obtained using forward scheduling algorithm.

Figure 4.2 The schedule generated by the backward scheduling algorithm

Note that without the process of assembly batching of split orders, the backward scheduling algorithm provides optimal solution to this single machine assembly scheduling problem. For each order group, orders share a common due date. As the objective is to minimize average waiting time without tardiness, there should be no inserted idle time among jobs within an order group. Orders are sequenced by the LPT rule, which minimize total earliness (Panwalkar, 1982). If the completion time of the last job is determined, the minimum earliness for this order group is determined. In line with the objective of minimizing total earliness without tardiness, the best completion time of the last job for an order group is set to the common due date of the orders within this group, which can be indicated by job 4 in Figure 4.2. However, in some cases, the completion time of the last job can not be scheduled to the common due date of the

order group. As shown in Figure 4.2, the last order 3(1) in order group 2 can not be scheduled to be completed at 8 as the first job 3(2) in order group 3 is released before 8. In this case, the best completion time of the last job in order group 2 is set to the release time of the first job in order group 3.

Therefore, without the step of assembly batching of split orders, the backward scheduling algorithm guarantees optimal solution to this single machine assembly scheduling problem based on three aspects: (1) Sequence orders by LPT rule in each order group, (2) No idle time inserted in the order sequence of each order group, (3) Set the completion time of the last order in each order group to the corresponding commonly due date. If that is not possible, set to the release time of the first order in the adjacent order group that has larger common due date. If the step of assembly batching of split orders is applied, the backward scheduling algorithm provides good solution which is close to the optimal solution. It is validated by comparing its results with that of forward scheduling algorithm as well as one industry practice heuristic in section 4.4.

4.3 Detail of the Synchronized Scheduling Methodology

As mentioned in chapter 1, the synchronized scheduling methodology decomposes the overall CESC synchronization problem into air transportation problem and assembly scheduling problem. The air transportation problems are formulated in chapter 3. The single machine assembly scheduling problem can also be solved by the backward scheduling algorithm, which is presented in section 4.2.2. Thus, the detailed picture of

SSM for solving the CESC synchronization problem with single machine in assembly can be depicted by the flow diagram in Figure 4.3.

Figure 4.3 Flow diagram of SSM

There are intermediate results generated in the procedure of SSM, which includes flight allocation results, order sequence before and after the assembly batching of split orders procedure (step 3 of the backward scheduling algorithm), etc. To demonstrate the working mechanism of SSM more clearly, a detailed numerical example is presented in Annexure of this chapter.

It is noted that the order release time obtained following the backward scheduling algorithm should be updated from the perspective of real practice. In real practice, more than one job is processed at the same time on different assembly lines in the assembly flow shop. It is inconsistent with the assumption that only one job is processed in assembly at a time, which is presented in section 4.1. There are several hours for each unit of product from the releasing to the completion. This lead time is generally stable if no machine breakdowns in assembly. Therefore, taking these issues into consideration, the release time in practice could be obtained by adding up the release time obtained in this research and the assembly lead time in real practice.

4.4 Computational Results

The objective of the computational study in this section is primarily to analyze the relative performance of the proposed SSM methodology with industry practice approaches as well as to compare the performance of the backward scheduling algorithm and the forward scheduling algorithm. To evaluate the contribution of SSM, it is compared with the EDD+ FCFS heuristic which is commonly used in the current industry practice. The heuristic mainly consists of two steps:

Step 1: Schedule jobs using Earliest Due Date (EDD) rule for the assembly. The accepted orders corresponding to the planning period are assigned into assembly manufacturing based on their due dates and released by EDD policy.

Step 2: First Come First Serve (FCFS) for transportation. Within the planning period, when the completed orders are transferred to airport, they are allocated to the available flights based on FCFS rule. Orders are allocated to the flight with the earliest departure time until the capacity of the flight is fulfilled or the flight departs. Then, the orders are allocated to the succeeding flight. Orders are allowed to split and allocated to more than one flight. In some cases, the remaining capacity of current flight is not enough for the selected order. The order is then split into two partial orders: the first partial order fulfills the remaining capacity of current flight and the second partial order is allocated to the succeeding flight.

Considering single machine assembly, two CESC synchronization problems are studied, taking into account the Type 1 and Type 3 air transportation problem respectively, as only Type 1 and Type 3 air transportation problem can be solved optimally by available mathematical programming software. Test problems of the two CESC synchronization problems are generated following the experiment design presented in section 3.2.2 and section 3.4.2 respectively.

4.4.1 Results and Analysis for Synchronizing Type 1 Air Transportation and the Single Machine Assembly

It is mentioned in section 4.2.1 that LPT rule leads to optimal solution in single machine scheduling with only earliness penalties. However, step 3 of the forward/backward scheduling algorithm may change the job sequence that is determined by the LPT rule. Thus, the performance of LPT rule needs to be compared with other sequence rules for validation. To evaluate the performance of the LPT rule

applied in Step 2 of both backward scheduling algorithm and forward scheduling algorithm, the LPT rule is compared with the EDD rule, which is a commonly applied rule in practice, where due date refers to the final delivery due date. Table 4.3 summarizes the average waiting time values of all the tested problems with LPT and EDD sequence rules applied in Step 2 for both backward scheduling algorithm and forward scheduling algorithm. The results are presented in terms of average values of the five instances for each tested problem.

Table 4.3 Performance of the proposed backward / forward scheduling algorithm in synchronizing Type 1 air transportation and the single machine assembly

Problem	Average waiting time				Reduction of (1) to (2) (%)	
	Backward scheduling algorithm (1)		Forward scheduling algorithm (2)			
	LPT	EDD	LPT	EDD	LPT	EDD
10j3f	2.21	2.74	5.47	5.99	59.60	54.26
20j4f	1.25	1.61	6.11	6.42	79.54	74.92
30j6f	0.99	1.38	3.92	4.31	74.74	67.98
40j8f	0.72	0.96	4.61	4.85	84.38	80.21
50j10f	0.61	0.86	3.52	3.76	82.67	77.13
60j12f	0.41	0.55	4.66	4.79	91.20	88.52
70j14f	0.39	0.53	4.96	5.10	92.14	89.61
80j16f	0.33	0.46	4.45	4.58	92.58	89.96
90j18f	0.31	0.40	4.50	4.59	93.11	91.29
100j20f	0.26	0.34	4.53	4.64	94.26	92.67

Table 4.3 shows that LPT rule provides smaller average waiting time compared to EDD rule for all the tested problems, for both backward scheduling algorithm and forward scheduling algorithm. It is also observed that backward scheduling algorithm can provide significantly smaller average waiting time than the forward scheduling algorithm. Backward scheduling algorithm yields 60% to 94% of average waiting time reduction over forward scheduling algorithm for different sized problems when LPT rule is applied. When EDD rule is applied in step 2 of the two proposed algorithms, backward scheduling algorithm yields 54% to 93% average waiting time reduction over forward scheduling algorithm for different sized problems. It is also interesting to note that for the backward scheduling algorithm, the average waiting time decreases as the problem size increases. In the forward scheduling algorithm, the values of average waiting time do not have much difference for all tested problems. It indicates that the backward scheduling algorithm provides better solutions in comparison with forward scheduling algorithm for large sized problems which involve complex scenarios with large number of orders and flights.

Since the backward scheduling algorithm is found to be better than the forward scheduling algorithm from the above comparison, the SSM methodology including the backward scheduling algorithm for assembly scheduling and the ILP model of air transportation will be compared with the existing industry practice heuristic (EDD+FCFS). The contribution of the backward scheduling algorithm is measured by the reduction of average waiting time and to the ILP model by total transportation cost reduction over EDD+FCFS heuristic respectively. The comparison results are reported in Table 4.4.

Table 4.4 shows that the proposed SSM methodology can significantly reduce average waiting time by 52% to 93% compared to the EDD+FCFS heuristic for different sized problems. It indicates the high efficiency of SSM in synchronizing assembly and air transportation in CESC compared to the EDD+FCFS heuristic. The proposed SSM methodology also yields 2.1% to 8.7% of delivery cost reduction per day (24 hour planning period) over the EDD+FCFS heuristic. Therefore, the comparison suggests that SSM can achieve reduced transportation cost and significantly decreased average waiting time compared to current industry practice heuristic.

Table 4.4 Performance of SSM and EDD+FCFS heuristic in synchronizing Type 1 air transportation and the single machine assembly

Problem	Average waiting time			Total transportation cost		
	Backward scheduling algorithm (1)	EDD+FCFS (2)	Reduction of (1) to (2) (%)	ILP (3)	EDD+FCFS (4)	Reduction of (3) to (4) (%)
10j3f	2.21	4.69	52.85	101445	104113	2.56
20j4f	1.25	5.24	76.10	172884	180888	4.42
30j6f	0.99	4.00	75.16	328715	355043	7.42
40j8f	0.72	4.28	83.23	361666	371262	2.58
50j10f	0.61	3.30	81.41	452978	462459	2.05
60j12f	0.41	3.98	89.64	509016	533568	4.60
70j14f	0.39	3.70	89.35	583146	638773	8.71
80j16f	0.33	3.91	91.66	687694	716792	4.06
90j18f	0.31	3.99	92.18	771986	804659	4.06
100j20f	0.26	3.86	93.24	844174	887688	4.90

4.4.2 Results and Analysis for Synchronizing Type 3 Air Transportation and the Single Machine Assembly

Consider synchronizing Type 3 air transportation and single machine assembly. Table 4.5 summarizes the results of average waiting time for all the tested problems by applying both backward scheduling algorithm and forward scheduling algorithm. For each algorithm, the performances are also compared when LPT and EDD sequence rules are employed in Step 2 respectively. The results are presented in terms of average values of the five instances for each tested problem. Table 4.5 shows that LPT rule leads to smaller average waiting time over EDD rule in all the tested problems, for both backward scheduling algorithm and forward scheduling algorithm.

Table 4.5 Performance of the proposed backward / forward scheduling algorithm in synchronizing Type 3 air transportation and the single machine assembly

Problem	Average waiting time				Reduction of (1) to (2) (%)	
	Backward scheduling algorithm (1)		Forward scheduling algorithm (2)			
	LPT	EDD	LPT	EDD	LPT	EDD
20j4f2d	3.10	3.55	5.05	5.50	38.61	35.45
30j6f2d	2.29	2.64	4.07	4.43	43.73	40.41
40j8f3d	2.85	3.11	4.50	4.76	36.67	34.66
50j10f3d	2.07	2.27	4.44	4.64	53.38	51.08
60j12f3d	2.13	2.30	4.43	4.60	51.92	50.00
70j14f4d	2.15	2.31	3.30	3.46	34.85	33.24
80j16f4d	2.39	2.52	3.91	4.04	38.87	37.62
90j18f4d	1.69	1.81	3.91	4.03	56.78	55.09
100j20f5d	2.50	2.60	4.90	5.00	48.98	48.00

It is also observed that the proposed backward scheduling algorithm can provide significantly smaller average waiting time than forward scheduling algorithm. Backward scheduling algorithm yields 34.8% to 56.7% average waiting time reduction over forward scheduling algorithm for different sized problems when LPT rule is applied. When EDD rule is applied in step 2 of the two proposed algorithms, backward scheduling algorithm yields 33.2% to 55.1% average waiting time reduction over forward scheduling algorithm for different sized problems. In addition, the average waiting time decreases as the problem size increases for backward scheduling algorithm. For forward scheduling algorithm, the average waiting time does not have much difference for all tested problems. It indicates that the proposed backward scheduling algorithm provides better solutions in comparison with forward scheduling algorithm for large sized problems which involve complex scenarios with large number of orders and flights.

To evaluate the contribution of SSM in synchronizing Type 3 air transportation and the single machine assembly, SSM is compared with the EDD + FCFS heuristic. The contribution of the backward scheduling algorithm is measured by the reduction of average waiting time and the ILP model by total cost reduction over EDD+FCFS heuristic. The comparison results are reported in Table 4.6. It is observed that SSM can obtain significant reduction of average waiting time by 52.4% to 87.3% over the EDD+FCFS heuristic for different sized problems. It is also observed that there are backlog orders when applying EDD+FCFS heuristic, while applying SSM can deliver all orders to customers. The backlog is resulted from the competition of orders with different destinations in assembly. Suppose all the flights' departure times to the

destination g are early, while the orders to this destination have late due dates. These orders are intended to be released late in assembly following the EDD rule. It is likely that some of the orders will miss the last flight to destination g , resulting in backlog. It is observed from Table 4.6 that EDD+FCFS heuristic always leads to 4.5% to 11.2% backlog, while SSM can deliver all the orders to customers within the planning period for various tested problems.

Table 4.6 Performance of SSM and EDD+FCFS heuristic in synchronizing Type 3 air transportation and the single machine assembly

Problem	Average waiting time			Backlog of EDD+FCFS	
	Backward scheduling algorithm (1)	EDD+FCFS (2)	Reduction of (1) to (2) (%)	Quantity	Backlog Percentage (%)
20j4f2d	3.10	5.32	41.75	115	4.48
30j6f2d	2.29	5.04	54.66	258	7.28
40j8f3d	2.85	5.53	48.39	422	8.60
50j10f3d	2.07	5.56	62.76	544	8.61
60j12f3d	2.13	5.35	60.13	417	5.60
70j14f4d	2.15	4.76	54.81	939	11.20
80j16f4d	2.39	5.08	52.87	615	6.20
90j18f4d	1.69	5.38	68.54	805	7.40
100j20f5d	2.50	6.03	58.46	1055	8.50

Total transportation costs are presented in the left half of the Table 4.7. It is observed that the proposed methodology can only yield minor delivery cost reduction per day (24 hr planning period) over the EDD+FCFS heuristic. This is due to the fact that the costs of the backlog orders are not included in the total cost of EDD+FCFS heuristic.

Identifying the cost reduction of ILP over EDD+FCFS should consider the cost of the backlog orders. The backlog orders will have higher delivery cost compared to the delivery cost of non-backlog orders. The higher cost accounts for the contracted penalty, lost of good will or hiring a chartered flight to deliver the orders within the planning period with much higher costs.

Table 4.7 Cost reduction of the ILP model to EDD+FCFS heuristic

problem	Total transportation cost		Cost reduction of ILP to EDD+FCFS		
	ILP	EDD+FCFS	$\theta = 1$ (%)	$\theta = 2$ (%)	$\theta = 3$ (%)
20j4f2d	282289	281960	4.37	8.48	12.24
30j6f2d	386330	394165	9.12	15.29	20.67
40j8f3d	566270	560718	7.70	15.00	21.24
50j10f3d	750243	762855	10.12	17.24	23.31
60j12f3d	882096	893263	6.78	11.72	16.17
70j14f4d	1007484	974018	8.15	17.40	24.96
80j16f4d	1267648	1306766	9.01	14.32	19.05
90j18f4d	1335490	1374271	10.01	16.21	21.61
100j20f5d	1668693	1633667	6.54	13.86	20.12

Next, A formula is developed to calculate the cost reduction of the ILP model to the EDD+FCFS heuristic, where the costs of backlog orders are included in the total transportation costs of EDD+FCFS. Let TC_1 be the total cost of non-backlog orders generated by the EDD+FCFS heuristic, TC_2 be the total cost generated by the ILP model, and ω be the backlog percentage. Suppose the average delivery cost of per unit backlogged product is θ times of the average delivery cost of per unit non-backlog

product. Then, the delivery cost for the backlog orders generated by EDD+FCFS

heuristic is $\frac{TC_1}{1-\omega}\omega\theta$. Thus, the total delivery cost resulting from the EDD+FCFS

heuristic for non-backlog orders and backlog orders is $TC_1 + \frac{TC_1}{1-\omega}\omega\theta$. Therefore, the

actual cost reduction of ILP to EDD+FCFS is calculated by the following formula:

$$1 - \frac{TC_2}{TC_1 + \frac{TC_1}{1-\omega}\omega\theta} \quad (4.1)$$

The actual cost reduction of the ILP model over the EDD+FCFS heuristic is presented in Table 4.7 for different value of θ . It is observed that even for $\theta=1$, i.e., no additional penalty imposed on the backlog orders, the cost reduction of ILP over EDD+FCFS is significant. Clearly, with the increasing value of θ , the cost reduction will be more significant.

Overall, the results of above comparisons indicate that the proposed SSM methodology can achieve more accurate delivery with lower cost and reduced waiting time before transportation compared to current industry practice heuristic. The proposed methodology also guarantees no backlog order, which is a significant feature compared to the EDD+FCFS heuristic.

4.5 Managerial Implications

The presented methodology of SSM and the computational results provide several management implications.

1. The transportation allocation model benefits transportation capacity planning.

Once the orders to be transported in a planning period are decided, the transportation allocation can be obtained by the ILP model. This is helpful for daily transportation capacity planning. The worst situation of no flight capacity to transport finished orders can be avoided. It is due to the fact that the transportation capacity is always larger than the order quantity to be transported, which are the inputs to the ILP model. Then, the unused flight capacity can be handled in advance. The allocated special capacity of each flight could be negotiated with airlines in advance and possibly obtained at a lower cost.

2. Total cost reduction. Delivery earliness and tardiness penalties for each order can be assigned according to the priority of the customer. Optimal delivery is guaranteed by the ILP model. In other words, the total cost which consists of transportation cost and delivery penalty cost is minimized. Higher delivery penalty cost and higher transportation cost per unit product generally lead to larger gap between the results of the ILP model and the result of the EDD+FCFS heuristic. Another aspect of cost saving is the reduction of average waiting time by the backward scheduling algorithm compared to EDD+FCFS heuristic. The temporary finished goods inventory always leads to storage cost, insurance cost, capital opportunity cost, etc.

3. *Evaluation of alternate planning and scheduling decisions*: The methodology can be used to evaluate various decisions, which have different parameter values. The decision alternatives can be consideration of different flights, schedules, acceptance of orders, etc. The decision leading to lower cost can be selected for execution by the presented methodology. This helps the planners for making decisions in volatile situations such as acceptance of an important emergency order, flight cancellation, order cancellation and customer priority changing.

SSM can be applied widely in consumer electronics industry, as well as other industries with the strategy of MTO and air transportation is exploited in distribution. The manufacturer can book flight capacity from the cargo airlines directly, or via agent. Also, it is widely applied in consumer electronics industry that the manufacturer outsources the transportation and distribution of the orders to a 3PL. The manufacturer and the 3PL may enter some sort of partnership or contract agreement that the 3PL will allocate some amount of cargo space in each flight for the manufacturer to use. In this case, the manufacturer will have a control over some cargo space of the flights. In all these situations, SSM can be applied.

4.6 Summary

This chapter formulates the single machine assembly scheduling problem. Two solution methods, namely forward scheduling algorithm and backward scheduling algorithm are presented. Performances of these two algorithms are evaluated quantitatively through numerical experiments. The proposed SSM methodology, which consists of the ILP

model for the air transportation problem and backward scheduling algorithm for the single machine assembly scheduling problem, is also compared with industry practice heuristic. Computational results suggest that the proposed SSM methodology can achieve more accurate delivery with lower delivery cost and reduced waiting time before transportation compared to current industry practice heuristic. SSM also guarantees no backlog order, which is a significant feature compared to the industry practice heuristic. Finally, the managerial implications of the presented methodology are summarized.

As single machine is the simplest situation of assembly, the more complicated situation of parallel machine assembly is investigated in the next chapter. In addition, the single machine assembly scheduling problem considering machine breakdowns is investigated in chapter 6.

Annexure: Demonstration of Synchronized Scheduling Methodology by A Numerical Example

In this section, a simple numerical example is presented to demonstrate the proposed methodology of SSM step by step. Backward scheduling algorithm is applied for assembly scheduling.

Inputs:

- a) Problem size:

PR=80

Number of flights = 3

Number of orders = 10

Planning period = 24 hours

- b) Order details as shown in Exhibit 1.

Exhibit 1 Input of order details to SSM

Order	Quantity (Q_i)	Due date (dd_i)	Priority (P_i)	Earliness penalty ($d\alpha_i$)	Tardiness penalty ($d\beta_i$)
1	131	13.1	2	5	6
2	76	10.7	2	3	7
3	96	11.4	1	6	7
4	132	6.4	3	3	8
5	119	20.3	3	3	7
6	191	10.5	3	4	5
7	91	9.1	2	5	8
8	178	7.4	2	5	6
9	106	9.5	1	3	5
10	54	20.8	3	4	7

c) Flight details as shown in Exhibit 2.

Exhibit 2 Input of flight details to SSM

Flight	Departure time (D_f)	Arrival time (A_f)	Normal area		Special area	
			Capacity ($NCap_f$)	Unit cost (NC_f)	Capacity ($SCap_f$)	Unit cost (SC_f)
1	5	7	377	13	83	12
2	10	12	422	8	103	14
3	15	17	369	13	118	10

Outputs:

a) The total transportation cost is 25318. The optimal order allocation to existing flights that are summarized in Exhibit 3

Exhibit 3 Optimal order allocation

Order	Flight 1		Flight 2		Flight 3		Total allocated quantity
	Normal capacity	Special capacity	Normal capacity	Special capacity	Normal capacity	Special capacity	
1	0	0	0	0	131	0	131
2	0	0	76	0	0	0	76
3	0	0	96	0	0	0	96
4	132	0	0	0	0	0	132
5	0	0	0	0	119	0	119
6	0	0	191	0	0	0	191
7	67	23	1	0	0	0	91
8	178	0	0	0	0	0	178
9	0	0	36	0	65	5	106
10	0	0	0	0	54	0	54

The details of capacity utilization for each flight can also be derived from this Exhibit.

Chapter 4: The Single Machine Assembly Scheduling Problem

b) Flight wise order sequence. As shown in Exhibit 4, only LPT and weighted priority rules are applied in this example.

Exhibit 4 Flight wise order sequence

Flight	Orders allocated to normal capacity	Orders allocated to special capacity	Flight wise order sequence	
			LPT ^a	Weighted priority ^b
1	4, 7(67)* ^c , 8	7(23)*	8, 4, 7(90)*	4, 8, 7(90)* ^d
2	2, 3, 6, 7(1)*, 9(36)*		7(1)*, 6, 3, 2, 9(36)*	7(1)*, 6, 2, 3, 9(36)*
3	1, 5, 9(65)*, 10	9(5)*	9(70)*, 1, 5, 10	9(70)*, 5, 1, 10

- a. Sequence by LPT with the process of assembly batching of split orders (step 3 of backward scheduling algorithm). In the following illustration, rule name means the rule with the process of assembly batching of split orders applied.
- b. Each order's weighted priority equals to its priority multiplied by its quantity. Orders are released from higher weighted priority to the lower one.
- c. * means a proportion of an order. The number in the bracket is the quantity of the proportion of the order that is allocated into the corresponding flight.
- d. Regardless of allocated to normal or special capacity, the quantity of order 7 that is allocated to flight 1 is 90.

c) Determined release time as shown in Exhibit 5.

Exhibit 5 Determined release time

Order	Determined release time	
	LPT	Weighted priority
1	11.20	12.69
2	8.60	7.40
3	7.40	8.35
4	2.23	0
5	12.84	11.20
6	5.01	5.01
7(90)	3.88	3.88
7(1)	5.00	5.00
8	0	1.65
9(36)	9.55	9.55
9(70)	10.32	10.32
10	14.33	14.33

Chapter 4: The Single Machine Assembly Scheduling Problem

d) The waiting time, earliness and tardiness as shown in Exhibit 6.

Exhibit 6 The waiting time, earliness and tardiness

Rules	Order	Priority	Delivery time	Due date	Earliness	Tardiness	Waiting time	Weighted waiting time ^a
By LPT	8	2	7	7.4	0.4		2.78	5.55
	4	3	7	6.4		0.6	1.13	3.38
	7	2	7	9.1	2.1		0.00	0.00
	7	2	12	9.1		2.9	4.99	9.97
	6	3	12	10.5		1.5	2.60	7.80
	3	1	12	11.4		0.6	1.40	1.40
	2	2	12	10.7		1.3	0.45	0.90
	9	1	12	9.5		2.5	0.00	0.00
	9	1	17	9.5		7.5	3.80	3.80
	1	2	17	13.1		3.9	2.16	4.33
	5	3	17	20.3	3.3		0.68	2.03
	10	3	17	20.8	3.8		0.00	0.00
Total					9.6	20.8	19.97	39.15
Average							1.67	1.57
By weighted priority	4	3	7	6.4		0.6	3.35	10.05
	8	2	7	7.4	0.4		1.13	2.25
	7	2	7	9.1	2.1		0.00	0.00
	7	2	12	9.1		2.9	4.99	9.97
	6	3	12	10.5		1.5	2.60	7.80
	2	2	12	10.7		1.3	1.65	3.30
	3	1	12	11		0.6	0.45	0.45
	9	1	12	10		2.5	0.00	0.00
	9	1	17	9.5		7.5	3.80	3.80
	5	3	17	20.3	3.3		2.31	6.94
	1	2	17	13.1		3.9	0.68	1.35
	10	3	17	20.8	3.8		0.00	0.00
Total					9.6	20.8	20.95	45.91
Average							1.75	1.84

^a. Weighted waiting time = Waiting time * Priority

e) Performance measure as shown in Exhibit 7.

Exhibit 7 Performance measure

Sequence Rule	Average waiting time	Weighted Average waiting time ^a
LPT	1.67 hour	1.57 hour
Weighted priority	1.75 hour	1.84 hour

$$^a \cdot \text{Weighted average waiting time} = \frac{\sum_i (\text{Waiting time}(i) * \text{Priority}(i))}{\sum_i \text{Priority}(i)}$$

Chapter 5: The Parallel Machine Assembly Scheduling

Problem

The work done in previous chapter is extended by considering parallel assembly lines in assembly flow shop. Each assembly line is treated as a single machine. Thus, this chapter investigates the parallel machine assembly scheduling problem, which is demonstrated in Figure 5.1. Different variations of this problem are studied subject to different assumptions of split /non-split processing and job dependent/independent penalty weights.

Figure 5.1 The assembly flow shop

5.1 Introduction and Problem Formulation

Without loss of generality, the machines in assembly are assumed to be unrelated parallel machines. There are N independent jobs $J = \{J_1, \dots, J_N\}$ to be scheduled on M unrelated machines. A processing time p_{jm} is associated with job j on each machine m .

As mentioned at the beginning of section 4.2, the assembly due dates are distinct. Therefore, this assembly scheduling problem is a distinct due date parallel machine scheduling problem to minimize total earliness and tardiness. Meanwhile, jobs are assigned a common assembly due date if they are allocated to the same flight in transportation. These two features of assembly due dates greatly facilitate the problem formulation and analysis, which will be shown in detail in the following sections.

As mentioned in chapter 3, one important constraint of the transportation ILP model is the production rate of assembly, i.e., enough assembly capacity to supply the products to be allocated in air transportation. In other words, each job should be completed before its due date of assembly. Thus, only earliness penalties occur. The problem addressed in this chapter is a Parallel machine Earliness Non-common Due date Scheduling Problem (PENDSP). Here, the capacity is the accumulated capacity of all the parallel machines, which also can be denoted by Figure 3.2. Each job is assigned a weight w_j for its early penalty. Therefore, the objective of PENDSP is to minimize total weighted earliness without tardiness, that is,

$$\text{Min } Z(\sigma) = \sum w_j (d_j - C_j) \quad (5.1)$$

σ is a schedule, C_j and d_j are the assembly completion time and the assembly due date of job j . w_j denotes the earliness weight of job j .

Generally, there are two types of weights: job independent weight and job dependent weight (Baker and Scudder, 1990). For job independent weights, $w_j = w$ for all jobs. On

the other hand, job dependent weights indicate that w_j differ for different jobs. Thus, the overall problem can be classified into two problems according to the two types of weights: (1) PENDSP with independent weights, (2) PENDSP with dependent weights. Hereafter, independent weights refer to job independent weights and dependent weights refer to job dependent weights. The study in this chapter shows that job independent weights simplify PENDSP, while PENDSP with job dependent weights is NP-complete.

Another important assumption relies on the permission of job being split arbitrarily and processed on more than one machine, which is represented by the term split processing. Thus, four sub-problems are studied for PENDSP: PENDSP with independent weights and split processing, PENDSP with independent weights and non-split processing, PENDSP with dependent weights and split processing, PENDSP with dependent weights and non-split processing. Particularly, if a job can be split and processed on more than one machine (the first and third sub-problem), it can be proved that the corresponding two sub-problems can be solved using the proposed backward scheduling algorithm, which is presented in section 4.2.2. Meanwhile, PENDSP with independent weights and non-split processing is formulated as a transportation problem, while PENDSP with dependent weights and non-split processing is proved to be NP-complete and SA based algorithms are developed. In addition, the assumptions presented in section 4.1 are also applicable to these four sub-problems.

5.2 PENDSP with Independent Weights

In practice, the weight of each job is chosen according to many considerations. In order to minimize the sum of total deviation of the job completion times from the due dates,

the weight of each job is set to be job independent, i.e., $w_j = w$ for all jobs. In many real applications, the weights are set equal to one for all jobs (Sun and Wang, 2003). Therefore, subject to $w_j=1$, the problem can be formulated as:

$$\text{Min } Z(\sigma) = \sum (d_j - C_j) \quad (5.2)$$

Within this problem, the two sub-problems with the assumptions of split processing and non-split processing are addressed in the following two sections respectively.

5.2.1 PENDSP with Independent Weights and Split Processing

Assuming that one job can be split arbitrarily into two or more partial jobs and be processed on more than one machine, the following theorem can be derived:

Theorem 5.1: PENDSP with independent weights and split processing can be solved by applying the proposed backward scheduling algorithm.

Proof. Suppose there are M parallel machines, each machine's capacity is $\lambda_m PR$, where PR is the overall production rate of all the machines and λ_m is the proportion of machine m 's production rate to PR . As mentioned in section 4.1, production rate is taken as the assembly capacity in this thesis. Each job is split into M partial jobs. Each partial job, which is denoted by J_{jm} , has the quantity of $\lambda_m Q_j$, where Q_j is the quantity of the job j before split. J_j and its partial jobs have identical assembly due date that is $d_{jm}=d_j$. Then, assign each partial job J_{jm} to the corresponding machine m with production rate of $\lambda_m PR$. Focusing on the scheduling problem on each machine, it is a

single machine scheduling problem with earliness penalties, which is denoted by **SMP_m**. By this manner, the problem of PENDSP with independent weights and split processing is decomposed.

As illustrated in chapter 4, taking the assembly flow shop as a single machine, the assembly scheduling problem can be solved by the proposed backward scheduling algorithm. This problem is denoted by **PO**. The processing time of job j is $p_j = \frac{Q_j}{PR}$.

For **SMP_m**, the processing time of job J_{jm} on machine m is $p_{jm} = \frac{\lambda_m Q_j}{\lambda_m PR} = \frac{Q_j}{PR} = p_j$. It

indicates that for a job j , all its partial jobs have identical processing time on the machine it is assigned to.

For **PO** and **SMP_m** with $m=1, 2, \dots, M$, the input parameters are the job processing times and the due dates. As $p_{jm} = p_j$ and $d_{jm}=d_j$, furthermore, as **PO** can be solved by the backward scheduling algorithm, **SMP_m** can also be solved by the backward scheduling algorithm. Therefore, PENDSP with independent weights and split processing can be solved by the backward scheduling algorithm. $\square$

5.2.2 PENDSP with Independent Weights and Non-split Processing

PENDSP with independent weights and non-split processing is more complex compared to one with split processing. In this section, a methodology is presented to solve this problem by decomposing the overall problem into sub-problems based on the

feature of distinct common due date. Then, each sub-problem is formulated as a transportation problem. The decomposition procedure is illustrated below.

5.2.2.1 Decompose PENDSP with Independent Weights and Non-split Processing

First of all, PENDSP with non-split processing addressed in this research does not consider the special case that tardiness may occur. Although the overall capacity is assumed to be sufficient, for some special case, the distinct capacity of each machine may not be sufficient for some jobs. For instance, given a schedule σ , the last job on a machine m can not be completed on time due to the insufficient capacity of machine m . However, there are spare capacities on the remaining machines. The capacity of each remaining machine is also not sufficient for the job, but the accumulated capacity of all machines is sufficient. The problem is that the job can not be split and processed separately on each machine at the same time. Thus, tardiness occurs. The example is detailed in Figure 5.2.

Figure 5.2 A special case of PENDSP with non-split processing

In the given example, there are three machines. To simplify the illustration, each machine is assumed to have identical capacity. The job assigned to each machine is denoted by a rectangle. Each job's due date is set to d . The last job on machine 3 is denoted by LJ in the corresponding rectangle. It is observed that completion time of this job is later than its due date. If all jobs' schedule is kept unchanged except the job LJ , it is observed that LJ exceeds any machine's remaining capacity. In contrast, the cumulative spare capacity of all the three machines is sufficient for processing job LJ .

This special case is an extreme case of PENDSP with non-split processing. Such a case is not considered in this research. Assembly capacity is assumed to be large enough that this situation is avoided. Also, large assembly capacity enables us to find feasible solution for PENDSP with independent weights and non-split processing in polynomial time.

To determine that whether there is a feasible solution for PENDSP with independent weight and non-split processing, the computational complexity depends on assembly capacity. First, an extreme case with limited assembly capacity is investigated. There are two identical machines in assembly flow shop, and a number of jobs with distinct processing time p_i and identical due date d . The total processing time for all jobs is $2d$. In order to find a feasible solution for this problem that there is no tardy job, jobs must be allocated to the two machines such that total processing time of all jobs allocated to each machine is d . It is evident that this is a partition problem, which is presented in section 3.3.3. Since partition problem is NP-complete, this problem is also NP-complete.

Second, a special case with sufficient assembly capacity is also studied. It is identical to the above case except with larger assembly capacity that the total processing time for all jobs equals d . It appears that feasible solution can be easily obtained by assigning all jobs to the two machines arbitrarily. Thus, it is not NP-complete to find a feasible solution for this special case.

Generally, in this thesis, it is assumed that there is sufficient assembly capacity, so that feasible solution for PENDSP with independent weights and non-split processing can be found in polynomial time. This research aims to provide a formulation and solution methodology for the general PENDSP with independent weights and non-split processing subject to no tardiness.

According to section 2.3.2, the common due date earliness tardiness scheduling problem has attracted much attention in the past two decades. In order to benefit from the results of these existing intensive studies, the overall parallel machine scheduling problem is decomposed into sub-problems. For each flight, there are several jobs allocated to it, which share a common due date. Only considering these jobs to be scheduled in assembly, it is a sub-problem of PENDSP. Each sub-problem is a common due date earliness tardiness scheduling problem, while the common due date for each sub-problem is different. The number of sub-problems equals to the number of different assembly due dates, or equals to the number of flights in air transportation. By solving each sub-problem, the release time of each job is obtained and the total earliness can be calculated. Among all the sub-problems, the one with largest common due date is solved first. To illustrate the decomposition procedure, considering a CESC

synchronization problem with five orders, two flights and two identical machines in assembly. The order allocation in air transportation and assembly processing times are reported in Table 5.1.

Table 5.1 Order processing times and air transportation allocation

Order	Processing time	Allocated to flight
1	2	1
2	2	1
3	1	1
4	1.5	2
5	3	2

The departure times are 5th and 10th hour for flights 1 and 2 respectively. The assembly problem is assumed to be a PENDSP with independent weights and non-split processing. Then, this problem is decomposed into two sub-problems. Jobs 1, 2, 3 belong to the first sub-problem. The common due date is 5. Jobs 4 and 5 belong to the second sub-problem with the common due date of 10. Generally, each sub-problem is an unrelated parallel machine earliness scheduling problem with common due date. As demonstrated at the beginning of section 4.2, each common due date is unrestricted.

Unlike the common due date parallel machine scheduling problem addressed in literature, one more parameter namely unavailable time U_m for each machine is introduced in the sub-problem of PENDSP with independent weights and non-split processing. U_m denotes the time point that the machine becomes unavailable in a sub-problem. It is common that the assembly capacity may not be sufficient for some

flights' allocation if the allocated orders are processed between the departure times of the preceding flight and current flight. As tardiness is not supposed to occur, some orders allocated to current flight must start processing in assembly before the departure time of the preceding flight to satisfy the capacity requirements of the orders allocated to current flight. Suppose the R_{lm} is the release time of the first job on machine m in sub-problem f , which is denoted by SP_f . C_{nm} is the completion time of the last job on machine m in SP_f . Therefore, jobs in SP_f will occupy machine m 's processing time length from R_{lm} to C_{nm} . For sub-problem SP_{f-1} which is prior to problem f , jobs within SP_{f-1} can not be processed on machine m after time R_{lm} . Thus, R_{lm} is the unavailable time of machine m in SP_{f-1} . Figure 5.3 details the concept of machine unavailable time.

Figure 5.3 Demonstration of machine unavailable time

Let SP_F be the last sub-problem that has the largest common due date, which is denoted by d_F in Figure 5.3. The schedule of jobs on each machine is obtained by solving SP_F . The jobs assigned to each machine are denoted by the rectangle in Figure 5.3. The time corresponding to the start point of each rectangle denotes the release time of the first job on each machine. The release time of the first job on each machine in SP_F , which is denoted by U_1 and U_2 respectively in the figure, can be taken as the unavailable time of each machine in SP_{F-1} . It is observed that the unavailable time can be either later or

earlier than the common due date of SP_{F-1} , which is denoted by d_{F-1} in the above figure. The machine unavailable times in a sub-problem are determined by solving its adjacent sub-problem with larger common due date. This provides another reason to solve the sub-problem with the largest common due date first.

5.2.2.2 Formulation of the Sub-Problems

Kubiak et al. (1990) studied unrelated parallel machine scheduling problem to minimize total earliness and tardiness penalties with job independent weights. The problem is formulated as a transportation problem. As mentioned in section 3.2.3, a transportation problem is specified by three elements: a set of supply points, a set of demand points, cost of shipping each unit from a supply point to a demand point. However, the problem addressed by Kubiak et al. (1990) generally has two differences from the sub-problems of PENDSP with independent weights and non-split processing. For the sub-problems of PENDSP with independent weights and non-split processing, the objective is to minimize total earliness without tardiness. In addition, a parameter of machine unavailable time U_m is introduced. The work of Kubiak et al. (1990) is extended in this research, so that the two differences are taken into account in formulating each sub-problem as a transportation problem.

Before giving the formulations of the sub-problems, the following notation is defined:

p_{jm} Processing time of job j on machine m .

d The common due date

U_m The unavailable time of machine m

The decision variable is defined as follows:

χ_{jmk} 1 if job j is processed on machine m , and there are $k-1$ jobs processed before it, 0 otherwise.

It is possible that all N jobs can be assigned to one machine with large capacity. Therefore, there are a maximum of N positions for each job to be assigned on each machine. Furthermore, the total possible positions are NM for a job to be assigned among the M parallel machines.

Particularly, SP_f can be viewed as a transportation problem. The N jobs are equivalent to N supply points. NM possible assignment positions for each job are equivalent to NM demand points. For a job j , if $\chi_{jmk}=1$, then the waiting time of job j is the total processing time of the jobs released after job j on machine m . It can be taken as the shipping cost from supply point j to demand point mk . As a job can only be assigned to one position, it can be viewed that each supply point can only supply 1. Each position only needs one job to be assigned to, which denotes that the demand of each position is also 1. Since N is less than NM , this problem is an unbalanced transportation problem (Winston, 1994) with total supply less than total demand.

The ILP model is presented below:

$$\text{Min} \quad \sum_{j=1}^N \sum_{m=1}^M \sum_{k=1}^N [(k-1)p_{jm} + \text{Max}(d - U_m, 0)] \chi_{jmk} \quad (5.3)$$

Subject to:

$$\sum_{m=1}^M \sum_{k=1}^N \chi_{jmk} = 1 \quad j=1, 2, \dots, N. \quad (5.4)$$

$$\sum_{j=1}^N \chi_{jmk} \leq 1 \quad m=1, 2, \dots, M, k=1, 2, \dots, N. \quad (5.5)$$

The decision variable is χ_{jmk} . The objective is to minimize total independent earliness penalties, i.e., to minimize total waiting time. Waiting time of job j consists of sum of the processing time of jobs that are processed after it on machine m . In addition, if U_m is earlier than d , additional waiting time of $U_m - d$ should be added to waiting time of each job. It also can be viewed as additional fixed transportation cost for transporting from supply points to the particular demand points mk with $k = 1, 2, \dots, N$. Constraint (5.4) insures that each job is assigned to one position. Constraint (5.5) ensures that each position on each machine has at most one job assigned to it. As an unbalanced transportation problem can be solved in polynomial time (Winston, 1994), this ILP model is also solvable in polynomial time.

PENDSP with independent weights and non-split processing can be solved by solving the sub-problems in a sequence of solving the one with the largest common due date first. By solving the ILP model of each sub-problem, the assignments of jobs to machines and the job sequence on each machine are determined. Since all the jobs in each sub-problem share a common due date, and as all jobs are scheduled to be finished on or before the due date, there should be no inserted idle time in the job sequence on each machine for each sub-problem, in order to minimize total earliness. Therefore, by

solving each sub-problem, each job's release time is determined if the last job's finish time is determined in the job sequence on each machine. The last job should be finished at the common due date if due date is earlier than the machine unavailable time. Otherwise, the last job is finished at the machine unavailable time.

5.3 PENDSP with Dependent Weights

This section considers PENDSP with job dependent weights. This problem is more complex compared to PENDSP with job independent weights, which is studied in section 5.2. This problem is addressed in the situations of split processing and non-split processing.

In practice, the weights of jobs are chosen according to internal and external considerations and limitations that can affect the future of the company. Choosing appropriate weights in practice is difficult and time consuming (Alidaee and Dragan, 1997). An alternative approach is to use weights that are proportional to processing times of the jobs. Arkin and Roundy (1991) stated that typically, jobs that have longer processing times are larger jobs that have higher selling prices and, consequently, a higher priority. In many systems, both the processing time and weights of a job are nearly proportional to the job's dollar value, resulting in weights that are proportional to processing times.

The weight considered in this problem is the waiting cost for each unit of the job that completed earlier than its assembly due date. The waiting cost accounts for the finished goods inventory cost before transportation. For all the jobs, per unit per hour earliness

penalty cost is assumed to be identical in this section. Therefore, the job dependent weight, $w_j = Q_j * e = PR * p_j * e = \gamma * p_j$, where Q_j is the job quantity, p_j is the job processing time, e is per unit per hour earliness penalty cost and γ is the fixed coefficient denoting the relationship between job processing time and the penalty weight of the job. Generally, for all the jobs, per unit per hour earliness penalty cost is set to 1. Similar to PENDSP with independent weights and split processing that was studied in section 5.2.1, if any job can be split and processed on more than one machine, it is proved in this section that PENDSP with dependent weights and split processing can be solved by the proposed backward scheduling algorithm. In addition, PENDSP with dependent weights and non-split processing is formulated using a MIP model. It is proved that this problem is NP-complete and SA algorithms are developed for it.

5.3.1 PENDSP with Dependent Weights and Split Processing

Before introducing the complexity of PENDSP with dependent weights and split processing, the following lemma is proposed based on the Lemma 2 of Ahmed and Sundararaghavan (1990):

Lemma 5.1: For a parallel machine earliness scheduling problem with common due date d and job dependent penalty weight $w_j = \gamma * p_j$, where γ is the fixed coefficient denoting the relationship between job processing time and the penalty weight of the job. Let σ be a schedule such that the last job on each machine is completed at the common due date. Let σ' be any schedule obtained from σ by permutating the jobs on each machine in σ . Then, $Z(\sigma') = Z(\sigma)$.

Proof: Although only earliness is considered in this problem, Lemma 5.1 can be proved following the procedure in proving Lemma 2 in Ahmed and Sundararaghavan (1990). However, a different procedure is presented to prove Lemma 5.1 in this section.

As mentioned by Ahmed and Sundararaghavan (1990), any permutation of jobs on each machine within σ can be thought of as a series of two exchanges among jobs on each machine. Furthermore, a general two-exchange can be obtained from a series of adjacent two-exchanges. Therefore, the proof only needs to consider an adjacent two-exchange. For a given schedule σ , let $\{J_k, J_{k-1}, \dots, J_2, J_1\}$ denote the jobs scheduled on machine m . J_1 is completed at the common due date d , as indicated in Figure 5.4.

Figure 5.4 Schedule on machine m within σ

As shown in Figure 5.4, the waiting time of J_1 equals to zero. The waiting time of J_2 is p_1 . For J_3 , the waiting time is $p_1 + p_2$. Clearly, the waiting time of a job is the total processing time of the jobs released after this job. Following this logic:

$$\begin{aligned}
 Z(\sigma) &= \sum_{j=1}^k w_j (d - C_j) \\
 &= \sum_{j=1}^k \gamma^* p_j (d - C_j) \\
 &= \gamma [p_2 p_1 + (p_1 + p_2) p_3 + \dots + (p_1 + p_2 + \dots + p_{k-1}) p_k] \\
 &= \gamma [p_1 (p_2 + p_3 + \dots + p_k) + p_2 (p_3 + p_4 + \dots + p_k) + \dots + p_{k-1} p_k]
 \end{aligned}$$

It is observed that $Z(\sigma)$ consists of γ and the product of the processing times of any two jobs. The result is independent of job sequence. For example, if two adjacent jobs job J_i and J_{i+1} are exchanged, the result is:

$$\begin{aligned} Z(\sigma') &= \gamma[p_1(p_2 + p_3 + \dots + p_k) + p_2(p_3 + p_4 + \dots + p_k) + \dots \\ &\quad + p_{i+1}(p_i + p_{i+2} + \dots + p_k) + p_i(p_{i+2} + p_{i+3} + \dots + p_k) + \dots + p_{k-1}p_k] \\ &= \gamma[p_1(p_2 + p_3 + \dots + p_k) + p_2(p_3 + p_4 + \dots + p_k) + \dots \\ &\quad + p_i(p_{i+1} + p_{i+2} + \dots + p_k) + p_{i+1}(p_{i+2} + p_{i+3} + \dots + p_k) + \dots + p_{k-1}p_k] \end{aligned}$$

Therefore, $Z(\sigma') = Z(\sigma)$. □

The following theorem can be derived with the assistance of Lemma 5.1.

Theorem 5.2: PENDSP with dependent weights and split processing can be solved by the proposed backward scheduling algorithm.

Proof: The proof is similar to the procedure in proving Theorem 5.1. The main idea is to split the jobs according to the capacity of each machine. The procedures of job splitting and assigning the split jobs to machines are identical to the splitting and assigning procedures in section 5.2.1. Since total assembly capacity is sufficient, the capacity of each machine is also sufficient to process the jobs assigned to it. More detailed illustration can be referred to the proof section of Theorem 5.1. The difference between these two proofs is the sequence rule for sequencing the jobs on each machine. The objective of PENDSP with independent weights and split processing is to minimize total un-weighted earliness. Thus, on each machine, the jobs allocated to the same flight

are sequenced by the optimal sequence rule of LPT. The objective of PENDSP with dependent weights and split processing is to minimize total weighted earliness. According to Lemma 5.1, on each machine, jobs with the same assembly due date can be sequenced in an arbitrary order. Therefore, PENDSP with dependent weights and split processing can be solved by applying the proposed backward scheduling algorithm even without the LPT sequence rule employed in step 2 of the backward scheduling algorithm. $\square$

5.3.2 Formulation of PENDSP with Dependent Weights and Non-split Processing

As stated in section 5.1, the objective of PENDSP is to minimize total earliness without tardiness. The special case that there is insufficient capacity on separate machine is not considered, which is illustrated in section 5.2.2.1. Therefore, the objective of PENDSP with dependent weights and non-split processing is to schedule a set of jobs with distinct common due dates in assembly to minimize total earliness without tardiness.

Before presenting the MIP model, the following notation is defined:

p_{jm} processing time of job j on machine m

d_j assembly due date of job j (equals to the flight departure time that job j allocated to)

w_j earliness penalty weight of job j (per hour). As the per unit per hour earliness penalty for each job is set to 1, $w_j = Q_j$.

E_j earliness of job j

$$E_j = \text{Max}(0, d_j - C_j) \quad (5.6)$$

LN a large positive number

Also, the decision variables are defined:

C_j completion time of job j

PI_{jim} 1 if job j precedes job i immediately on machine m , 0 otherwise

V_{jm} 1 if job j is assigned to machine m , 0 otherwise

Note that in some cases, even though there is sufficient assembly capacity, the job is on hold for a while to ensure that the job is completed close to its due date. Thus, idle times are considered to be inserted in the job sequence on each machine.

The model is expressed as follows:

$$\text{Min } \sum_{j=1}^N w_j E_j \quad (5.7)$$

Subject to:

$$\sum_{m=1}^M V_{jm} = 1 \quad j = 1, \dots, N \quad (5.8)$$

$$\sum_{i=1}^{N+1} PI_{jim} = V_{jm} \quad i \neq j, \quad j = 0, 1, \dots, N, \quad m = 1, \dots, M \quad (5.9)$$

$$\sum_{i=0}^N PI_{ijm} = V_{jm} \quad i \neq j, \quad j = 1, \dots, N, N+1, \quad m = 1, \dots, M \quad (5.10)$$

$$\sum_{j=1}^{N+1} PI_{j0m} = 0 \quad m = 1, \dots, M \quad (5.11)$$

$$\sum_{j=0}^N PI_{(N+1)jm} = 0 \quad m = 1, \dots, M \quad (5.12)$$

$$C_i - C_j - PI_{jim}LN \geq p_{im} - LN \quad i \neq j, \quad i = 1, \dots, N, \quad j = 0, 1, \dots, N, \quad m = 1, \dots, M \quad (5.13)$$

$$C_j \leq d_j \quad j = 1, \dots, N, N+1 \quad (5.14)$$

$$PI_{jim}, V_{jm} \in \{0, 1\} \quad i, j = 1, \dots, N \quad (5.15)$$

The decision variables are V_{jm} , PI_{jim} and C_j . The objective of the above formulation is to minimize the sum of weighted earliness penalties of all N jobs. Note that two dummy jobs are created in scheduling in order to facilitate the representation of the immediate preceding and succeeding relationships of the jobs on each machine. Thus, $V_{om}=1$ and $V_{(N+1)m}=1$ for all m . The two dummy jobs have zero quantity and infinite due dates. They are assigned to be the first and the last job on each machine. Constraint (5.8) ensures that each job is assigned to one and only one machine. Constraint (5.9) and (5.11) ensures that on each machine, all the jobs should have an immediate preceding job except the first job. Constraint (5.10) and (5.12) ensures that on each machine, all the jobs should have an immediate succeeding job except the last job. Constraint (5.13) represents the completion time relationship between any two jobs on each machine. The completion time of a job is larger or equal to the sum of its processing time and the completion time of the immediate preceding job. This constraint indicates that there might be inserted idle times in the job sequence on each machine. Constraint (5.14) ensures that each job is completed before its due date, thus no tardiness will occur. This constraint is based on the condition of sufficient assembly capacity.

5.3.3 NP-Completeness of PENDSP with Dependent Weights and Non-split Processing

In this section, PENDSP with dependent weights and non-split processing is first decomposed into sub-problems. Then, the sub-problems are proved to be NP-complete. In this way, PENDSP with dependent weights and non-split processing is proved to be NP-complete. First of all, the problem is decomposed following the same procedure of decomposing PENDSP with independent weights and non-split processing, which was

illustrated in section 5.2.2.1. Each sub-problem is an unrelated parallel machine scheduling problem to minimize total earliness with unrestricted common due date. The last sub-problem with the largest common due date is selected and proved to be NP-complete in this section. This is due to the fact that each machine's unavailable time U_m is automatically set to the common due date of this sub-problem. There is no need to obtain the value of U_m by solving its adjacent sub-problem.

For the last sub-problem, the common due date is the departure time of the last flight. Since all orders are supposed to catch the allocated flight, there is no late job. On each machine, the completion time of the last job is scheduled to be the common due date d . In order to simplify the problem further, it is assumed that the machines are identical. Given a schedule, completion time of job j is denoted by C_j , the objective is to find a schedule σ that minimizes the total weighted earliness. That is: $\text{Min } Z(\sigma) = \sum w_j(d - C_j)$.

This problem is represented by **P**. All the jobs share a common due date, furthermore, all jobs are scheduled to be finished on or before the due date, there should be no inserted idle time in the job sequence on each machine for **P**, in order to minimize total earliness.

In this section, the NP-completeness is established for **P**, where the common due date is unrestricted and the earliness weight is proportional to the processing time of each job ($w_j = \gamma * p_j$). For the simplest **P** that only consider two parallel machines, the NP-completeness is established based on the partition problem (Garey and Johnson, 1979), which is known to be NP-complete.

Partition: Given positive integers n , B and a set of positive integers $G = \{x_1, x_2, \dots, x_n\}$

with $\sum_{i=1}^n x_i = 2B$, is there a subset $G_I \subseteq G$ such that $\sum_{x_i \in G_I} x_i = \sum_{x_i \in G - G_I} x_i = B$?

Based on any given instance of **Partition**, construct an instance of **P** by introducing the following quantities:

$$G' = \{J_1, J_2, \dots, J_n\}, M=2, \quad (5.16)$$

$$p_i = x_i, i=1, 2, \dots, n, \quad (5.17)$$

$$d = \sum_{i=1}^n p_i = 2B, \quad (5.18)$$

$$y = \gamma \left(B^2 - \sum_{i=1}^n x_i^2 / 2 \right) \quad (5.19)$$

There are two identical machines in **P**. In constraint (5.16), the job set G' is taken as the partition job set, which corresponds to the partition set G . Constraint (5.17) indicates that the processing time of each job in G' equals to the value of an element in G . Constraint (5.18) sets the unrestricted common due date to the sum of all job's processing times. Constraint (5.19) defines a variable y , which is used in the following illustration. The NP-completeness of **P** is established by showing that there exists a solution for this instance of **P** with $Z(\sigma) \leq y$ if and only if there exists a solution for the **Partition**. Thus, the NP-completeness is established based on the following two lemmas:

Lemma 5.2: If there exists a desired solution to **Partition**, then there exists a solution to **P** where $Z(\sigma) = y$.

Proof: If there is a desired solution of **Partition**, the two partitions are denoted by G_I and $G-G_I$. Then, G_I' and $G'-G_I'$ represent the two job sets corresponding to the two partitions. The two job sets are assigned to the two machines respectively. As any job sequence on each machine leads to the same penalty according to Lemma 5.1, jobs in G_I' and $G'-G_I'$ are sequenced in an arbitrary order. In order to minimize total earliness, there is no inserted idle time in the job sequence and the last job on each machine is completed at the due date d . Suppose $G_I' = \{ J_1, J_2, \dots, J_k \}$ and $G'-G_I' = \{ J_{k+1}, J_{k+2}, \dots, J_n \}$, Hence,

$$\begin{aligned}
 Z(\sigma) &= \gamma[p_2 p_1 + (p_1 + p_2)p_3 + \dots + (p_1 + p_2 + \dots + p_{k-1})p_k] \\
 &\quad + \gamma[p_{k+2} p_{k+1} + (p_{k+1} + p_{k+2})p_{k+3} + \dots + (p_{k+1} + p_{k+2} + \dots + p_{n-1})p_n] \\
 &= \gamma[(p_1 + p_2 + \dots + p_k)^2 / 2 - (p_1^2 + p_2^2 + \dots + p_k^2) / 2] \\
 &\quad + \gamma[(p_{k+1} + p_{k+2} + \dots + p_n)^2 / 2 - (p_{k+1}^2 + p_{k+2}^2 + \dots + p_n^2) / 2] \\
 &= \gamma[B^2 - (p_1^2 + p_2^2 + \dots + p_n^2) / 2] \\
 &= \gamma(B^2 - \sum_{i=1}^n p_i^2 / 2) \\
 &= \gamma(B^2 - \sum_{i=1}^n x_i^2 / 2) \\
 &= y
 \end{aligned}$$

□

Lemma 5.3: If there exists a solution to **P** with $Z(\sigma) \leq y$, then there exists a desired solution to **Partition**.

Proof: Suppose there is a schedule σ that $Z(\sigma) \leq y$. The two job sets scheduled on the two machines are denoted by G_I' and $G'-G_I'$ respectively. According to Lemma 5.1,

jobs in G_I' and $G' \setminus G_I'$ are sequenced in an arbitrary order. Suppose $G_I' = \{J_1, J_2, \dots, J_k\}$, $G' \setminus G_I' = \{J_{k+1}, J_{k+2}, \dots, J_n\}$, and $B_1 = \sum_{i=1}^k p_i$. Hence,

$$\begin{aligned}
 Z(\sigma) &= \gamma[p_2 p_1 + (p_1 + p_2)p_3 + \dots + (p_1 + p_2 + \dots + p_{k-1})p_k] \\
 &\quad + \gamma[p_{k+2} p_{k+1} + (p_{k+1} + p_{k+2})p_{k+3} + \dots + (p_{k+1} + p_{k+2} + \dots + p_{n-1})p_n] \\
 &= \gamma[(p_1 + p_2 + \dots + p_k)^2 / 2 - (p_1^2 + p_2^2 + \dots + p_k^2) / 2] \\
 &\quad + \gamma[(p_{k+1} + p_{k+2} + \dots + p_n)^2 / 2 - (p_{k+1}^2 + p_{k+2}^2 + \dots + p_n^2) / 2] \\
 &= \gamma[B_1^2 / 2 + (2B - B_1)^2 / 2 - (p_1^2 + p_2^2 + \dots + p_n^2) / 2] \\
 &= \gamma[B_1^2 / 2 + (2B - B_1)^2 / 2 - \sum_{i=1}^n x_i^2 / 2] \\
 &= \gamma(B_1^2 - 2BB_1 + 2B^2 - \sum_{i=1}^n x_i^2 / 2)
 \end{aligned}$$

It is obvious that $Z(\sigma) \leq y$ leads to the only solution of $B_I = B$. Thus, if there is a solution to **P** with $Z(\sigma) \leq y$, there exists a solution for **Partition**. $\square$

Theorem 5.3: **P** with two parallel machines is NP-complete.

Proof: Based on Lemma 5.2 and Lemma 5.3, it can be stated that there exists a solution for this instance of **P** with $Z(\sigma) \leq y$ if and only if there exists a solution for **Partition**.

Thus, **P** with two parallel machine is NP-complete. $\square$

Two parallel machine represents the basic situation of **P**. The general problem of **P** with more than two machines is more complex than the two machine case, which can be

proved based on the above. For $\mathbf{P}$ with three machines, a special instance is constructed based on constraints (5.16) to (5.19), except $M = 3$. Also, each machine is assigned a large production rate. In other words, for each machine, if all jobs are allocated to it, each job can be completed on time. The total production rate of all three machines is PR . The production rate of the first machine is $\frac{1}{2}PR$, $\frac{1}{4}PR$ for machine 2 and machine 3. If the two machines with $\frac{1}{4}PR$ are treated as a single machine, the production rate is $\frac{1}{2}PR$. This problem is identical to $\mathbf{P}$ with two machines, which is NP-complete. Furthermore, suppose there is a solution for this problem, i.e., job set $G_1 \Leftarrow \{ J_1, J_2, \dots, J_k \}$ is scheduled on machine 1 and $G' \Leftarrow \{ J_{k+1}, J_{k+2}, \dots, J_n \}$ is scheduled on the other machine which combines machine 2 and 3. However, the detailed schedule of job set $G' \Leftarrow \{ J_{k+1}, J_{k+2}, \dots, J_n \}$ on machine 2 and 3 needs to be determined, which is also identical to $\mathbf{P}$ with two machines. Thus, $\mathbf{P}$ with three machines is more complex than $\mathbf{P}$ with two machines. By a similar procedure, it can be proved that $\mathbf{P}$ with M machines is more complex than $\mathbf{P}$ with $M-1$ machines, where $M > 2$. Therefore, it can be stated that $\mathbf{P}$ with any $M \geq 2$ is NP-complete.

Furthermore, as mentioned at the beginning of this section, $\mathbf{P}$ is only a sub-problem of PENDSP with dependent weights and non-split processing. It can be stated that PENDSP with dependent weights and non-split processing is NP-complete.

5.3.4 Solution Methodologies for PENDSP with Dependent Weights and Non-split Processing

As PENDSP with dependent weights and non-split processing is NP-complete, finding optimal solution to large sized problem is not practical because of the vast amount of

computational time required. Heuristics is one way that can be applied to obtain near optimal solution in a reasonably short computational time.

According to the MIP model in section 5.3.2, idle time is possibly to be inserted in the job sequence on each machine. Thus, this problem is solved in two steps, which is in line with the typical method to solve earliness tardiness scheduling problem with distinct due dates. The two phases approach is mentioned in Sridharan and Zhou (1996). In the first phase, a sequence is generated. Then, in the second phase, deliberate idle times are inserted in the given sequence to form a schedule. For PENDSP with dependent weights and non-split processing, if the job sequence on each machine is given, the proposed backward scheduling algorithm is applied to insert idle time in the job sequence on each machine to obtain the release time of each job. Therefore, the emphasis in solving this problem is on generating a good job sequence on each machine.

As mentioned in section 2.3.3, simulated annealing (SA) is a neighborhood search technique that has produced good results for difficult combinatorial problems. It has been widely applied in the production scheduling problems in recent years. In this work, SA-based algorithms are designed in the attempt to yield good job sequence on each machine. Compared to other popular applied intelligent searching algorithms such as Tabu Search and Genetic Algorithm, SA is easier to implement. Also, for Tabu Search, it is hard to define the appropriate tabu for this problem. Tabu is commonly defined as the whole or partial job sequence in the machine scheduling problems. If the tabu is set to the partial job sequence in a schedule, some search areas that have not been explored will be forbidden in Tabu search. It is because all the schedules containing the defined

tabu will be discarded. In contrast, the long tabu will increase the search time significantly, which decreases the performance of Tabu Search. In this research, the commonly applied SA algorithm in literature, which is reviewed in section 2.3.3, is also employed to solve PENDSP with dependent weights and non-split processing. Furthermore, the SA algorithm is adapted by introducing the concept of neighboring population and diversification, which are the most important characters of Genetic Algorithm and Tabu Search. Thus, SA based algorithms are proposed in the following sub-sections.

5.3.4.1 Basic Simulated Annealing (BSA)

Based on the idea of SA, which is illustrated in section 2.3.3, a SA algorithm is developed in this section. It is termed as Basic Simulated Annealing (BSA) in this research. Before presenting the detailed steps of BSA, the following notation is defined:

ΔZ the difference between the neighboring solution and the current solution,
 $\Delta Z = OV_{cur} - OV_{nei}$, where OV_{cur} is the objective value of current solution, and
 OV_{nei} is the objective value of the neighboring solution.

T temperature, which is the control parameter and decreased by a cooling rate

r a cooling rate

TI search iterations at each value of T

BS the best solution ever found

It is noted that the criteria to evaluate a solution found by SA is the total penalty cost which is determined by the objective function of (5.7). As the solution of SA is only the

job allocation on each machine, total penalty cost can only be obtained by applying the backward scheduling algorithm to the solution of SA. In other words, the backward scheduling algorithm transfers the result of SA into a schedule, which contains the release time of each job. Therefore, PENDSP with dependent weights and non-split processing can be solved by this SA procedure.

The pseudo-code for the BSA algorithm is listed below:

1. Set T, t, r, TI .
2. Generate initial solution, set BS to the initial solution.
3. Set $n = 0$.
4. Generate a neighboring solution and calculate the objective value of it.
5. Evaluate the neighboring solution: if it is improved from current solution, replace current solution with the neighboring solution. Else, if the random number generated from uniform $(0, 1) < \exp(\Delta Z/T)$, replace current solution with the neighboring solution. Else, go to step 7.
6. If the accepted neighboring solution is better than BS , replace BS by the accepted solution.
7. $n=n+1$, if $n < TI$, go to step 4. Else, go to step 8.
8. $T=rT$.
9. If $T > t$, go to step 3. Else, go to step 10.
10. Stop.

The search procedure of BSA in the search space is denoted in Figure 5.5.

Figure 5.5 Search procedure of BSA in the search space

5.3.4.2 Population based Simulated Annealing (PSA)

As demonstrated in Figure 5.5, only one neighbor is generated in BSA. In seeking to cover larger search space at each searching iteration, thus to increase the possibility to find better solution, a neighboring population is generated at every searching iteration for each value of T . The neighboring population contains all the neighboring solutions of current solution. Thus, another SA based algorithm is proposed in this section, namely PSA. Among the neighboring population of the current solution, the best BL ones are chosen, where BL is a given positive integer. Current solution is replaced by one of the BL best neighbors. The search procedure of PSA in the search space is denoted in Figure 5.6.

Figure 5.6 Search procedure of PSA in the search space

The following pseudo-code demonstrates PSA.

1. Set T, t, r, TI, BL .
2. Generate initial solution, set BS to the initial solution.
3. Set $n=0$.
4. Generate all the neighbors of current solution, calculate the objective value of them, and store the best BL neighbors.
5. Evaluate the neighboring solutions:
 - 5.1 Set $l=1$.
 - 5.2 If l th best neighboring solution is improved from current solution, replace current solution with this neighboring solution. Else, if the random number generated from uniform $(0, 1) < \exp(\Delta Z/T)$, replace current solution with this neighboring solution.
 - 5.3 If the l th best neighboring solution is accepted, go to 6. Else, $l=l+1$.
 - 5.4 If $l < BL$, go to step 5.2. Else, accept the l th best neighboring solution and go to step 6.
6. If the accepted neighboring solution is better than BS , replace BS by the accepted solution.
7. $n=n+1$, If $n < TI$, go to step 4. Else, go to step 8.
8. $T=rT$.
9. If $T > t$, go to step 3. Else, go to step 10.
10. Stop.

Note that in step 5.4, if $l=BL$, the l th best neighbor is accepted regardless values of the random number and $\exp(\Delta Z/T)$. The reason is that if the l th best neighbor is still rejected, the current solution will be not changed. Thus, the neighboring population generated from current solution will not be changed. Therefore, PSA is becoming cycling. In order to avoid being cycling, the last one of the stored BL best neighbors is accepted even it is rejected by SA evaluation.

5.3.4.3 Population based Simulated Annealing with Diversification (PSAD)

PSA covers larger search space for each value of T compared to BSA. Therefore, there is higher probability of PSA to find better solution. However, PSA moves slowly towards the area that is far way from current search area. Although the faraway area contains better solution, PSA still needs to perform intensive search in the area around the current solution. Then, PSA could move toward the better area slowly. This search procedure is demonstrated in Figure 5.6. The search efficiency is particularly decreased by this inflexible moving strategy. In order to avoid the inflexibility, the feature of diversification is introduced to PSA.

Diversification has the purpose of guiding the algorithm to visit new regions, which may not be reached in PSA. Diversification is carried out by redirecting the search path or re-starting the search from a different initial solution. This feature enables the heuristic to avoid being limited to a subset of the solution space. The search procedure of PSAD can be illustrated in Figure 5.7.

Figure 5.7 Search procedure of PSAD in the search space

Furthermore, unlike searching TI iterations under each value of T in BSA and PSA, PSAD controls the search iteration under each value of T by two parameters namely No Improve (NI) and Max No Improve (MNI), together with the parameter of BS . When a neighbor better than BS is found in the neighboring population, BS is replaced by the neighbor and NI is set to 0. Otherwise, NI is increased by 1. If NI reaches MNI , the search under current value of T is terminated. Then, T is decreased and diversification performed. This procedure implies that if PSAD found improved solution in a local area, PSAD keeps searching in this area. If PSAD could not find improved solution for MNI iterations in a local area, PSAD moves to a new area for search. This strategy enables PSAD to search more thoroughly in the area that has historically found good solutions. After the intensive search in one area, PSAD jumps to a new area for search.

The pseudo-code of PSAD is presented below:

1. Set T, t, r, MNI, BL .
2. Generate initial solution, set BS as the initial solution.
3. Set $NI=0$.

4. Generate all the neighbors of current solution, calculate the objective value of them, and store the best BL neighbors.
5. Evaluate the neighboring solutions:
 - 5.1 Set $l=1$.
 - 5.2 If lth best neighboring solution is improved from current solution, replace current solution with the neighboring solution. Else, if the random number generated from uniform $(0, 1) < \exp(\Delta Z/T)$, replace current solution with the neighboring solution.
 - 5.3 If the lth best neighboring solution is accepted, go to step 6. Else, $l=l+1$.
 - 5.4 If $l < BL$, go to step 5.2. Else, accept the lth best neighboring solution and go to step 6.
6. If the accepted neighboring solution is better than BS , replace BS by the accepted solution, and set $NI=0$. Else, $NI=NI+1$.
7. If $NI < MNI$, go to step 4; Else, go to step 8.
8. $T=rT$.
9. Diversification. If the new initial solution is better than BS , replace BS by the new initial solution.
10. If $T > t$, go to step 3. Else, go to step 11.
11. Stop.

5.3.4.4 PSA Iterate by Diversification (PSAID)

From another viewpoint, it seems that there is more chance to find better solution with more diversifications. In PSAD, the number of diversification equals to the times of T decreasing until T reaches t . In order to free Diversification Number (DN) from the T

decreasing procedure, a new algorithm named PSAID is presented. Unlike the search iterations determined by T , r and t , PSAID's search iteration is determined by DN . Each DN has a corresponding T value. As DN increased by step of 1, T is decreased by r . However, if T is less than t but DN is less than the Maximum Diversification Number (MDN), PSAID will not stop until DN reaches MDN . In contrast, if DN reaches MDN and T is larger than t , PSAID stops. To obtain the pseudo-code of PSAID, only steps 1, 2, 10 need to be changed from the steps of PSAD. The pseudo-code of these three steps is presented below.

1. Set T , t , r , MNI , BL , MDN .
2. Set $DN = 0$. Generate initial solution, set BS as the initial solution.
10. $DN = DN + 1$. If $DN < MDN$, go to step 3. Else, go to step 11.

5.3.4. 5 Set Algorithmic Parameters

To solve PENDSP with dependent weights and non-split processing using the presented algorithms, furthermore, to compare the performance of the four algorithms, a number of decisions have to be made to determine the algorithmic parameters. These parameters can be classified into two groups: generic parameters and algorithm dependent parameters. The generic parameters are applicable in all the presented algorithms. These parameters include T , t , r , the methodology to generate initial feasible solution and the methodology to generate neighbors of current solution. The algorithm dependent parameters are listed below:

- BSA: *TI*.
- PSA: *TI* and *BL*.
- PSAD: *MNI*, *BL* and the diversification scheme
- PSAID: *MNI*, *BL*, *MDN* and the diversification scheme.

The following section describes the settings for the generic parameters and the succeeding section provides details of the configurations for the algorithm dependent parameters.

5.3.4.5.1 Set T , t and r

As mentioned earlier, SA escapes local optimal by accepting inferior neighbors by a probability. T is a major parameter to control the probability. Larger T gives higher probability to accept inferior neighbors. However, even from the beginning of the search, not all of the inferior neighbors will be accepted with a high probability. In this research, it is determined that the inferior neighbor is accepted with a probability of 0.9 only when the difference between the inferior neighbor and the current solution is within 100. That is $\exp(100/T) = 0.9$. Thus, $T = 949$. Then, set $T = 1000$ by rounding it.

Eglese (1990) reported that the typical cooling rate values used in literature lies between 0.8 and 0.99. A random set of preliminary experimental runs show that $r = 0.99$ is the best for this problem. Furthermore, t is set to 0.5. The combination of $T = 1000$, $r = 0.99$ and $t = 0.5$ appears to work better in comparison with other combination of T ranging from 1000 to 4000, and r ranging from 0.9 to 0.99. The preliminary experimental results of the four algorithms show that the combination of $T = 1000$ and r

$= 0.99$ provides similar results for the combinations of $T > 1000$ with $r = 0.99$, while the computational time is the smallest. Furthermore, for the same value of T , $r = 0.99$ provides much better solutions than $r = 0.9$.

5.3.4.5.2 Initial and Neighboring Solution Generation

According to the Observation 8 of Johnson et al. (1989), there can be advantage to start search with a good solution rather than a randomly generated one. In this section, the initial solution is generated based on the sense that uniformly distributes the jobs among the parallel machines. The heuristic used to generate an initial schedule is implemented in four phases. First, group the jobs that are allocated to the same flight. Next, sequence the job groups by the rule of earliest flight departure time first. Third, allocate jobs to the parallel machines. Finally, insert idle time in the job sequence on each machine and determine each job's release time. The first two phases are obvious, which is similar to the step 1 of the proposed backward scheduling algorithm. The last two phases are described as follows:

3. As the sequence of all the jobs are obtained in step 1 and 2, allocate jobs to machines cyclically. That is to allocate the first M jobs to the M machines respectively. Then, allocate the second M jobs to the M machines one by one. This procedure repeats until all the jobs are allocated. For each machine, the sequence of jobs belonging to the same job group can be sequenced arbitrarily. Here, the group refers to the jobs that are allocated to the same flight and also assigned to the same machine. According to Lemma 5.1, the sequence of jobs for each group on each machine does have no impact on the total cost.

4. Once the job sequence on each machine is determined in step 3, the release time of each job can be obtained by applying the backward scheduling algorithm on each machine.

Preliminary experiments show that this scheme is better than the scheme of allocating jobs to machines randomly.

Since the assembly schedule is determined by which machine each job is allocated to, a scheme to generate neighboring solution from current solution can be derived by changing the allocated machine of one job. As BSA generates only one neighbor at each iteration while the remaining three algorithms generate all the neighbors of current solution, the neighboring solution generation scheme is introduced separately for BSA and the remaining three algorithms as shown below:

- *Generate one neighbor in BSA:* Randomly select one job, change its allocated machine from m to $m+1$. If $m+1 > M$, allocate the job to the first machine. Then, apply the backward scheduling algorithm to insert idle time to the new job sequence on each machine to obtain a new schedule.
- *Generate all the neighbors of current solution and store the best BL neighbors:* The scheme is demonstrated in pseudo-code. M denotes the number of machines, N denotes number of jobs.
 1. Create a result array to store the best *BL* neighbors within the current solution's neighboring population. Initialize each unit of the array.
 2. $j=1$
 3. $k=1$

4. Allocate job j from current machine m to machine $m+1$. If $m+1 > M$, $m=1$.
5. Apply the backward scheduling algorithm to insert idle time to the new job sequence on each machine to obtain a new schedule, calculate objective value of the new schedule. If it is better than a neighbor stored in the result array, insert the neighbor into the array and delete the worst one from the result array.
6. $k = k+1$, if $k < M$, go to step 4. Else, go to step 7.
7. $j = j+1$, if $j \leq N$, go to step 3, Else, go to step 8.
8. Stop.

In light of the above pseudo-code, $(M-1)N$ neighbors are generated for current solution. It is noted that SA might generate infeasible initial and neighboring solution such that the total quantity of allocated jobs to one machine exceeds the capacity of the machine. Thus, the assembly of some jobs can not be completed on time, which results in these jobs missing their flights. In this situation, the tardy job in assembly is assigned a significantly high penalty, such that it is twice the air transportation cost of the job. Thus, the infeasible solution will have much higher cost than the feasible solution. As SA keeps finding the solution with lower cost, infeasible solutions will be discarded and better feasible solutions are expected to be found.

5.3.4.5.3 Set Algorithm Dependent Parameters

The parameters in this category are set and tested by a random set of preliminary experimental runs. The best values obtained for the parameters are presented in Table 5.2.

Table 5.2 Value for algorithm dependent parameters

Algorithms	Search iterations at each temperature (TI)	Max no-improve number (MNI)	Best number of neighbors (BL)	Max diversification number (MDN)
Basic SA (BSA)	100	^a /	/	/
Population based SA (PSA)	10	/	N	/
Population based SA with diversification (PSAD)	/	10	N	/
Population based SA iterate by diversification (PSAID)	/	20	N	1000

^a / denotes this parameter is not applicable to this algorithm.

Considering BSA, for iteration numbers TI for each value of T , it is obvious that larger TI will lead to better results. Considering the computational time issue, the reasonable TI is set to 100.

Three diversification schemes for PSAD and PSAID are developed:

- *Random*: Allocate each job to a machine chosen randomly from 1 to M .
- *Half Random*: For $N/2$ jobs if N is even, or $(N+1)/2$ jobs if N is odd, allocate them to a machine chosen randomly from 1 to M . The assignments of the remaining jobs are kept unchanged.
- *All add one*: For each job, allocate it from current machine m to the next machine $m+1$. If $m+1 > M$, allocate it to the first machine.

Preliminary experimental runs show better results by applying the *Half Random* diversification scheme for PSAD and PSAID compared to the remaining two diversification schemes.

5.3.5 Experimental Design

Following the scheme of experimental design presented in section 3.4.2, a series of computational experiments were conducted using randomly generated test problems. Instead of treating the assembly flow shop as a single machine, the assembly flow shop consists of three parallel machines. The production rate of each machine is partial to the total production rate. The total production rate is generated following the *PR* configure in the Table 3.4. The production rates of the three machines are set to be $3/10$, $3/10$ and $2/5$ to the total production rate. The input data of the assembly scheduling problem is obtained by solving the Type 3 air transportation problem, which assumes multiple destinations and split delivery. Detail of the ILP formulation of Type 3 air transportation problem can be found in section 3.4. Lingo 8.0 software is used to construct the mathematical programming model of the Type 3 air transportation problem. The problem is solved optimally, thus, to provide the input data to the parallel machine assembly scheduling problem. All the air transportation problems and the assembly scheduling problems are solved on a Pentium 4, 2.4 GHz computer with 256 MB RAM.

5.3.6 Computational Results

Computational results are presented in this section for each algorithm. As SA algorithm will accept an inferior solution by some probability, the algorithm does not always yield the same answer on the same input. The probabilistic nature of the algorithm makes it necessary to carry out multiple runs on the same problem instance in order to get meaningful results. For each problem instance, each SA algorithm runs ten times and the average value is calculated. For demonstration purpose, the detailed results for

PSAD are presented in Annexure of this chapter. To show the performance of PSAD with respect to the probabilistic nature, among results of the ten runs for each problem instance, the minimum and maximum value are identified. The average value, sample variance and standard deviation are calculated.

In Table 5.3 to Table 5.6, the results for each tested problem are presented in terms of average value of the five instances. For each instance, the value is obtained by calculating the average value of the ten runs. Each problem is denoted by *NjFfLd*. Firstly, the computational results of the four SA based algorithms are presented. Then, the best algorithm among them are chosen and compared with an alternative SA algorithm, which is developed based on the work of Radhakrishnan and Ventura (2000).

5.3.6.1 Comparison of the Four SA based Algorithms

Table 5.3 reports the results of applying the four algorithms for each problem. For each problem, the best solution among the four solutions is presented in bold in each row of Table 5.3. It is observed that the performance of PSA is inferior to BSA for the first four problems. In contrast, for most of the following problems, PSA outperforms BSA. It implies that introduce neighbouring population alone to BSA can not guarantee to enhance the performance of BSA. It is also observed that the best solution is contained in results of PSAD or PSAID. For most of the test problems, these two algorithms provide similar results. It implies that the performance of BSA can be improved by introducing both diversification and neighbouring population.

For each problem, to evaluate the performance improvement of PSAD and PSAID over BSA and PSA, furthermore, to differentiate the performances of PSAD and PSAID, a measurement named inferior percentage is defined for evaluation of the solution obtained by each algorithm for a specific problem. Among the four solutions provided by the four algorithms for a test problem, the best solution is identified. Inferior percentage represents how inferior is the current solution to the best solution. The inferior percentage of a solution is calculated by the following formula:

$$IPER = \frac{TC_{cur} - TC_{best}}{TC_{best}}, \quad (5.20)$$

where $IPER$ is the inferior percentage, TC_{cur} is current solution and TC_{best} is the best solution.

Table 5.3 Cost performances of the four algorithms

problem	Early penalty cost by the four algorithms				Inferior percentage of each algorithm's solution compared to the best solution (%)			
	BSA	PSA	PSAD	PSAID	BSA	PSA	PSAD	PSAID
20j4f2d	4835.7	5674.4	4718.7	4717.2	2.51	20.29	0.03	/
30j6f2d	5617.6	7344.2	5154.1	5155.3	8.99	42.49	/ ^a	0.02
40j8f3d	7500.7	8285.9	6315.5	6312.9	18.82	31.25	0.04	/
50j10f3d	9096.7	10612.7	7146.6	7118.4	27.79	49.09	0.40	/
60j12f3d	11903.8	11640.9	8971.5	8922.2	33.42	30.47	0.55	/
70j14f4d	12259.2	12248.0	9730.1	9680.1	26.64	26.53	0.52	/
80j16f4d	16203.5	14031.5	10051.6	9833.0	64.79	42.70	2.22	/
90j18f4d	16259.9	14308.1	10583.0	10526.8	54.46	35.92	0.53	/
100j20f5d	24538.3	20918.9	13784.1	13106.5	87.22	59.61	5.17	/

^a denotes that inferior percentage is not applicable for the best solution.

All of the inferior percentages are presented in the right half of Table 5.3. Inferior percentage also denotes the advantage of the best solution over current solution. Table 5.3 shows that the inferior percentage of BSA increases almost consistently with the problem size increases. It indicates that larger the problem, higher the improvement of applying neighboring population and diversification concepts to BSA. Note that for the last problem, the inferior percentage of BSA is 87.22%, which denotes the significant improvement of PSAID over BSA. For PSA, the overall inferior percentage is also high, which indicates the importance of diversification.

From Table 5.3, it is also interesting to note that for problem size less than 100j20f5d, the differences are minor between the solutions of PSAD and PSAID. For the problem of 100j20f5d, PSAD is nearly 6% inferior to PSAID. These might be explained by the diversification times of these two algorithms. For PSAD, the initial temperature $T=1000$ and it is decreased at the rate of 0.99 until it reaches the minimum acceptable temperature $t=0.5$. As mentioned in section 5.3.4.3, diversification is performed at each time T is decreased. It is calculated that the diversification number in PSAD is 758, which means PSAD diversifies 758 times for each instance of a problem. For PSAID, the diversification number is set to 1000 for each problem. For the problem size less than 100j20f5d, the computational results indicate that both PSAD and PSAID perform enough diversifications, which leads to similar solutions. For the problem 100j20f5d, the solution of PSAD is inferior to that of PSAID. It might be explained by the fact that PSAD's diversification time is less than that of PSAID for this problem. It can be stated that for larger problem, more diversifications lead to better solutions. However, there seems to be a maximum of diversification numbers for a certain sized problem. If the

diversification time exceeds this maximum, the improvement of the search is minor. This can be explained by the results of PSAD and PSAID for problems ranging from 20j4f2d to 90j18f4d. Even the diversification time of PSAID is 242 more than that of PSAD, the results are similar.

In addition to the solution quality, the search time is the other important criterion for comparison of the four algorithms. Table 5.4 reports the computational time of the four algorithms for each problem.

Table 5.4 Performances of the four algorithms in terms of computational time

problem	Computational time (s)				Proportion to PSAID (%)		
	BSA	PSA	PSAD	PSAID	BSA	PSA	PSAD
20j4f2d	19.89	11.52	12.79	32.76	60.71	35.16	39.04
30j6f2d	47.36	24.75	26.5	70.83	66.86	34.94	37.41
40j8f3d	90.65	46.65	49.99	133.27	68.02	35.00	37.51
50j10f3d	152.04	74.88	81.19	214.39	70.92	34.93	37.87
60j12f3d	246.73	116.63	123.4	331.81	74.36	35.15	37.19
70j14f4d	337.15	156.83	165.01	446.91	75.44	35.09	36.92
80j16f4d	505.72	232.56	246.83	665.66	75.97	34.94	37.08
90j18f4d	702.32	316.68	337.47	901.98	77.86	35.11	37.41
100j20f5d	897.18	401.45	433.27	1151.26	77.93	34.87	37.63

It is observed from Table 5.4 that PSAID always spend the longest computational time compared to the remaining three algorithms. For the computational time of PSAD, it is only 35% to 39% to PSAID's computational time. However, PSAD provides similarly good solutions compared to the results of PSAID according to Table 5.3. It is also

interesting to note that PSAD's computational time is almost half of the computational time of BSA, while the solution quality is improved significantly. PSAD spent similar computational time as PSA.

Overall, it can be stated that PSAD can obtain good solution within short computational time for most of the test problems. Therefore, it is chosen for further comparison with other algorithms.

5.3.6.2 Comparison of PSAD and an Alternative SA Algorithm

The advantage of the proposed PSAD is also identified by comparing it with an alternative SA algorithm. Radhakrishnan and Ventura (2000) proposed a SA based algorithm to solve the parallel machine scheduling problem with distinct due dates to minimize sum of absolute deviations of completion times from the due dates. Sequence dependent setup times are also taken into account. The main procedure is similar to that of BSA. However, the strategies to set the initial temperature T , minimal acceptable temperature t and cooling rate r are different from that of BSA. In addition, the number of search iterations at each temperature value also varies. Thus, an alternative SA algorithm is developed by adapting BSA using the strategies presented in Radhakrishnan and Ventura (2000). The strategies are summarized briefly below:

$$T = \Delta Z_{min} + (\Delta Z_{max} - \Delta Z_{min})/10;$$

$$t = \Delta Z_{min}$$

ΔZ_{min} and ΔZ_{max} are set to the smallest and largest uphill moves found in 100,000 moves.

An uphill moves means an inferior neighbor generated from current solution. The difference between these two is denoted by ΔZ . For the cooling process,

$$T_{n+1} = \frac{T_n}{1 + \beta T_n}, \text{ where } \beta = \frac{T - t}{ptT} \text{ and } p = \frac{50\delta N(N-1)}{2}$$

T_{n+1} denotes the decreased temperature from T_n . δ is a parameter that controls the rate of cooling. N denotes the number of jobs in the schedule. Among the four tested values of δ in Radhakrishnan and Ventura (2000), two of them are selected: $\delta = 0.1$ and $\delta = 0.01$. Clearly, larger value of δ leads to slower cooling rates.

For a specific value of T , the search consists of several stages. Each search stage includes TIS search iterations, where TIS is a given positive integer. The objective value for the last iteration of the uth search stage, Z_u , is saved. The value is compared with the objective value for the last iteration of all previous stages, $Z_0, Z_1, \dots, Z_{u-1}$. The system has reached equilibrium when the following condition is satisfied:

$$\left| \frac{Z_u - Z}{Z} \right| < \varepsilon$$

where Z is any one of $Z_0, Z_1, \dots, Z_{n-1}$, and ε is a small number. Once the equilibrium has been attained, the search at the current temperature is completed. The values of TIS and ε are set to $2N$ and 0.5 respectively, which is inline with Radhakrishnan and Ventura (2000).

Tables 5.5 and 5.6 present the computational results and computational times of the adapted BSA with $\delta = 0.01$ and $\delta = 0.1$ respectively. As larger values of δ lead to slower cooling rate, it implies more intensive search effort, which leads to better solution quality. These can be observed clearly from the following two tables. The results of adapted BSA are also compared with the results of PSAD.

Table 5.5 Computational results comparison of adapted BSA ($\delta=0.01$) with PSAD

Problem	Early penalty cost		Inferior percentage of (1) to (2) (%)	Computational time (s)		(4)/(3) (%)
	Adapted BSA (1)	PSAD (2)		Adapted BSA (3)	PSAD (4)	
20j4f2d	5035.8	4718.7	6.72	5.89	12.79	217.11
30j6f2d	5740.0	5154.1	11.37	28.83	26.5	91.91
40j8f3d	7499.6	6315.5	18.75	126.79	49.99	39.43
50j10f3d	8844.3	7146.6	23.75	405.50	81.19	20.02
60j12f3d	11326.0	8971.5	26.24	1156.78	123.4	10.67
70j14f4d	11636.7	9730.1	19.59	2408.28	165.01	6.85
80j16f4d	15380.9	10051.6	53.02	5604.06	246.83	4.40
90j18f4d	14874.0	10583.0	40.55	11240.02	337.47	3.00
100j20f5d	22271.8	13784.1	61.58	18811.82	433.27	2.30

Table 5.6 Computational results comparison of adapted BSA ($\delta=0.1$) with PSAD^a

Problem	Early penalty cost		Inferior percentage of (1) to (2) (%)	Computational time (s)		(4)/(3) (%)
	Adapted BSA (1)	PSAD (2)		Adapted BSA (3)	PSAD (4)	
20j4f2d	4795.1	4718.7	1.62	28.48	12.79	44.91
30j6f2d	5416.1	5154.1	5.08	250.50	26.5	10.58
40j8f3d	7122.8	6315.5	12.78	1188.39	49.99	4.21
50j10f3d	8373.6	7146.6	17.17	3901.34	81.19	2.08
60j12f3d	10805.5	8971.5	20.44	11373.70	123.4	1.08
70j14f4d	11145.0	9730.1	14.54	23058.22	165.01	0.72

^a The computational time of adapted BSA for 70j14f4d is more than 6 hours. Larger sized problems are not tested due to longer computational time.

First of all, comparing the results of adapted BSA in Table 5.5 and 5.6 with the results of BSA in Table 5.3 and 5.4, adapted BSA can gain improvement in terms of total cost at the expense of computational time for most of the tested problems. However,

comparing adapted BSA and PSAD, it is observed from both Table 5.5 and 5.6 that the inferior percentage of adapted BSA increases almost consistently with the problem size increases. For adapted BSA with $\delta = 0.01$, the largest inferior percentage of adapted BSA is more than 60%. For adapted BSA with $\delta = 0.1$, the inferior percentage is slightly smaller than the inferior percentage of adapted BSA with $\delta = 0.01$ for each test problem. It implies that slightly improved solution quality can be obtained by adapted BSA with $\delta = 0.1$ compared to adapted BSA with $\delta = 0.01$. After all, the largest inferior percentage of adapted BSA with $\delta = 0.1$ still reaches 20.44%. Thus, the advantage of PSAD over adapted BSA is identified through the above comparison of early penalty costs.

The advantage of PSAD can be further identified by the significantly reduced computational time compared to adapted BSA. For adapted BSA with $\delta = 0.01$, it has larger computational time than PSAD for all the problems except the first problem of 20j4f2d. From the problem of 30j6f2d, the computational time of PSAD is only a fraction of adapted BSA. With the increase of problem size, the value of the proportion decreases. Particularly, for the last problem of 100j20f5d, the computational time of adapted BSA is more than 3 hours. The computational time of PSAD is only 2.3% of adapted BSA. While for the slower cooling process with $\delta = 0.1$, the computational time of PSAD is smaller than that of adapted BSA for all the tested problems. The computational time reaches 3 hours for adapted BSA when problem size reaches 60j12f3d. At this level, computational time of PSAD is only 1% of adapted BSA. Furthermore, for larger sized problem, this ratio is decreased to less than 1%. Therefore,

through the comparison, the advantage of significantly reduced computational time of PSAD over adapted BSA is also identified.

Larger computational times of adapted BSA over PSAD imply that adapted BSA performs more intensive search in the search space than PSAD. However, the search result of PSAD is superior to adapted BSA. Also, comparing BSA and PSAD, similar implication can also be obtained. The major differences of PSAD to BSA and adapted BSA, which lead to the advantages of PSAD in terms of computational time and solution quality, are the concepts of neighboring population and diversification.

5.4 Summary

This chapter considers parallel assembly lines in assembly manufacturing. Each assembly line is treated as a single machine. Four scheduling problems are studied according to the assumptions on job dependent/independent penalty weight and split/non-split processing. Assuming job split in processing, PENDSP can be solved by applying the proposed backward scheduling algorithm. With job independent weights and non-split processing, PENDSP is formulated as a transportation problem. Therefore, the problem can be solved in polynomial time.

Assuming job dependent weight and non-split processing, PENDSP is formulated using a MIP model. It is proved that this problem is NP-complete. Thus, SA based algorithms are developed. It is observed that the basic SA which is widely applied in literature has the drawbacks of slow moving in the search space and small chance to get improved solution. Thus, the neighboring population and diversification concepts are introduced

into the basic SA. Numerical experiments show that the combination of these two concepts improve the performance of basic SA significantly.

So far, the study concerns on the deterministic assembly scheduling problems. In order to investigate the more realistic problem, the assembly scheduling problem considering machine breakdowns is studied in the next chapter.

Annexure: Demonstration of Probabilistic Nature of PSAD**Exhibit 1** Detailed results of PSAD (to be continued)

Problem		Min	Max	Average	Variance	Standard deviation
20j4f2d	Instance 1	4211.4	4237.6	4227.6	71.5	8.5
	Instance 2	6790.2	6813.1	6801.8	42.6	6.5
	Instance 3	6508.7	6513.0	6510.0	2.5	1.6
	Instance 4	3498.3	3508.8	3504.4	8.3	2.9
	Instance 5	2543.4	2561.1	2549.7	40.1	6.3
30j6f2d	Instance 1	5510.9	5613.8	5566.5	958.4	31.0
	Instance 2	6019.4	6047.7	6033.3	72.8	8.5
	Instance 3	5864.9	5886.1	5872.5	48.9	7.0
	Instance 4	4408.7	4417.5	4412.7	8.0	2.8
	Instance 5	3871.5	3896.0	3885.7	35.8	6.0
40j8f3d	Instance 1	5574.2	5586.6	5581.2	12.9	3.6
	Instance 2	5548.5	5630.5	5596.0	660.5	25.7
	Instance 3	5568.7	5682.7	5605.0	849.8	29.2
	Instance 4	7337.9	7445.3	7381.2	987.5	31.4
	Instance 5	7405.5	7424.0	7414.0	33.4	5.8
50j10f3d	Instance 1	7857.8	7899.2	7869.9	138.4	11.8
	Instance 2	6720.9	6760.4	6741.9	163.5	12.8
	Instance 3	9218.0	10806.9	9958.8	369018.9	607.5
	Instance 4	5035.3	5050.8	5044.1	29.3	5.4
	Instance 5	6096.5	6137.2	6118.4	154.7	12.4
60j12f3d	Instance 1	8655.5	9058.0	9013.6	14260.0	119.4
	Instance 2	9450.0	9493.1	9480.7	137.2	11.7
	Instance 3	7694.3	7971.7	7855.2	5281.3	72.7
	Instance 4	9433.1	9753.9	9485.5	8162.0	90.3
	Instance 5	9013.5	9032.7	9022.3	46.2	6.8
70j14f4d	Instance 1	8708.6	9268.6	8840.6	23676.8	153.9
	Instance 2	13302.0	13325.5	13319.8	46.5	6.8
	Instance 3	10929.5	11265.6	11191.7	7896.9	88.9
	Instance 4	8460.0	8850.4	8624.8	14611.7	120.9
	Instance 5	6536.8	6775.6	6673.9	4046.6	63.6

Chapter 5: The Parallel Machine Assembly Scheduling Problems

Exhibit 1 Detailed results of PSAD (continued)

80j16f4d	Instance 1	8127.6	8302.6	8184.3	2615.6	51.1
	Instance 2	8860.7	8906.4	8878.1	237.2	15.4
	Instance 3	8089.2	8381.3	8196.1	10401.4	102.0
	Instance 4	12456.3	14058.5	12896.5	340888.7	583.9
	Instance 5	11764.3	12480.0	12102.9	84974.2	291.5
90j18f4d	Instance 1	9398.7	9740.5	9677.9	9706.3	98.5
	Instance 2	12217.1	12638.4	12467.6	9720.3	98.6
	Instance 3	9830.4	9952.7	9879.6	1369.4	37.0
	Instance 4	11356.1	11552.2	11389.4	3072.2	55.4
	Instance 5	9367.1	10005.8	9500.6	35084.6	187.3
100j20f5d	Instance 1	12912.5	15545.0	14022.3	855904.1	925.2
	Instance 2	10608.6	11602.7	11212.3	70595.6	265.7
	Instance 3	13860.8	15623.7	14845.2	287331.6	536.0
	Instance 4	11077.7	12042.3	11807.1	75532.4	274.8
	Instance 5	16801.3	17320.9	17033.8	15170.1	123.2

Chapter 6: The Single Machine Assembly Scheduling Problem with Breakdowns

In chapter 4 and chapter 5, the assembly scheduling problems are investigated in deterministic situation. However, today's manufacturing environment has a highly time-varying and stochastic nature of objectives and constraints due to environmental uncertainties and executional uncertainties (Szelke and Markus 1997). These uncertainties can be triggered by machine breakdown, shortage of materials, interruption of machine operations when their performances violate quality control standards, etc. The process of interruptions and the time required for assembly to resume from the interruptions are often of a highly stochastic nature. The deterministic schedule obtained prior to the start of assembly processing is affected and become inappropriate. Thus, the deterministic schedule should be updated so as to minimize the effect of the disturbance. This chapter investigates the CESC synchronization problem considering machine breakdowns in assembly. The single machine assembly scheduling problem is investigated. For air transportation, the Type 3 air transportation problem is addressed, which considers multiple destinations and split delivery.

6.1 Problem Description

For the problem of synchronizing single machine assembly and Type 3 air transportation in the deterministic situation, the Synchronized Scheduling Methodology (SSM) provides optimal solution. However, if a machine breakdown is encountered, the

pre-determined optimal assembly and transportation performances will no longer be obtained. If assembly is resumed, the remaining jobs are released in the same sequence that is determined by the backward scheduling algorithm. The disturbance resulted from the machine breakdown will lead a certain number of orders to be late and miss their flights. It is assumed that if an order missed its flight, it is transported by a commercial flight, which has higher cost. Also, the commercial flights are assumed to have more flexibility, i.e., orders can be transported immediately after assembly.

As mentioned in Chapter 3, the air transportation ILP model yields the decision of choosing the outbound flights and the capacities to book on the flights. If the capacity is booked, it is assumed to be unchangeable. The transportation allocation is also assumed to be unchangeable. In other words, the orders are still to be scheduled to catch its pre-determined flight when assembly is resumed from machine down. If an order misses its flight, the transportation capacity assigned to this order becomes un-filled. Assume that the unloaded capacity can be sold or perished and could not be used by other orders. Therefore, the objective of the assembly scheduling problem is to schedule each order to catch its pre-determined flight, and to minimize the total penalty cost of orders that are transported by the commercial flights. In the following sections, the problem is formulated using a MIP model. The problem is proved to be NP-hard and two solution methodologies are presented for solving this problem.

6.2 Problem Formulation

A breakdown process is characterized by a breakdown time and a duration. As the breakdown disturbs the original schedule determined by the backward scheduling

algorithm, the release times of the remaining unprocessed jobs are updated when assembly resumes. The completed jobs prior to the breakdown are discarded in this scheduling problem. For the job in process at the breakdown time, it is assumed to be split into two parts. One is the part processed prior to the breakdown. It is treated as a completed job. The other is the remaining part which needs to be processed.

Regardless of how many times of breakdown are encountered, for each occurrence of breakdown, a new schedule is generated when assembly is resumed. The new schedule is generated based on the assumption that there will be no breakdown in the remaining time of the planning period. However, if breakdown occurs again, this schedule becomes inappropriate and new schedule is generated once more. Thus, without looking ahead for breakdowns in future, only the current breakdown needs to be considered in the formulation of the assembly scheduling problem. The input data consists of a set of orders to be processed, the machine capacity, the allocation of orders to flights, transportation cost by commercial flight, and delivery earliness tardiness cost per hour for each order. The objective is to minimize the total waiting cost between assembly and transportation, the total transportation cost, total delivery earliness tardiness cost, and the penalty cost of missing flight. A new schedule is generated when assembly is resumed from the breakdown. As assumed that no breakdown will be encountered from then on, it implies that no stochastic issue taken into account in this scheduling problem. Therefore, this assembly scheduling problem considering machine breakdown is equivalent to the deterministic scheduling problem without breakdown. The idea can be demonstrated in the following figure.

Figure 6.1 Procedure to solve the single machine assembly scheduling problem with breakdowns

Prior to introducing the MIP model, the following notation is defined.

N_r number of remaining unprocessed jobs when breakdown occurs

j remaining job index, $j=1, 2, \dots, N_r$.

dt downtime, denotes the time of breakdown;

DU duration of breakdown

p_j processing time of job j

C_j completion time of job j in assembly

β_{1j} per unit transportation cost of job j by the commercial flight

w_j earliness penalty weight of job j for assembly, $w_j = Q_j$.

EF_{jf} 1 if job j is earlier than flight f 's departure time, which means job j can catch flight f . 0 otherwise.

PA_{jf} predetermined air transportation allocation by the ILP model, 1 if job j is allocated to flight f , 0 otherwise.

TC_{jf} air transportation cost matrix which is determined by the ILP model.

Also, the following decision variables are defined:

PI_{ij} 1 if job i precedes job j immediately, 0 otherwise.

R_j release time of job j ;

The objective of the MIP model includes both the early penalties of the orders that can catch the determined flights and the late penalties of the orders that miss the predetermined flights. The model is expressed as follows:

Min

$$\begin{aligned} & \sum_{j=1}^{N_r} \sum_{f=1}^F PA_{jf} EF_{jf} (TC_{jf} + w_j * \text{Max}(0, D_f - C_j) + d\alpha_j * \text{Max}(0, dd_j - A_f) + d\beta_j * \text{Max}(0, A_f - dd_j)) \\ & + \sum_{j=1}^{N_r} (1 - \sum_{f=1}^F PA_{jf} EF_{jf}) \\ & * (\beta_{1j} Q_j + \sum_{f=1}^F PA_{jf} (d\alpha_j * \text{Max}(0, dd_j - (A_f + C_j - D_f)) + d\beta_j * \text{Max}(0, (A_f + C_j - D_f) - dd_j))) \end{aligned} \quad (6.1)$$

Subject to:

$$C_j = R_j + p_j \quad j=0, 1, \dots, N_r, N_r+1 \quad (6.2)$$

$$R_0 = dt + DU \quad (6.3)$$

$$R_{N_r+1} \geq \sum_{j=1}^{N_r} p_j \quad (6.4)$$

$$\sum_{j=1}^{N_r+1} PI_{ij} = 1 \quad i \neq j, \quad i=0, 1, \dots, N_r \quad (6.5)$$

$$\sum_{j=0}^{N_r} PI_{ji} = 1 \quad i \neq j, \quad i=1, \dots, N_r, N_r+1 \quad (6.6)$$

$$\sum_{j=1}^{N_r+1} PI_{j0} = 0 \quad (6.7)$$

$$\sum_{j=0}^{N_r} PI_{(N_r+1)j} = 0 \quad (6.8)$$

$$C_i - C_j - PI_{ji} LN \geq p_i - LN \quad (6.9)$$

$$EF_{jf} = 1 \quad \text{For } j, f \text{ with } C_j \leq D_f \quad (6.10)$$

$$EF_{jf} = 0 \quad \text{For } j, f \text{ with } C_j > D_f \quad (6.11)$$

$$PI_{ij} \in \{0, 1\}, \quad i, j=0, 1, \dots, N_r, N_r+1 \quad (6.12)$$

The decision variables are R_j and PI_{ij} . The first term in the objective function is the early penalties of the orders that can catch the predetermined flights. The early penalties consist of earliness costs before transportation, predetermined flight transportation costs, final delivery earliness and tardiness costs. The second term in the objective function is the late penalties of the orders that miss the predetermined flights. The late penalties consist of commercial flight transportation cost, final delivery earliness and tardiness costs.

Note that two dummy jobs are created in the schedule in order to facilitate the representation of the immediate preceding and succeeding relationships of the jobs. They are the first and the last job which have zero quantity. Constraint (6.2) represents

relationship among the release time, completion time and processing time of each job. Constraint (6.3) sets the release time of the first job to the assembly resume time, which is the sum of downtime dt and the breakdown duration DU . Constraint (6.4) sets the release time of the last job to be larger or equal to the total processing time of all the jobs. These two constraints denote that there might be inserted idle time in the job sequence. Constraint (6.5) and (6.7) ensure that all the jobs should have a preceding job except the first job. Constraint (6.6) and (6.8) ensure that all the jobs should have a succeeding job except the last job. Constraint (6.9) represents the completion time relationship between any two jobs. The completion time of a job is larger or equal to the sum of its processing time and the completion time of the immediate preceding job. Constraint (6.10) and (6.11) indicates that when a job's assembly completion time is earlier than a flight's departure time, it can catch the flight.

6.3 NP-Hardness of the Problem

In this section, the formulated problem is reduced to a single machine scheduling problem with distinct due windows, and job dependent earliness tardiness penalty weights, which is known to be NP-hard (Wan and Yen, 2002). For each job, the due window is defined by an earliest due date and a latest due date. If the completion time of the job is between the earliest and the latest due date, there is no penalty imposed.

The reduced problem: Assume the air transportation cost and transportation time for each order are zero. Also, assume the final delivery earliness penalty for each order is also zero. In other words, the early delivery to customers leads to no penalty.

Distinct due windows: For assembly, there is waiting cost if a job is completed earlier than its assembly due date. For final delivery, there is no earliness penalty. Only tardiness cost is taken into account if the order is delivered later than the final due date. In addition, as air transportation time is zero, the assembly completion time can be taken as the final delivery time. Therefore, the assembly of a job completed between its assembly due date and final due date leads to no penalty. The assembly due date of a job is determined by the departure time of the flight that the job is allocated to. As flights have distinct departure times, the assembly due dates are distinct. In addition, the final due date of each job is distinct. Therefore, the reduced problem is a distinct due windows scheduling problem.

Job dependent earliness penalty: If a job is completed earlier than its assembly due date, there is a waiting penalty, which depends on the product of the early time length and the quantity of the job.

Job dependent tardiness penalty: If an order is delivered later than its final due date, a tardy delivery penalty, which is the product of tardy time length, the order quantity and the per unit delivery tardiness cost, is incurred.

Wan and Yen (2002) show that the single machine scheduling problem with distinct due windows to minimize total weighted earliness and tardiness is NP-hard. As the reduced problem is equivalent to the problem studied by Wan and Yen (2002), the reduced problem is NP-hard. Therefore, this single machine assembly scheduling problem with breakdowns is NP-hard.

6.4 Solution Methodologies

Two methodologies are presented to solve the problem. Firstly, the schedule repair method is presented which is based on the pre-determined assembly schedule by the backward scheduling algorithm. Then, the reschedule method is presented employing the SA based algorithm.

6.4.1 Schedule Repair Method

In some cases, it is not desirable to reschedule all the jobs when a breakdown occurs. Instead, the required changes should be performed in such a way that the entire system is affected as little as possible (Roslöf et al. 2001). This process is termed schedule repair in this thesis. During the repair of the schedule, the valid parts of the original schedule should be re-used as far as possible, and only the parts affected by the disturbance adjusted (Szelke and Markus 1997). At the beginning of assembly, the original schedule obtained by the backward scheduling algorithm is executed. Suppose the machine breaks down at time dt and assembly resumes after time length DU . Thus, the jobs to be released during dt to $dt+DU$ in the original schedule determined by the backward scheduling algorithm are the disturbed jobs by the breakdown. In line with the concept of schedule repair, the original schedule after time $dt+DU$ is valid part and should be kept unchanged. Only the schedule of disturbed jobs should be adjusted.

For the schedule generated by the backward scheduling algorithm, there might be inserted idle times in the job sequence. These idle times can be utilized to process the disturbed jobs in this problem. Therefore, a heuristic to repair the disturbed schedule, namely remaining optimal heuristic is proposed in this section. The idea is to try to

insert the disturbed jobs into the idle time spans. If there are still some jobs that can not be inserted into the idle times, they are appended at the end of the original schedule.

Figure 6.2 illustrates this idea in detail.

Figure 6.2 The remaining optimal heuristic

In Figure 6.2, the x axis denotes time. The block denotes the scheduled jobs. During the downtime dt to $dt+DU$, the jobs to be processed in the original schedule determined by the backward scheduling algorithm are denoted using shaded block. These jobs are to be inserted into the idle times between job groups in the original schedule, as denoted by the arcs in Figure 6.2.

The remaining optimal heuristic is presented step by step next:

1. Sequence the jobs scheduled during dt to $dt+DU$ in the original schedule by LPT rule.
2. Insert disturbed jobs into the idle time spans between job groups. Suppose there are N_d disturbed jobs sequenced by LPT rule. There have S idle time spans from time $dt+DU$ in the original schedule. The detailed steps are:

- 2.1. Set $s = 1, j = 1$
- 2.2. If $\text{Length}[\text{span}(s)] > \text{ProcessingTime}[\text{job}(j)]$, insert job j into span s .
 $\text{Length}[\text{span}(s)] = \text{Length}[\text{span}(s)] - \text{ProcessingTime}[\text{job}(j)]$. Go to step 2.5.
 Else, go to 2.3.
- 2.3. $s = s + 1$. If $s \leq S$, go to 2.2. Else, go to 2.4.
- 2.4. Append job j at the end of the schedule, set $s = 1$.
- 2.5. $j = j + 1$. If $j > N_d$, go to 2.6. Else, go to 2.2.
- 2.6. Stop.

6.4.2 Reschedule Method using Simulated Annealing

In some cases, a complete reschedule is required rather than repair the schedule when breakdown is encountered. For rescheduling, the original schedule determined by the backward scheduling algorithm is no longer useful. The motivation of developing the reschedule method is also based on the intuition that a complete reschedule may give better result compared to the remaining optimal heuristic, which is presented in section 6.4.1.

As the problem is proved to be NP-hard in section 6.3, optimal solution can not be obtained even for reasonable sized problem. Heuristics can be utilized to obtain good near optimal solutions. There are four SA based heuristics proposed in chapter 5. Among them, the algorithm of Population based Simulated Annealing with Diversification (PSAD) is examined to perform the best in terms of both solution quality and computational time. Thus, PSAD is also applied for this reschedule problem. As mentioned in section 5.3.4.1, SA algorithm determines the job sequence on the

machine. Backward scheduling algorithm is then applied to obtain the schedule by inserting idle time in the sequence. However, in the scheduling problem studied in this chapter, idle times are not considered to be inserted in the job sequence. This is due to the following two reasons. When breakdown occurs, the main consequence is insufficient assembly capacity to complete all the jobs on time. Furthermore, the assembly late penalty cost accounting for the cost of missing flight is much higher than the early cost accounting for the waiting cost before transportation. Thus, the proposed forward scheduling algorithm is applied to obtain the schedule based on the jobs sequence determined by PSAD without considering inserted idle time. Preliminary experiments help to choose the following parameters for the PSAD algorithm:

Table 6.1 Value for algorithmic parameters of PSAD

Algorithms	Initial temperature (T)	Minimum acceptable temperature (t)	Cooling rate (r)	Max no-improve number (MNI)	Best number of neighbors (BL)
PSAD	1000	0.5	0.99	10	N_r

Generate Initial Solution: Johnson et. al. (1989) stated that better solution is expected to be obtained by SA starting from a good solution other than starting from random generated solution. As mentioned in section 6.4.1, the remaining optimal heuristic is expected to obtain a good solution. Therefore, the schedule generated by remaining optimal heuristic is taken as the initial solution of PSAD. It is worthy to mention that the performance of remaining optimal heuristic and PSAD can also be compared by taking the results obtained by the remaining optimal heuristic as the initial solution of PSAD. If the remaining optimal heuristic is good enough, PSAD can not obtain much

improvement from the initial solution. Otherwise, PSAD is tested to be more efficient to this assembly scheduling problem with breakdowns compared to the remaining optimal heuristic.

Generate Neighboring Solution: Secondly, the neighboring solution is generated by the scheme named *swap*: two jobs in the current schedule are swapped leading to a new neighbor. For the current solution, the two jobs in each pair are swapped. Thus, the neighboring population is generated. The best *BL* neighbors are selected. One of them will be chosen as the new current solution.

Diversification Scheme: Thirdly, a diversification scheme named *insert block* is applied: randomly select a block of jobs which contains k jobs. The job block can be inserted in front of any of the remaining jobs. In other words, the insertion position is selected randomly.

6.5 Experimental Design

Following the scheme of experimental design presented in section 3.4.2, a series of computational experiments were carried out using randomly generated test problems to evaluate the efficiency of the remaining optimal heuristic and PSAD algorithms. In addition, the per unit transportation cost by the commercial flight, β_{li} , is set fixed and identical for all the jobs with the same destination. The value of the fixed per unit transportation cost is set to $100+20g$, where g is the destination number of the flight.

In order to validate the performance of the remaining optimal heuristic and PSAD in solving the single machine assembly scheduling problem with different downtimes in the planning period, two downtimes are given, which are the 5th hour and the 10th hour. Also, the duration of breakdown is another important parameter that affects the schedule. Two breakdown durations of 1 hour and 3 hours are introduced. Based on the values of downtime and duration, 4 breakdowns are obtained: breakdown1[5(1)], breakdown2[10(1)], breakdown3[5(3)] and breakdown4[10(3)]. Within the squared bracket, the number denotes the downtime. The number in the parentheses denotes the duration of breakdown.

The air transportation data and the machine production rate data are generated following the scheme of experimental design presented in section 3.4.2. Nine test problems are generated. Five instances are generated for each test problem. Before considering breakdown, the problem is to synchronize Type 3 air transportation problem and the single machine assembly. Thus, SSM is applied to solve the overall CESC synchronization problem. In detail, the ILP model of the air transportation problem is solved first and the input data for the assembly scheduling problem is obtained. Subsequently, an assembly schedule is generated by the backward scheduling algorithm. Then, breakdown is introduced into the schedule. At the downtime, the finished jobs are discarded and the remaining jobs are taken as the input data for the single machine assembly scheduling problem with breakdowns.

6.6 Computational Results

In this section, the advantage of the remaining optimal heuristic is first identified by comparing the performance of the remaining optimal heuristic with two alternative scheduling repair heuristics for solving this single machine assembly scheduling problem with breakdowns. Then, the performances of the remaining optimal heuristic and PSAD are compared.

6.6.1 Comparison of the Remaining Optimal Heuristic with Two Alternative Schedule Repair Heuristics

Firstly, the remaining optimal heuristic is evaluated by comparing it with two alternative schedule repair heuristics, which are adopted in practice. For the first approach, namely append to last heuristic, the disturbed jobs that to be released during dt to $dt + DU$ in the original schedule determined by the backward scheduling algorithm are appended to the end of the original schedule. There is no inserted idle time in the new schedule for insufficient assembly capacity caused by the breakdown. By this manner, at least the jobs to be released after the time assembly is resumed, i.e., $dt + DU$ in the original schedule can be completed before their assembly due dates. This approach is illustrated in Figure 6.3.

Figure 6.3 The append to last heuristic

For the second approach, namely Shift heuristic, the sequence of the unprocessed jobs at time dt kept unchanged. New schedule is obtained by shifting the remaining original schedule to the assembly resume time $dt + DU$, and remove the idle times in the job sequence. This approach is recommended when the breakdowns are of short durations. If the duration of breakdown is long, this approach will impose large number of jobs be completed later than the due dates. This approach is illustrated in Figure 6.4.

Figure 6.4 The Shift heuristic

Breakdown1 to breakdown4 are introduced. For the 9 tested problems, by applying the remaining optimal heuristic, the append to last heuristic and the shift heuristic, the results of total penalty cost before transportation and the number of resulted tardy jobs are reported in Tables 6.2 to 6.5 respectively. The results of each test problem are presented in terms of average values of the five instances. Note that the results obtained by applying each schedule repair heuristic is for the schedule of the remaining unprocessed jobs, which are the jobs in the schedule determined by the backward scheduling algorithm from the down time to the end of the planning period.

In Table 6.2.a, 6.3.a, 6.4.a and 6.5.a, the column ‘backward scheduling algorithm’ denotes the early cost of the remaining original schedule from dt . In other words, it denotes the results in the situation of no breakdowns. This early cost is used to evaluate

the relative performances of the schedule repair heuristics. The columns ‘remaining optimal’, ‘append to last’ and ‘shift’ report the average costs when the remaining jobs are scheduled using the remaining optimal heuristic, append to last heuristic and the shift heuristics respectively. The columns T_{RO} , T_{AL} , T_{Shift} indicate the number of tardy jobs resulted from remaining optimal heuristic, append to last heuristic and the shift heuristic respectively. As the schedule is disturbed by a breakdown, there is increased cost of the repaired schedule compared to the original schedule without breakdown. The cost increment can be measured by the inferior percentage which is defined in section 5.3.6.1.

For the solution obtained by each schedule repair algorithm, the inferior percentage is calculated using the formula (5.20). For each introduced breakdown, the inferior percentage of the results obtained by each schedule repair heuristic to the results obtained by the backward scheduling algorithm for each problem are reported in Table 6.2b, Table 6.3b, Table 6.4b, and Table 6.5b respectively. The corresponding percentages of tardy jobs over the total number of remaining jobs are also reported. According to the results presented in Tables 6.2 to 6.5, it is observed that the remaining optimal heuristic outperforms the append to last heuristic and the shift heuristic in terms of both smaller inferior percentages and smaller tardy job percentage for most of the test problems with any breakdown introduced. It implies that the remaining optimal heuristic can achieve lower cost and ensure more jobs catching their flights compared to the append to last heuristic and the shift heuristic.

Table 6.2a Performances of the schedule repair heuristics with breakdown1

Problem	Total penalty cost				Number of tardy jobs		
	Backward scheduling algorithm	Remaining optimal	Append to last	Shift	T _{RO}	T _{AL}	T _{Shift}
20j4f2d	224826	236344	244346	237215	2.00	2.80	3.40
30j6f2d	294279	303118	314198	309614	2.40	2.80	4.20
40j8f3d	430775	450771	460281	459916	3.80	3.80	6.60
50j10f3d	565111	596542	617138	599910	3.40	5.00	8.20
60j12f3d	671817	689936	730794	696484	2.80	6.80	6.00
70j14f4d	774712	803750	813355	826673	4.20	4.40	11.80
80j16f4d	964081	986020	1018875	996000	3.40	6.20	7.00
90j18f4d	1014352	1039330	1096856	1051542	3.60	7.80	8.40
100j20f5d	1259602	1282958	1348478	1301524	3.20	8.40	8.40

Table 6.2b Evaluation of the schedule repair heuristics with breakdown1

Problem	Inferior percentage to backward scheduling algorithm (%)			Percentage of tardy jobs (%)		
	Remaining optimal	Append to last	Shift	T _{RO}	T _{AL}	T _{Shift}
20j4f2d	5.12	8.68	5.51	12.20	17.07	20.73
30j6f2d	3.00	6.77	5.21	10.34	12.07	18.10
40j8f3d	4.64	6.85	6.76	12.10	12.10	21.02
50j10f3d	5.56	9.21	6.16	8.33	12.25	20.10
60j12f3d	2.70	8.78	3.67	5.91	14.35	12.66
70j14f4d	3.75	4.99	6.71	7.22	7.56	20.27
80j16f4d	2.28	5.68	3.31	5.14	9.37	10.57
90j18f4d	2.46	8.13	3.67	4.86	10.54	11.35
100j20f5d	1.85	7.06	3.33	4.02	10.55	10.55

Table 6.3a Performances of the schedule repair heuristics with breakdown2

Problem	Total penalty cost				Number of tardy jobs		
	Backward scheduling algorithm	Remaining optimal	Append to last	Shift	T _{RO}	T _{AL}	T _{Shift}
20j4f2d	122958	129286	129286	129886	0.80	0.80	1.60
30j6f2d	204388	214593	215613	209604	1.20	1.40	1.80
40j8f3d	295699	304834	306513	311244	1.80	2.00	3.40
50j10f3d	382876	397348	399990	394645	1.40	1.60	3.00
60j12f3d	515240	518195	537021	534518	1.00	3.20	4.80
70j14f4d	509894	535205	546375	552352	4.20	4.80	8.80
80j16f4d	646984	665967	681635	664816	3.40	5.40	4.60
90j18f4d	711185	726594	747602	742022	4.00	5.20	8.00
100j20f5d	904754	908695	939219	924122	0.80	3.80	4.80

Table 6.3b Evaluation of the schedule repair heuristics with breakdown2

Problem	Inferior percentage to backward scheduling algorithm (%)			Percentage of tardy jobs (%)		
	Remaining optimal	Append to last	Shift	T _{RO}	T _{AL}	T _{Shift}
20j4f2d	5.15	5.15	5.63	10.26	10.26	20.51
30j6f2d	4.99	5.49	2.55	8.22	9.59	12.33
40j8f3d	3.09	3.66	5.26	9.18	10.20	17.35
50j10f3d	3.78	4.47	3.07	5.60	6.40	12.00
60j12f3d	0.57	4.23	3.74	2.92	9.36	14.04
70j14f4d	4.96	7.15	8.33	12.00	13.71	25.14
80j16f4d	2.93	5.36	2.76	7.94	12.62	10.75
90j18f4d	2.17	5.12	4.34	8.06	10.48	16.13
100j20f5d	0.44	3.81	2.14	1.44	6.86	8.66

Table 6.4a Performances of the schedule repair heuristics with breakdown3

Problem	Total penalty cost				Number of tardy jobs		
	Backward scheduling algorithm	Remaining optimal	Append to last	Shift	T _{RO}	T _{AL}	T _{Shift}
20j4f2d	224826	258299	273680	264987	5.00	5.40	6.80
30j6f2d	294279	320754	343214	331418	4.40	5.60	8.20
40j8f3d	430775	501921	514183	509466	10.20	9.20	14.00
50j10f3d	565111	632263	681067	662819	9.00	12.20	17.00
60j12f3d	671817	733519	780676	741576	9.40	11.40	13.60
70j14f4d	774712	891350	924396	916027	13.60	13.80	26.00
80j16f4d	964081	1079008	1135240	1102976	12.80	16.20	26.60
90j18f4d	1014352	1108792	1197444	1136538	14.00	17.20	25.00
100j20f5d	1259602	1343184	1438416	1348586	12.40	18.20	17.20

Table 6.4b Evaluation of the schedule repair heuristics with breakdown3

Problem	Inferior percentage to backward scheduling algorithm (%)			Percentage of tardy jobs (%)		
	Remaining optimal	Append to last	Shift	T _{RO}	T _{AL}	T _{Shift}
20j4f2d	14.89	21.73	17.86	30.49	32.93	41.46
30j6f2d	9.00	16.63	12.62	18.97	24.14	35.34
40j8f3d	16.52	19.36	18.27	32.48	29.30	44.59
50j10f3d	11.88	20.52	17.29	22.06	29.90	41.67
60j12f3d	9.18	16.20	10.38	19.83	24.05	28.69
70j14f4d	15.06	19.32	18.24	23.37	23.71	44.67
80j16f4d	11.92	17.75	14.41	19.34	24.47	40.18
90j18f4d	9.31	18.05	12.05	18.92	23.24	33.78
100j20f5d	6.64	14.20	7.06	15.58	22.86	21.61

Table 6.5a Performances of the schedule repair heuristics with breakdown4

Problem	Total penalty cost				Number of tardy jobs		
	Backward scheduling algorithm	Remaining optimal	Append to last	Shift	T _{RO}	T _{AL}	T _{Shift}
20j4f2d	122958	139133	140615	139257	2.40	2.40	2.60
30j6f2d	204388	235880	238729	235903	3.80	4.40	6.00
40j8f3d	295699	338581	345851	362554	6.00	6.20	10.20
50j10f3d	382876	421449	432867	441828	4.80	5.60	10.80
60j12f3d	515240	575137	621240	604530	7.80	13.00	14.60
70j14f4d	509894	605372	624496	603266	13.20	14.60	15.60
80j16f4d	646984	731292	762939	728263	11.20	14.40	16.40
90j18f4d	711185	793470	823056	832154	11.80	13.00	21.20
100j20f5d	904754	982987	1039624	984448	10.80	16.20	16.20

Table 6.5b Evaluation of the schedule repair heuristics with breakdown4

Problem	Inferior percentage to backward scheduling algorithm (%)			Percentage of tardy jobs (%)		
	Remaining optimal	Append to last	Shift	T _{RO}	T _{AL}	T _{Shift}
20j4f2d	13.16	14.36	13.26	30.77	30.77	33.33
30j6f2d	15.41	16.80	15.42	26.03	30.14	41.10
40j8f3d	14.50	16.96	22.61	30.61	31.63	52.04
50j10f3d	10.07	13.06	15.40	19.20	22.40	43.20
60j12f3d	11.62	20.57	17.33	22.81	38.01	42.69
70j14f4d	18.73	22.48	18.31	37.71	41.71	44.57
80j16f4d	13.03	17.92	12.56	26.17	33.64	38.32
90j18f4d	11.57	15.73	17.01	23.79	26.21	42.74
100j20f5d	8.65	14.91	8.81	19.49	29.24	29.24

6.6.2 Comparison of the Remaining Optimal Heuristic with PSAD

For the same set of problems tested in the previous section, the results of total penalty cost and tardy jobs by applying the PSAD algorithm are collected. For the four breakdowns, the corresponding results are presented in the following four tables respectively. In order to compare PSAD with remaining optimal heuristic, the relative cost and tardy job reduction of PSAD to remaining optimal heuristic are also presented in Table 6.6 to Table 6.9.

Generally, according to the results presented in the following tables, the performance of PSAD is improved from the remaining optimal heuristic in terms of cost reduction. However, the improvement is not significant, which ranges from zero to 7.07%. For most of the tested problems, the improvement of PSAD to the remaining optimal heuristic is within 5%.

Table 6.6 Performance comparison of the remaining optimal heuristic and PSAD with
breakdown1

Problem	Total penalty cost		Reduction of (1) to (2) (%)	Tardy jobs		Reduction of (3) over (4) (%)
	PSAD (1)	Remaining Optimal (2)		PSAD (3)	Remaining Optimal (4)	
20j4f2d	227571	236344	3.71	2.20	2.00	-10.00
30j6f2d	299515	303118	1.19	2.20	2.40	8.33
40j8f3d	438040	450771	2.82	3.40	3.80	10.53
50j10f3d	575102	596542	3.59	3.20	3.40	5.88
60j12f3d	677592	689936	1.79	3.20	2.80	-14.29
70j14f4d	785505	803750	2.27	6.20	4.20	-47.62
80j16f4d	956983	986020	2.95	1.40	3.40	58.82
90j18f4d	1017026	1039330	2.15	5.80	3.60	-61.11
100j20f5d	1273828	1282958	0.71	5.60	3.20	-75.00

Table 6.7 Performance comparison of the remaining optimal heuristic and PSAD with
breakdown2

Problem	Total penalty cost		Reduction of (1) to (2) (%)	Tardy jobs		Reduction of (3) over (4) (%)
	PSAD (1)	Remaining Optimal (2)		PSAD (3)	Remaining Optimal (4)	
20j4f2d	126816	129286	1.91	1.20	0.80	-50.00
30j6f2d	206461	214593	3.79	1.00	1.20	16.67
40j8f3d	298975	304834	1.92	2.00	1.80	-11.11
50j10f3d	383231	397348	3.55	1.60	1.40	-14.29
60j12f3d	511647	518195	1.26	0.20	1.00	80.00
70j14f4d	516024	535205	3.58	1.80	4.20	57.14
80j16f4d	642647	665967	3.50	1.20	3.40	64.71
90j18f4d	721992	726594	0.63	4.60	4.00	-15.00
100j20f5d	898858	908695	1.08	1.20	0.80	-50.00

Table 6.8 Performance comparison of the remaining optimal heuristic and PSAD with
breakdown3

Problem	Total penalty cost		Reduction of (1) to (2) (%)	Tardy jobs		Reduction of (3) over (4) (%)
	PSAD (1)	Remaining Optimal (2)		PSAD (3)	Remaining Optimal (4)	
20j4f2d	246494	258299	4.571	4.40	5.00	12.00
30j6f2d	313738	320754	2.187	4.60	4.40	-4.55
40j8f3d	472249	501921	5.912	8.40	10.20	17.65
50j10f3d	616382	632263	2.512	10.20	9.00	-13.33
60j12f3d	714286	733519	2.622	11.60	9.40	-23.40
70j14f4d	831433	891350	6.722	15.20	13.60	-11.76
80j16f4d	1005191	1079008	6.841	14.00	12.80	-9.37
90j18f4d	1077022	1108792	2.865	18.80	14.00	-34.29
100j20f5d	1300916	1343184	3.147	17.60	12.40	-41.94

Table 6.9 Performance comparison of the remaining optimal heuristic and PSAD with
breakdown4

Problem	Total penalty cost		Reduction of (1) to (2) (%)	Tardy jobs		Reduction of (3) over (4) (%)
	PSAD (1)	Remaining Optimal (2)		PSAD (3)	Remaining Optimal (4)	
20j4f2d	138128	139133	0.72	4.40	5.00	12.00
30j6f2d	226333	235880	4.05	4.60	4.40	-4.55
40j8f3d	332644	338581	1.75	8.40	10.20	17.65
50j10f3d	409201	421449	2.91	10.20	9.00	-13.33
60j12f3d	549917	575137	4.39	11.60	9.40	-23.40
70j14f4d	562553	605372	7.07	15.20	13.60	-11.76
80j16f4d	685645	731292	6.24	14.00	12.80	-9.37
90j18f4d	764591	793470	3.64	18.80	14.00	-34.29
100j20f5d	941201	982987	4.25	17.60	12.40	-41.94

It is mentioned in section 6.4.2 that PSAD takes the results of the remaining optimal heuristic as the initial solution and performs intensive searching in the search space to find improved solution. The slight improvement of PSAD implies that the remaining optimal heuristic also performs well in solving the single machine assembly scheduling problem with breakdowns. It is also interesting to note that there is no such consistency that earlier the machine break down in the planning period, higher the improvement of PSAD over the remaining optimal heuristic. This further indicates that the remaining optimal heuristic performs well regardless of the time of machine break down. The slight improvement of PSAD can be explained by the reduced search space for PSAD. From dt to $dt + DU$, the jobs scheduled in this duration are not dominant in the whole schedule determined by the backward scheduling algorithm for the relatively short downtime of the system compared to the entire planning period. The remaining jobs'

release times determined by the backward scheduling algorithm are still good for the new schedule. It is due to the fact that these jobs can catch their flight since penalty of missing flight dominates the total penalty costs. When rescheduling all jobs, it is still good to schedule these jobs near the original release time. Thus, the challenge is to find a good schedule for the disturbed jobs. As these jobs only form a minority proportion of total jobs compared to the total remaining jobs, the search space of PSAD is significantly reduced. Therefore, only slight improvement can be obtained by PSAD from the results of the remaining optimal heuristic.

Considering breakdown1 and breakdown3, the downtime is identical and duration of breakdown3 is longer than that of breakdown1. It is observe that for most of the problems, PSAD achieves larger cost reduction against the remaining optimal heuristic in breakdown3 compared to that of breakdown1. Similar observation can also be obtained in comparison of the results of breakdown2 and breakdown4. It implies that longer the breakdown duration length higher the improvement of PSAD to the remaining optimal heuristic. This in turn implies that the performance of the remaining optimal heuristic might be decreased by the long breakdown duration since long breakdown duration will result in more disturbed jobs. The inserting logic in the remaining optimal heuristic might not guarantee good solutions that close to the optimal solution in this situation of long breakdown duration.

Although PSAD achieves lower cost compared to the remaining optimal heuristic for each test problem, there is no such consistency that PSAD leads to smaller number of tardy jobs against the remaining optimal heuristic for each test problem. This can be

explained by the fact that PSAD always finds better solutions with lower cost regardless the number of tardy jobs.

It can be stated that the remaining optimal heuristic gives good solution for the single machine assembly scheduling problem with breakdowns. PSAD is recommended to enhance the performance of the remaining optimal heuristic in the situation of long breakdown duration.

6.7 Summary

This chapter studies the single machine assembly scheduling problem considering machine breakdowns. The problem is formulated using a MIP model and shown to be NP-hard. Two methods are presented to solve this problem: schedule repair method and reschedule method. A remaining optimal heuristic is developed for schedule repair. The reschedule approach employs the algorithm of Population based SA with Diversification (PSAD), which is proposed in chapter 5. Problem with different downtimes and various breakdown durations are tested by these two approaches. The computational results indicate that the performance of PSAD is slightly superior to the remaining optimal heuristic.

Chapter 7: Application of the Synchronized Scheduling Methodology to A Supply Chain Case

The previous chapters have developed the Synchronized Scheduling Methodology (SSM) for synchronized scheduling of assembly and air transportation in CESC. This chapter presents the application of the methodology to the supply chain of a consumer electronics manufacturer, named Global Electronics Manufacturer (GEM). The purpose is to demonstrate the efficiency of SSM in improving the performance of GEM's supply chain. The case information is obtained from the interactions between our research group and staff of GEM.

7.1 Supply Chain of GEM Singapore

GEM is a multinational electronics manufacturer producing a variety of electronics products feeding customer demand all over the world. GEM is well known for the last several decades for its innovative design in personal computing products and software. Although, the United States and Europe represent GEM's largest two market places, the company aims to increase its presence in Asia-Pacific region. GEM Singapore is the regional headquarter of GEM for the Asia-Pacific region. The prominent markets are distributed in the countries including Japan, China, Korea, India, Malaysia, Indonesia, Singapore, Philippines, and Thailand. Australia and New Zealand markets are also taken as part of Asia-Pacific region. The market distribution in Asia-Pacific region is depicted in Figure 7.1.

Figure 7.1 The market distribution in Asia-Pacific region

This case is concerned with GEM's supply chain in Asia-Pacific region. Majority of the products supplied by GEM is its desktop and laptop series. GEM has an assembly facility located in Singapore. The components for the assembly of final products are outsourced to the vendors in the surrounding countries. The final products are delivered to customers by air transportation directly. The detailed order processing procedure and the material flow throughout the supply chain is demonstrated in section 1.2.

7.2 Challenges Confronted by GEM Singapore

GEM is facing fierce competition in the consumer electronics industry, especially in the computing products market. The competition advantage of GEM is obtained not only by providing high quality products at competitive price, but also through efficient supply chain management, thus to provide high delivery service level. As mentioned in chapter 1, delivery accuracy is one of the key objectives of the operations management of GEM.

The current operations management of assembly and distribution is based on heuristics, such as 'EDD+FCFS'. The orders are received through Web, phone or fax. The due date of each order is assigned by adding up the rough assembly time based on current assembly capacity and the rough air transportation time from Singapore to the destination. In addition, each order is assigned a priority. Generally, Web orders have higher priority. Also, the orders placed by individual customers, such as a company, have higher priority. In contrast, orders placed by retailers are assigned relatively lower priorities.

Generally, order release time in assembly is determined based on the due date and priority of the order. The general rule is that orders with earliest due date are released first and orders with highest priority are released first. The detailed schedule in GEM's assembly flow shop is determined by the MRP system and be adjusted according to the experience of the staff. The detail of the procedure to determine daily assembly schedule is confidential, which is named GEM's scheduling method in this chapter. It is hard to formulate the procedure that determines assembly schedule. Fortunately, with

the help of the staff in GEM, we obtain the real data of daily assembly schedule, which is enough for validation of SSM.

When assembly of an order is completed, the order is transferred to the hub in Changi airport, waiting to be allocated to flights. The flight capacities are booked subject to manager's experience. When flights to the particular destination are available, orders are chosen by the simple rule of FCFS and allocated to the flight with the earliest departure time. This procedure is illustrated in detail in section 4.4.

However, the heuristics and experience supporting the operations management of GEM have led to several unexpected consequences, which are illustrated as follows:

- *Missing flight.* GEM observes that there always have some orders missed flights in each planning period. In other words, there is no order to be allocated to the booked flight capacity because of the delay in assembly. The booked flight capacity is attempted to be sold. The delivery of the order that missed the flight is delayed.
- *Heavy penalty of missing flight.* The penalty of missing flight is significant. If an order is delayed in assembly and missed the flight, three options are available to deliver the delayed orders. The first option is to book more flight capacity in the following planning periods to deliver the delayed orders as quickly as possible. The transportation cost is likely to be doubled due to the paid but unused flight capacity in the current planning period. The second option which is often used by GEM is to assign low priority to the delayed orders in assembly. When these

orders are completed, they are transported by sea. Although the cost is low, the long delayed delivery time will seriously harm the good will of GEM, which results in lost of sales in future. The last option is to deliver the orders on time by a chartered flight. This option is for the orders place by very important customers. Of course, the cost is extremely high. GEM only did this once. It has never been tried again due to the unacceptable high cost.

- *High waiting cost in the hub.* Generally, the cost paid for holding inventory in the airport hub is more expensive than in the local warehouses. According to the strategy of GEM, finished products are transferred to the hub in airport right after assembly completed. GEM applies this strategy due to the following reasons: (i) high land cost and warehouse construction cost in Singapore, (ii) to avoid the risk of traffic jam, which may results in orders missing flight, (iii) efficient storage and retrieve service in the airport hub. Holding costs are incurred until the products are allocated to flights. The relatively long waiting time in the hub always yield large amount of money paid by GEM.

In summary, reducing the waiting time of orders staying in the hub and the frequency of orders missing flights are the two major challenges confronted by GEM Singapore.

7.3 Application of SSM to the Supply Chain of GEM Singapore

This section seeks to apply SSM to the supply chain of GEM Singapore for pilot evaluation. The key assumption is that the assembly flow shop is treated as parallel machines without machine breakdown. Also, it is assumed that split delivery is permitted for air transportation. In reality, there generally is no specific evaluation for

the final delivery earliness and tardiness. However, the early or tardy delivery really harms the delivery performance because it always leads to complaints from customers. Thus, a cost is assumed to be associated with per hour early or tardy delivery for each order based on the order priority.

The evaluation is conducted through the comparison between the performance of SSM, the method that GEM applies currently and a method adopted from literature.

7.3.1 Input Data

The overall market is divided into three sub-regions according to the geographical distribution and the importance of the customers. The sub-regions are: Japan, Australia and Asia. Orders come from the major cities of each region. Table 7.1 presents the cities and sub-regions the city belongs to. The air transportation time from Singapore to each city is also reported.

Table 7.1 Destination cities and the transportation times

Destination	City	Sub-Region	Transportation time (h)
1	TOKYO	JAPAN	6.83
2	OSAKA	JAPAN	6.33
3	HONGKONG	ASIA	3.67
4	TAIPEI	ASIA	4.5
5	SHANGHAI	ASIA	5
6	KUALALUMPUR	ASIA	0.92
7	SEOUL	ASIA	6.33
8	BANGKOK	ASIA	2.33
9	NEWDELHI	ASIA	5.33
10	SYDNEY	AUSTRALIA	7.33
11	PERTH	AUSTRALIA	4.92
12	SINGAPORE	ASIA	/ ^a

^a Singapore orders are delivered by local land transportation.

Chapter 7: Application of the Synchronized Scheduling Methodology to A Supply Chain Case

The planning period is 24 hours. Hereafter, the planning period is represented by the term 'day'. Each day, orders are received with the information of quantity and destination. Information assigned to the order after acceptance includes: order number ID, priority and due date. Note that due date is assigned according to the information of assembly lead time, assembly capacity of the day, orders to be processed on the day, and transportation time to the destination. The delivery earliness and tardiness cost of each order is assigned based on the order priority. Generally, orders with higher priority are assigned higher delivery earliness and tardiness cost. The waiting cost of an order in the hub depends on the quantity and waiting time of the order. The waiting cost is set to 1 per unit product per hour. The quantity of an order varies from 1 to hundreds.

Orders received during the current day will be processed in the next day in the assembly flow shop. There are three assembly lines in the assembly flow shop. The production rates of the three lines are: 60 units/hour, 35 units/hour and 10 units/hour respectively. Assembly capacity can be adjusted by opening different assembly lines. The lead time of assembly is 3 hours. Prior to the beginning of the assembly processing for a specific day, schedule of the orders on each assembly line is determined. The orders are processed one after another on each assembly line without inserted idle time.

GEM delivers the orders to customers utilizing the transportation service provided by Singapore Airlines Cargo. Through a third party logistics company, GEM books the flight capacities of a day according to the assembly schedule and the flight schedule of Singapore Airlines Cargo on the day. Overall, there are more than 60 flights available each day with destinations covering all the 11 destination cities.

To analyze the scheduling of assembly and transportation for each day, large amount of order and flight data are taken into account. Also, there is considerable amount of historical daily assembly scheduling data, which helps to observe the daily assembly operation of GEM. Furthermore, several days' scheduling are analyzed in this case study. Instead of presenting the large amount of input data for every day, one day's input data is presented in the Annexure of this chapter to provide a disguised version of the real data.

7.3.2 Results Analysis

In this case study, the air transportation problem is formulated as Type 3 air transportation problem. The assembly problem is formulated as PENDSP with non-split processing and job dependent earliness weights, which is investigated in section 5.3.

For each day, based on the order data, flight data, and the production rates data, the total operation cost, which consists of the air transportation costs and the waiting costs before transportation, are obtained by applying SSM. The waiting cost is obtained by applying the algorithm of Population Based Simulated Annealing with Diversification (PSAD), which is proposed in section 5.3.4. Air transportation cost is obtained by the solving the ILP model of Type 3 air transportation, which is described in chapter 3. The total cost is compared with the real operation costs of GEM. The waiting cost of GEM is derived from GEM's assembly scheduling, which is indicated by the historical daily assembly scheduling data. The air transportation cost of GEM is obtained by applying the FCFS policy, which is illustrated in section 7.2.

In order to validate SSM furthermore, SSM is also compared with a method adopted from literature. Most of published works treat production scheduling problem and transportation problem separately. For the production scheduling problem, a large amount of papers have been published addressing machine scheduling problem with earliness and tardiness penalties. The problems are formulated based on the assumption that jobs can be delivered to customers immediately as production completed. In most cases, final delivery due date is taken as production due date by assuming zero transportation time (Lee and Chen, 2001). The CESC synchronization problem can also be solved following the methods in literature. Assuming that all orders can be transported immediately after assembly completion, the assembly scheduling problem can be formulated as an unrelated parallel machine scheduling problem with earliness and tardiness penalties. According to Table 7.1, the air transportation times are significant compared to the processing time of each order. Thus, the assembly due date of each order is obtained by subtracting air transportation time from the final due date. Only Della Croce and Trubian (2002) studied a special case of this problem. The mathematical formulation can be represented by the MIP model in Chapter 5 by removing constraint (5.14) and replacing the objective function (5.7) by

$$\text{Min } \sum_{j=1}^n d\alpha_j E_j + d\beta_j T_j, \text{ where } T_j = \text{Max}(0, C_j - d_j).$$

PSAD algorithm can also be employed to solve this assembly scheduling problem. The parameters of PSAD are set following the scheme presented in section 5.3.4. Once obtaining assembly completion time, orders are allocated to flights following FCFS rule. This method is named ‘MIP+FCFS’.

Two situations of air transportation are examined: large flight capacity and constrained flight capacity. Figure 7.2 details the scheme of comparing the results obtained by SSM, GEM's scheduling method and 'MIP+FCFS'.

Figure 7.2 Comparison between SSM, GEM's scheduling method and 'MIP+FCFS'

If GEM books flight capacity that is to be used weeks later, there is probably large flight capacity available. To examine the performances of SSM, GEM's scheduling method and 'MIP+FCFS' in the situation of sufficient flight capacity, the booked capacity of each flight is set to 200 based on the total order quantity and the flights available each day. The following two tables present the comparison between SSM, current GEM's scheduling method and 'MIP+FCFS' in terms of the waiting costs and the air transportation costs. From Table 7.2, it is observed that there are backlog orders even with large available flight capacity, following GEM's scheduling method and 'MIP+FCFS'. Taking into account the cost of backlog orders, transportation cost reduction of the ILP model in SSM over GEM's scheduling method and 'MIP+FCFS' is calculated respectively following formula (4.1). Set $\theta = 2$, which means the average

Chapter 7: Application of the Synchronized Scheduling Methodology to A Supply Chain Case

delivery cost of a backlog unit is twice of the delivery cost of a non-backlog unit. As shown in Table 7.2, the results of transportation cost reduction of ILP over GEM's scheduling method indicate that there is higher cost reduction if there is backlog resulted from GEM's scheduling method. However, the overall cost reduction is not significant in the situation of large flight capacity. Comparing the results of the ILP model with that of 'MIP+FCFS', similar observations can be obtained. Furthermore, GEM's scheduling method slightly outperforms the algorithm of 'MIP+FCFS' in terms of lower air transportation cost for most of the test problems.

Table 7.2 Air transportation costs with large available flight capacity

Day	Air Transportation Costs			Backlog of				Cost reduction with $\theta=2$ (%)	
				(2)		(3)			
	ILP (1)	GEM's schedule (2)	MIP+ FCFS (3)	units	Percent age (%)	units	Percen tage (%)	(1) over (2)	(1) over (3)
1	138421.7	133441	140498	400	17.89	422	18.87	27.75	32.76
2	21029.7	21426.4	21369.5	/ ^a	/	/	/	1.85	1.59
3	64706.5	70520.4	75911.7	/	/	/	/	8.24	14.76
4	67536.4	69973.1	69609	/	/	/	/	3.48	2.98
5	61097.3	61329.9	61862.1	/	/	/	/	0.38	1.24
6	26634.1	27034.1	27430.2	/	/	/	/	1.48	2.90
7	91959.6	94686.9	116006	3	0.20	18	1.23	3.27	22.65

^a means no backlog

Table 7.3 Waiting costs in hub with large available flight capacity

Day	Waiting costs			Reduction (%)	
	PSAD	GEM's schedule (2)	MIP+FCFS (3)	PSAD over (2)	PSAD over (3)
1	6934	21733.5	25055.1	68.10	72.32
2	28.8	1185.5	355.437	97.57	91.90
3	1642.6	3417.5	3380.38	51.94	51.41
4	628.9	3084.4	2789.8	79.61	77.46
5	139	9294.6	7829.19	98.50	98.22
6	7.4	1607.3	1123.94	99.54	99.34
7	1362.5	9687.3	9217.63	85.94	85.22

The advantage of SSM over GEM's scheduling method and 'MIP+FCFS' is also identified from the waiting cost reduction. It is observed from Table 7.3 that the waiting cost is significantly reduced by applying PSAD in SSM compared to GEM's scheduling method and 'MIP+FCFS'. The reason is that SSM consider accurate synchronization of assembly schedule with air transportation allocation. GEM's scheduling method and 'MIP+FCFS' solve the CESC scheduling problem without detailed investigation of the relationship between assembly and air transportation. As shown in table 7.3, the performance of GEM's scheduling method is comparable with 'MIP+FCFS' in terms of waiting cost.

In order to further investigate the advantage of SSM over GEM's scheduling method and 'MIP+FCFS', the operations costs are calculated in the situation of constrained flight capacity. In reality, the capacity of each flight is not set large enough. For each day, the total available flight capacity is set equal to the total quantity of products to be transported. It is termed as constrained flight capacity in this section. The available capacity of each flight is determined by the allocation results of the ILP model in the situation of large flight capacity. In other words, the available capacity of each flight equals the units of products allocated to this flight in the ILP model's results. Because the ILP model leads to the optimal solution of the air transportation problem, the constrained flight capacity shows the optimal booking of the flight capacity, which results in the lowest transportation cost if the booked capacity is fully utilized. Results in the following table show the level that how well do the assembly schedules obtained by GEM's scheduling method and 'MIP+FCFS' synchronize with air transportation in the situation of constrained flight capacity.

Table 7.4 Air transportation costs with constrained flight capacity

Day	Air Transportation Costs			Backlog of				Cost reduction with $\theta=2$ (%)	
				(2)		(3)			
	ILP (1)	GEM's schedule (2)	MIP+ FCFS (3)	units	Percent age (%)	units	Percen tage (%)	(1) over (2)	(1) over (3)
1	138421.7	98335.1	98926.9	776	34.70	848	37.92	31.76	37.03
2	21029.7	12633.9	12633.9	88	40.37	88	40.37	29.29	29.28
3	64706.5	7484.4	3606.25	730	88.48	775	94.51	47.16	49.38
4	67536.4	55396.8	59503.7	164	20.45	98	12.22	19.48	11.22
5	61097.3	60879.1	60515.5	5	0.79	9	1.43	1.21	1.88
6	26634.1	24395.4	21788.2	34	11.60	55	18.77	13.52	16.40
7	91959.6	89602.2	78956.1	41	2.79	188	12.80	2.94	9.96

The air transportation costs are presented in the left of Table 7.4. Note that the air transportation costs resulted from GEM's schedule method and 'MIP+FCFS' only account for non-backlog orders. As SSM is applied, the ILP model of the Type 3 air transportation problem guarantees no backlog, which is explained in detail in section 4.4.2. In Table 7.4, the backlog percentage indicates the level that the assembly schedules obtained by GEM's scheduling method and 'MIP+FCFS' matching with the constrained capacity respectively. Comparing SSM and GEM's scheduling method, it is observed that the backlog percentage of each day varies. Day 5 has the lowest backlog percentage of 0.79%, which leads to the smallest gap between the results of the ILP model and GEM's transportation scheduling in terms of air transportation cost. Day 3 has the highest backlog percentage of 88.48%, which yields 47.16% cost reduction of the ILP model over GEM's scheduling method. Comparing SSM and 'MIP+FCFS', similar observation can also be obtained. Overall, the assembly schedules obtained by GEM's scheduling method and 'MIP+FCFS' do not match well with the constrained flight capacity. In addition, 'MIP+FCFS' results in higher backlog percentages

compared to GEM's scheduling method for all days except the fourth day. Thus, GEM's scheduling method slightly outperforms 'MIP+FCFS' in terms of lower air transportation cost for most of the days.

According to Table 7.5, PSAD achieves significant reduction of waiting cost compared to GEM's scheduling method. However, for the third day, PSAD leads to much higher waiting cost than GEM's scheduling method. It is because that the third day has the highest backlog orders. Note that waiting cost is set to zero for backlog orders since the waiting cost is assumed to be included in the overall penalty costs of the backlog orders. For day 3 of GEM's scheduling, there are only 11.5% non-backlog orders whose waiting times are taken into account. Thus, it explains why PSAD leads to much higher waiting cost than GEM's scheduling method for day 3. Comparing SSM and 'MIP+FCFS', similar observation can be obtained too. Furthermore, GEM's scheduling method is comparable with 'MIP+FCFS' in terms of waiting cost.

Table 7.5 Waiting costs in hub with constrained flight capacity

Day	Waiting costs			Reduction (%)	
	PSAD	GEM's schedule (2)	MIP+FCFS (3)	PSAD over (2)	PSAD over (3)
1	6934	21389.6	28419.5	67.58	75.60
2	28.8	3537.54	1312.71	99.19	97.81
3	1642.6	1000.59	895.081	-64.16	-83.51
4	628.9	4755.38	4711.29	86.77	86.65
5	139	9492.3	8015.74	98.54	98.27
6	7.4	2017.38	1243.07	99.63	99.40
7	1362.5	11567.3	10706.3	88.22	87.27

7.4 Summary

This chapter discusses the application of SSM to the supply chain of GEM Singapore. The problem is to synchronize the scheduling of Type 3 air transportation and parallel machine assembly. The operation costs obtained by SSM are compared with the operation costs of current scheduling of GEM and the cost resulted from the algorithm of MIP+FCFS, which is adopted from literature. Both situations of large flight capacity and constrained flight capacity are examined. Results indicate that the schedule generated by SSM is superior to current GEM's scheduling method and 'MIP+FCFS'. The waiting cost before transportation is reduced by 50% to 99%. SSM guarantees no backlog, while there are backlog orders resulted from GEM's schedule method and 'MIP+FCFS' in both situations of large and constrained flight capacity. Total air transportation cost is also decreased by SSM over GEM's schedule method and 'MIP+FCFS'. For the methods without consideration of synchronization between assembly scheduling and air transportation allocation, GEM's schedule method slightly outperforms 'MIP+FCFS'.

Annexure: Input Data of Day 1

Exhibition 1. Order data (to be continued)

Order ID	Receive Time	Quantity	City ID	Order Priority	Earliness Cost	Tardiness Cost	Due date
1	1/6/2003 0:51	1	TOKYO	C	6	8	27.8
2	1/6/2003 0:51	10	OSAKA	E	8	10	23.0
3	1/6/2003 2:00	1	SINGAPORE	xx ^a	1	2	20.9
4	1/6/2003 2:04	1	TOKYO	D	7	9	8.6
5	1/6/2003 2:14	21	KUALALUMPUR	A	4	6	10.0
6	1/6/2003 2:28	1	SHANGHAI	D	7	9	24.9
7	1/6/2003 2:31	33	SYDNEY	E	8	10	21.8
8	1/6/2003 2:53	1	SHANGHAI	A	4	6	5.4
9	1/6/2003 3:09	20	KUALALUMPUR	C	6	8	16.4
10	1/6/2003 3:19	1	SEOUL	E	8	10	27.1
11	1/6/2003 3:32	30	KUALALUMPUR	D	7	9	16.9
12	1/6/2003 3:57	7	TAIPEI	xx	1	2	15.4
13	1/6/2003 4:03	1	PERTH	D	7	9	5.4
14	1/6/2003 4:28	10	NEWDELHI	A	4	6	20.5
15	1/6/2003 4:30	1	TAIPEI	A	4	6	13.1
16	1/6/2003 4:47	60	BANGKOK	E	8	10	14.8
17	1/6/2003 5:00	6	SINGAPORE	E	0	0	20.2
18	1/6/2003 5:29	2	SINGAPORE	xx	1	2	1.1
19	1/6/2003 5:44	1	KUALALUMPUR	B	5	7	1.5
20	1/6/2003 7:20	1	SHANGHAI	B	5	7	5.4
21	1/6/2003 7:30	2	TAIPEI	B	5	7	24.3
22	1/6/2003 7:38	14	OSAKA	A	4	6	17.5
23	1/6/2003 7:47	20	SINGAPORE	A	0	0	17.3
24	1/6/2003 7:51	1	HONGKONG	E	8	10	4.7
25	1/6/2003 7:56	2	SHANGHAI	xx	1	2	24.5
26	1/6/2003 8:00	4	PERTH	B	5	7	24.6
27	1/6/2003 8:05	1	SEOUL	xx	1	2	26.8
28	1/6/2003 8:13	1	KUALALUMPUR	C	6	8	1.3
29	1/6/2003 9:25	1	SHANGHAI	D	7	9	24.5
30	1/6/2003 9:40	6	TOKYO	A	4	6	26.7
31	1/6/2003 10:11	3	BANGKOK	C	6	8	2.6
32	1/6/2003 10:37	4	SHANGHAI	B	5	7	24.4
33	1/6/2003 11:30	5	SHANGHAI	xx	1	2	24.1
34	1/6/2003 11:35	2	PERTH	C	6	8	5.0
35	1/6/2003 11:39	1	TOKYO	xx	1	2	27.3
36	1/6/2003 11:53	1	PERTH	D	7	9	24.7
37	1/6/2003 11:54	1	OSAKA	B	5	7	14.8
38	1/6/2003 12:00	7	KUALALUMPUR	B	5	7	1.6

Chapter 7: Application of the Synchronized Scheduling Methodology to A Supply Chain Case

Exhibition 1. Order data (continued)

39	1/6/2003 12:24	1	SEOUL	D	7	9	7.7
40	1/6/2003 12:26	21	KUALALUMPUR	xx	1	2	14.6
41	1/6/2003 13:02	10	SINGAPORE	xx	1	2	15.7
42	1/6/2003 13:06	521	KUALALUMPUR	B	5	7	9.6
43	1/6/2003 13:08	1	PERTH	A	4	6	5.8
44	1/6/2003 14:25	2	KUALALUMPUR	D	7	9	20.9
45	1/6/2003 14:51	2	SYDNEY	D	7	9	18.1
46	1/6/2003 15:26	6	SINGAPORE	A	0	0	18.4
47	1/6/2003 16:28	1	BANGKOK	C	6	8	7.8
48	1/6/2003 16:45	2	HONGKONG	B	5	7	22.6
49	1/6/2003 16:58	16	TOKYO	B	5	7	26.5
50	1/6/2003 17:02	40	BANGKOK	E	8	10	11.0
51	1/6/2003 18:19	1	SHANGHAI	E	8	10	5.5
52	1/6/2003 18:27	1	SEOUL	E	8	10	8.7
53	1/6/2003 18:41	1	SEOUL	A	4	6	7.9
54	1/6/2003 19:02	7	BANGKOK	E	8	10	12.9
55	1/6/2003 19:12	32	NEWDELHI	A	4	6	14.8
56	1/6/2003 19:20	10	OSAKA	C	6	8	15.8
57	1/6/2003 20:00	165	HONGKONG	xx	1	2	16.6
58	1/6/2003 20:05	7	SYDNEY	D	7	9	16.7
59	1/6/2003 20:21	150	TOKYO	xx	1	2	25.0
60	1/6/2003 20:23	5	KUALALUMPUR	E	8	10	1.2
61	1/6/2003 20:30	179	SINGAPORE	D	0	0	23.8
62	1/6/2003 20:42	2	BANGKOK	B	5	7	1.6
63	1/6/2003 20:55	38	KUALALUMPUR	A	4	6	11.3
64	1/6/2003 20:58	1	SHANGHAI	B	5	7	24.7
65	1/6/2003 21:06	2	SEOUL	D	7	9	26.6
66	1/6/2003 21:41	1	TOKYO	D	7	9	27.7
67	1/6/2003 21:47	210	TAIPEI	D	7	9	24.3
68	1/6/2003 21:49	2	KUALALUMPUR	E	8	10	1.4
69	1/6/2003 21:52	263	NEWDELHI	A	4	6	11.0
70	1/6/2003 22:33	16	SYDNEY	C	6	8	27.6
71	1/6/2003 22:49	1	KUALALUMPUR	C	6	8	1.6
72	1/6/2003 22:49	10	PERTH	B	5	7	21.6
73	1/6/2003 22:56	65	BANGKOK	A	4	6	12.8
74	1/6/2003 22:58	50	OSAKA	C	6	8	13.7
75	1/6/2003 23:04	1	SEOUL	xx	1	2	8.0
76	1/6/2003 23:16	2	NEWDELHI	C	6	8	9.5
77	1/6/2003 23:20	4	TAIPEI	D	7	9	22.3
78	1/6/2003 23:25	100	OSAKA	E	8	10	22.5
79	1/6/2003 23:46	1	OSAKA	xx	1	2	26.9

^a indicates the priority is lower than A

Exhibition 2. Flight data (to be continued)

Flight No.	Destination	Departure Time	Arrival Time	Transportation Cost
SQ1	1	9:15	17:05	84
SQ2	1	9:45	17:35	83
SQ3	1	14:10	21:35	82
SQ4	1	18:00	2:00-N ^a	83
SQ5	1	20:00	5:55-N	82
SQ6	1	21:00	6:00-N	80
SQ7	1	23:40	7:30-N	81
SQ8	2	8:25	15:45	82
SQ9	2	14:10	21:35	81
SQ10	2	20:00	5:55-N	81
SQ11	2	20:30	6:00-N	81
SQ12	3	3:30	7:10	56
SQ13	3	8:30	12:15	56
SQ14	3	11:00	14:45	56
SQ15	3	13:30	17:15	60
SQ16	3	17:00	20:45	58
SQ17	3	19:00	22:45	58
SQ18	4	8:30	13:05	61
SQ19	4	17:10	21:45	61
SQ20	4	17:20	21:50	61
SQ21	5	8:10	13:20	63
SQ22	5	10:15	15:25	60
SQ23	5	15:50	21:00	65
SQ24	5	16:55	22:05	63
SQ25	6	8:30	9:25	36
SQ26	6	12:35	13:30	34
SQ27	6	14:55	15:50	35
SQ28	6	17:00	17:55	38
SQ29	6	20:00	20:55	38
SQ30	7	8:00	15:20	72
SQ31	7	9:05	16:25	71
SQ32	7	10:00	17:10	72
SQ33	7	22:55	8:45-N	74
SQ34	7	23:45	6:50-N	71
SQ35	8	8:50	10:10	48
SQ36	8	10:40	12:00	46
SQ37	8	16:00	17:20	48
SQ38	8	18:30	19:50	47
SQ39	8	20:00	21:20	47
SQ40	9	6:00	8:50	51

Chapter 7: Application of the Synchronized Scheduling Methodology to A Supply Chain Case

Exhibition 2. Flight data (continued)

SQ41	9	7:40	10:45	53
SQ42	9	18:50	21:40	53
SQ43	10	7:35	17:00	90
SQ44	10	9:40	20:15	88
SQ45	10	20:00	5:20-N	88
SQ46	10	20:20	5:40-N	87
SQ47	10	20:40	7:15-N	89
SQ48	11	9:35	14:30	68
SQ49	11	18:50	23:45	69
SQ50	11	19:40	0:40-N	67

^a means next day

Exhibition 3. GEM's assembly scheduling (to be continued)

Date of production	Assembly line	Order ID	Order quantity
2/6/2003	1	42	521
2/6/2003	1	5	21
2/6/2003	1	50	40
2/6/2003	1	63	38
2/6/2003	1	73	65
2/6/2003	1	55	32
2/6/2003	1	7	33
2/6/2003	1	16	60
2/6/2003	1	78	100
2/6/2003	1	9	20
2/6/2003	1	11	30
2/6/2003	1	23	20
2/6/2003	1	67	210
2/6/2003	1	61	179
2/6/2003	2	34	2
2/6/2003	2	18	2
2/6/2003	2	60	5
2/6/2003	2	62	2
2/6/2003	2	28	1
2/6/2003	2	8	1
2/6/2003	2	39	1
2/6/2003	2	20	1
2/6/2003	2	68	2
2/6/2003	2	13	1
2/6/2003	2	51	1

Chapter 7: Application of the Synchronized Scheduling Methodology to A Supply Chain Case

Exhibition 3. GEM's assembly scheduling (continued)

2/6/2003	2	53	1
2/6/2003	2	19	1
2/6/2003	2	75	1
2/6/2003	2	71	1
2/6/2003	2	69	263
2/6/2003	2	57	165
2/6/2003	2	59	150
2/6/2003	2	30	6
2/6/2003	2	14	10
2/6/2003	2	77	4
2/6/2003	2	70	16
2/6/2003	2	49	16
2/6/2003	2	46	6
2/6/2003	2	48	2
2/6/2003	2	33	5
2/6/2003	2	17	6
2/6/2003	2	65	2
2/6/2003	2	32	4
2/6/2003	2	35	1
2/6/2003	2	27	1
2/6/2003	2	29	1
2/6/2003	2	25	2
2/6/2003	2	79	1
2/6/2003	2	26	4
2/6/2003	2	64	1
2/6/2003	2	36	1
2/6/2003	2	10	1
2/6/2003	2	21	2
2/6/2003	2	6	1
2/6/2003	2	3	1
2/6/2003	2	66	1
2/6/2003	2	1	1
2/6/2003	2	44	2
2/6/2003	3	38	7
2/6/2003	3	4	1
2/6/2003	3	43	1
2/6/2003	3	24	1
2/6/2003	3	31	3
2/6/2003	3	52	1
2/6/2003	3	74	50
2/6/2003	3	47	1

Chapter 7: Application of the Synchronized Scheduling Methodology to A Supply Chain Case

Exhibition 3. GEM's assembly scheduling (continued)

2/6/2003	3	76	2
2/6/2003	3	58	7
2/6/2003	3	37	1
2/6/2003	3	56	10
2/6/2003	3	15	1
2/6/2003	3	45	2
2/6/2003	3	22	14
2/6/2003	3	12	7
2/6/2003	3	54	7
2/6/2003	3	40	21
2/6/2003	3	41	10
2/6/2003	3	2	10
2/6/2003	3	72	10

Chapter 8: Summary, Conclusions and Future Work

This chapter reviews the work done in this research. The contributions and conclusions are also summarized. Limitations of this research and the possible directions for the future work are proposed at the end of this chapter.

8.1 Summary and Conclusions

In this research, a Synchronized Scheduling Methodology (SSM) is proposed to synchronize the scheduling of assembly and air transportation within the complex Consumer Electronics Supply Chain (CESC) to achieve accurate delivery with minimized intermediate waiting time and reduced air transportation cost. The overall problem is first decomposed into air transportation problem and assembly scheduling problem. Then, the overall problem is solved by solving air transportation problem first and the assembly scheduling problem later. The two sub-problems are formulated and solved considering synchronization with each other. The allocation of orders to flights is constrained by assembly capacity. On the other hand, the assembly due date of each order is determined by the departure time of the flight that the order is allocated to, which is obtained by solving the air transportation problem. Based on various assumptions, different formulations of the sub-problems are proposed. For the air transportation problem, it is first addressed in the situation of single destination and split delivery. Then, the air transportation problem with multiple destinations and non-split delivery is formulated. For the assembly scheduling problem, it is first formulated

as a single machine scheduling problem. Subsequently, the parallel machine assembly scheduling problem is addressed. Furthermore, the single machine assembly scheduling problem considering machine breakdowns is investigated. By replacing the formulation of each sub-problem, SSM can solve the CESC synchronization problem in different situations.

The objective of the air transportation problem is to obtain optimal allocation of orders to the available flights subject to minimized total cost, which consists of transportation cost, delivery earliness and tardiness costs. With respect to the assumptions on single or multiple destination and the assumptions on split/non-split delivery, four types of the air transportation problems are addressed in chapter 3. Four ILP models are developed for the corresponding problems. It is proved that the Type 1 and Type 3 air transportation problems are solvable in polynomial computational time, whereas Type 2 and Type 4 air transportation problems are NP-complete.

In chapter 4, the assembly scheduling problem is first formulated as a single machine scheduling problem to minimize earliness without tardiness. The orders allocated to the same flights have the same assembly due date. Also, there is sufficient assembly capacity to complete all the orders on time, which is a constraint of the ILP model of the air transportation problem. Thus, the problem can be solved in polynomial time. Two algorithms namely forward scheduling algorithm and backward scheduling algorithm are developed, which are based on forward scheduling logic and backward scheduling logic respectively. Computational results indicate that backward scheduling algorithm outperforms forward scheduling algorithm and the industry practice heuristic

in synchronizing the schedule of single machine assembly with Type 1 and Type 3 air transportation in CESC.

Considering parallel machines in assembly, chapter 5 studies a Parallel machine Earliness Non-common Due date Scheduling Problem (PENDSP). In detail, four types of PENDSP are studied subject to the assumptions on split or non-split processing and the assumptions on job dependent or independent penalty weight. It is proved that the two sub-problems of PENDSP with split processing can be solved by the backward scheduling algorithm. PENDSP with independent weights and non-split processing is formulated as a transportation problem. Thus, it is can be solved in polynomial computational time. PENDSP with dependent weights and non-split processing is proved to be NP-complete. SA based algorithms are developed to solve this problem.

The single machine assembly scheduling problem with machine breakdown is addressed in chapter 6. The problem is formulated using a MIP model and proved to be NP-hard. Various schedule repair algorithms are developed for this problem. In addition, reschedule algorithm is also proposed, which is based on the simulated annealing algorithm. Computational results indicate that the reschedule algorithm outperforms the schedule repair algorithms.

A case study addressing the supply chain of GEM is conducted in chapter 7 to demonstrate the application of SSM proposed in this thesis. The operation costs obtained by applying SSM are compared with the operation costs of current scheduling method of GEM and the 'MIP+FCFS' algorithm. Results indicate that SSM

outperforms the ‘MIP+FCFS’ algorithm and current scheduling method of GEM in terms of significantly reduced waiting cost before transportation, capability to avoid transportation backlog, and decreased total air transportation cost.

In summary, the contributions of this thesis are listed below:

- This thesis serves as a starting point to investigate the assembly scheduling and transportation scheduling problems within CESC in a synchronization perspective.
- The overall CESC synchronization problem is formulated. A method is also proposed to decompose the overall problem into sub-problems. Each sub-problem is formulated and solved considering synchronization with the other sub-problem.
- In this thesis, CESC synchronization problem is investigated in various situations of air transportation and assembly manufacturing.
- To solve the NP-complete assembly scheduling problem, an efficient SA based algorithm is proposed by introducing neighboring population, diversification plus the method to determine appropriate diversification time to the basic SA algorithm.

8.2 Limitations

This research may serve as a starting point for investigating the CESC synchronization problem. Due to the complex structure of CESC, the overall CESC synchronization problem is complicate. This research is conducted based on the simplified situations of the problem. Thus, there are some limitations of this research, which are illustrated below:

- The SSM methodology is developed based on the assumption of sufficient capacity in air transportation and assembly manufacturing. However, it is

common that the capacities of assembly and air transportation are constrained in reality.

- The assembly flow shop is treated as single machine or parallel machines, which is not normally true in real practice.
- The assembly scheduling problems are formulated based on one assumption that only one job is processed in one assembly line at a time in assembly flow shop. This assumption is not consistent with the situation in real practice. In real practice, more than one job is processed at the same time on different stations of the assembly line in the assembly flow shop.
- The air transportation ILP models focus on the allocation of orders to the available flight capacities. The detailed operations of the loading and unloading processes in air transportation are neglected by assuming that the corresponding time and cost are included in the transportation time and cost. Also, the business processing, such as customs, is not considered. In reality, these processes are complex with relatively long lead time and considerable processing cost.
- Local land transportation and destination land transportation are not considered in current SSM methodology. Actually, operations of these two transportation legs are complex.

8.3 Future Work

In order to improve the applicability of the methodologies presented in this thesis, current research can be extended in the following directions:

- Although the Type 2 and Type 4 air transportation problems are proved to be NP-complete, solution methodologies for these two problems are not available.

In future, Branch and Bound algorithm or other alternative optimal algorithms can be developed for small sized problems and heuristics can be developed for large sized problems.

- Further work needs to be conducted considering constrained flight and production capacities for the four air transportation problems. In reality, it is common that the capacities of assembly and air transportation are constrained. For the air transportation problem, it means that total order quantity is larger than total flight capacities. The objective of the problem is to select appropriate orders to fulfill the flight capacities, i.e., to accept appropriate orders considering current available air transportation and production capacities. The formulation of the air transportation problem in this situation is similar to the formulations presented in Chapter 3. For example, considering single destination and order split in allocation, the problem can be formulated as an ILP model that is similar to the model of Type 1 air transportation problem. Order split indicates that orders can be accepted partially. The ILP model can be expressed by the objective function (3.3), together with constraints (3.4) and (3.8). Constraints (3.5) to (3.7) are modified as follows. For constraint (3.5), ‘=’ is replaced by ‘ $\leq$ ’. It implies that for each order, it is not necessary to allocate all its units to the flight capacities. It’s based on the assumption that orders can be accepted partially. For constraints (3.6) and (3.7), ‘ $\leq$ ’ is replaced by ‘=’. They indicate that all the flight capacities are fulfilled. The model can be constructed in Lingo 8.0 and be tested by random generated problems, which is similar to the work done in Chapter 3.

- It is mentioned in chapter 1 that there is a batch processor in the assembly line. In reality, it is the most important and difficult process for assembly scheduling. The assembly scheduling problem can be extended to consider the batch processor, either assume assembly is a single batch processor or include the batch processor in the single or parallel machine assembly scheduling problem.
- In this research, the assembly flow shop is simplified to single or parallel machines. It is also assumed that only one job is processed on one assembly line at a time in assembly flow shop. In reality, there are many jobs processed on each assembly line of the assembly flow shop at the same time. Future research can be conducted addressing flow shop assembly scheduling problem. First, two sequential machines in assembly should be considered. Mathematical formulation of this problem needs to be constructed. Also, computational complexity of the problem should be identified. If the problem is NP-complete, properties of the optimal solutions should be derived if that is possible. These help to develop methodologies to solve the problem optimally for small sized problems and to obtain near optimal solutions for large sized problems. Furthermore, scheduling problem of the assembly flow shop, which consists of parallel assembly lines with multiple sequential machines in each assembly line, can be addressed in future. The problem can be investigated with the similar procedure addressing the assembly scheduling problem with sequential machines, as illustrated above.
- In addition, further work can be conducted by considering machine breakdown in parallel machine assembly, or multiple stage assembly line. Therefore, both

earliness and tardiness penalties are considered in the assembly scheduling problem.

- Last but not least, local land transportation scheduling problem should be considered within the overall CESC synchronization problem.

Reference

- Abdul-Razaq, T. and C. Potts. (1988). Dynamic programming state-space relaxation for single machine scheduling. Journal of Operational Research Society, 39, 141-152.
- Achuthan, N. R. and Caccetta L. (1990). Models for vehicle routing problems. Pervan G. P., editor. Proceedings of 10th ASOR Conference, 276–294.
- Ahmed, M.U. and Sundararaghavan, P.S. (1990). Minimizing the weighted sum of late and early completion penalties in a single machine. IIE Transactions, 22(3), 288-290.
- Al-Fawzan, M. and Al-Sultan, K. (1996). The vehicle routing problem: a survey. Proceedings of the 5th Industrial Engineering Research Conference, 269-274.
- Alidaee, B. and Ahmadian, A. (1996). Scheduling on a single processor with variable speed. Information Processing Letters, 60, 189-193.
- Alidaee, B. and Dragan, I. (1997). A note on minimizing the weighted sum of tardy and early completion penalties in a single machine: a case of small common due date. European Journal of Operational Research, 96, 559–563.
- Almeida, M.T. and Centeno, M. (1998). A composite heuristic for the single machine early/tardy job scheduling problem. Computers & Operations Research, 25(7-8), 625-635.
- Anonymous, (1999). 1999 Annual report – air freight: Forwarders on the rebound. Logistics Management and Distribution Report. 38(7), 87.
- Arkin, E.M. and Roundy R.O. (1991). Weighted-tardiness scheduling on parallel machines with proportional weights. Operations Research, 39(1), 65-81.
- Ayres, R.U. and Williams, E. (2004). The digital economy: Where do we stand? Technological Forecasting and Social Change, In Press.

- Azevedo, A. L. and Sousa, J. P. (2000). A component-based approach to support order planning in a distributed manufacturing enterprise. Journal of Materials Processing Technology, 107, 431-438.
- Azizoglu, M. and Webster, S. (1997). Scheduling job families about an unrestricted common due date on a single machine. International Journal of Production Research, 35(5), 1321-30.
- Azizoglu, M. and Webster, S. (1997). Scheduling about an unrestricted common due window with arbitrary earliness/tardiness penalty rates. IIE Transactions, 29(11), 1001-1006.
- Balakrishnan N, Kanet JJ, and Sridharan V. (1999). Early/tardy scheduling with sequence dependent setups on uniform parallel machines. Computers & Operations Research, 26(2), 127-141.
- Baker, K.R. and Scudder, G.D. (1990). Sequencing with earliness and tardiness penalties: A review. Operations Research, 38(1), 22-36.
- Bank, J., Werner, F. (2001). Heuristic algorithms for unrelated parallel machine scheduling with a common due date, release dates, and linear earliness and tardiness penalties. Mathematical and Computer Modeling, 33(4-5), 363-383.
- Balfour, J. (2004). EC competition law and airline alliances. Journal of Air Transportation Management, 10(1), 81-85.
- Berger, J. and Barkaoui, M. (2004). A parallel hybrid genetic algorithm for the vehicle routing problem with time windows. Computers & Operations Research, 31(12), 2037-2053.
- Barnhart, C, Belobaba, P, Odoni, AR. (2003). Applications of operations research in the air transport industry. Transportation Science, 37(4), 368-391.

- Barnhart, C. and Schneur, R.R. (1996) Air network design for express shipment service. Operations Research, 44(6), 852-863.
- Barnhart, C., Cohn, A.M., Johnson, E.L., Klabjan, D., Nemhauser, G.L. and Vance, P.H. (2003). Airline crew scheduling. Hall, R.W., ed. Handbook of Transportation Science (2nd edition). Norwell: Kluwer Academic.
- Benjamin, J. (1989). An analysis of inventory and transportation costs in a constrained network. Transportation Science, 23(3), 177-183.
- Biskup, D. and Feldmann, M. (2001). Benchmarks for scheduling on a single machine against restrictive and unrestrictive common due dates. Computers & Operations Research, 28, 787-801.
- Blumenfeld, D.E., Burns, L.D. and Daganzo, C.F. (1991). Synchronizing production and transportation schedules. Transportation Research, 25B, 23-27.
- Blumenfeld, D.E., Burns, L.D., Diltz, J.D. and Daganzo, C.F. (1985). Analyzing trade-offs between transportation, inventory and production costs on freight networks. Transportation Research, 19B(5), 361-380.
- Bowen, J.T. (2004). The geography of freighter aircraft operations in the Pacific Basin. Journal of Transport Geography, 12, 1–11.
- Bureau of Transportation Statistics. (2000). The changing face of transportation.
<https://www.bts.gov/pdc/user/products/src/products.xml?p=480&c=-1>
- Butchers, E.R., Day, P.R., Goldie, A.P., Miller, S, Meyer, J.A., Ryan, D.M., Scott, A.C. and Wallace, C.A. (2001). Optimized crew scheduling at Air New Zealand. Interfaces, 31, 30-56.

- Cai, X., Lum, V.Y.S. and Chan, J.M.T. (1997). Scheduling about a common due date with job-dependent asymmetric earliness and tardiness penalties. European Journal of Operational Research, 98(1), 154-168.
- Cai, X. and Tu, F.S. (1996). Scheduling jobs with random processing times on a single machine subject to stochastic breakdowns to minimize early-tardy penalties. Naval Research Logistics, 43(8), 1127-1146.
- Cai, X.Q., and Zhou, X. (2000). Asymmetric earliness and tardiness scheduling with exponential processing times on an unreliable machine. Annals of Operations Research, 98, 313-331.
- Cerny, V. (1985). Thermodynamical approach to the traveling salesman problem: an efficient simulation algorithm. Journal of Optimization Theory and Applications, 45, 41-45.
- Chand, S. and Schneeberger, H. (1988). Single machine scheduling to minimize weighted earliness subject to no tardy jobs. European Journal of Operational Research, 34, 221-230.
- Chandra P. and Fisher M. (1994). Coordination of production and distribution planning. European Journal of Operational Research, 72, 503-517.
- Chang, P.C. (1999). A branch and bound approach for single machine scheduling with earliness and tardiness penalties. Computers & Mathematics with Applications, 37(10), 133-144.
- Chaovalitwongse, W., Kim, D. and Pardalos P.M. (2003). GRASP with a new local search scheme for Vehicle Routing Problems with Time Windows. Journal of Combinatorial Optimization, 7(2), 179-207.

-
- Chatwin, R.E. (1996). Multi-period airline overbooking with multiple fare classes. Naval Research Logistics, 43, 603-612.
- Chen, J.Y. and Lin S.F. (2002). Minimizing weighted earliness and tardiness penalties in single-machine scheduling with idle time permitted. Naval Research Logistics, 49(8), 760-780.
- Chen, P. (2000). Integrating production and transportation scheduling in a make-to-order environment. Ph.D. thesis, University of Cornell.
- Chen, Z.L. (1997). Scheduling with batch setup times and earliness-tardiness penalties. European Journal of Operational Research, 96(3), 518-537.
- Chen, Z.L. and Powell, W.B. (1999). A column generation based decomposition algorithm for a parallel machine just-in-time scheduling problem. European Journal of Operational Research, 116, 220-232.
- Cheng, T.C.E. (1988). Optimal common due date with limited completion time deviation. Computers & Operations Research, 15, 91-96.
- Cheng, T.C.E. and Kahlbacher, H.G. (1991). A proof for the longest-job-first policy in one-machine scheduling. Naval Research Logistics, 38, 715-720.
- Chien, T.W. (1993). Determining profit-maximizing production/shipping policies in a one-to-one direct shipping, stochastic demand environment. European Journal of Operational Research, 64, 83-102.
- Cohen, M.A. and Lee, H.L. (1988). Strategic analysis of integrated production-distribution systems: models and methods. Operations Research, 36(2), 216-228.
- Coleman, B.J. (1992). A simple model for optimizing the single machine early/tardy problem with sequence-dependent setups. Production and Operation Management, 1, 225-228.

- Crainic, T.G. (2000). Service network design in freight transportation. European Journal of Operational Research, 122, 272-288.
- Davis, J.S. and Kanet, J.J. (1993). Single machine scheduling with early and tardy completion costs. Naval Research Logistics, 40(1), 85–101.
- Della Croce F., Trubian, M. (2002). Optimal idle time insertion in early-tardy parallel machines scheduling with precedence constraints. Production Planning & Control, 13(2), 133-142.
- De, P., Ghosh, J.B. and Wells, C.E. (1991). Scheduling to minimizing weighted earliness and tardiness about a common due date. Computers & Operations Research, 18, 465–475.
- De, P., Ghosh, J.B and Wells, C.E. (1993). On the general solution for a class of early/tardy problems. Computers & Operations Research, 20, 141–149.
- Dear, R. and Sherif, Y.S. (1991). An algorithm for computer assisted sequencing and scheduling of terminal area operations. Transportation Research. 25A, 129-139.
- Destrochers, M., Lenstra, J. and Savelsbergh, M. (1990). A classification scheme for vehicle routing and scheduling problem. European Journal of Operations Research, 46, 322-332.
- Dhaenens-Flipo, C. and Finke, G. (2001). An integrated model for an industrial production-distribution problem. IIE Transactions, 33(9), 705-715.
- Eglese, R.W. (1990). Simulated annealing: A tool for operational research. European Journal of Operational Research, 46, 271-281.
- Emmons, H. (1987). Scheduling to a common due date on parallel uniform processors. Naval Research Logistics, 34, 803-810.

- Federgruen A, and Mosheiov G. (1996). Heuristics for multi-machine scheduling problems with earliness and tardiness costs. Management Science, 42, 1544–1556.
- Federgruen, A. and Mosheiov, G. (1997). Single machine scheduling problems with general breakdowns, earliness and tardiness costs. Operations Research, 45(1), 66-71.
- Feldmann, M. and Biskup, D. (2003). Single-machine scheduling for minimizing earliness and tardiness penalties by meta-heuristic approaches. Computers & Industrial Engineering, 44(2), 307-323.
- Feng, Y. and Xiao, B. (2001) A dynamic set inventory control model and its optimal policy. Operations Research. 49, 938-949.
- Fisher, M. (1995). Vehicle routing. In Ball, M.O., Monma, C.L, Magnanti, T.L. and Nemhauser, G.L., editors, Network Routing. Horth Holland, Amsterdam.
- Fisher, M.L. (1981). The lagrangian relaxation method for solving integer programming problems. Management Science, 27(1), 1-18.
- French, S. (1982). Sequencing and scheduling: an introduction to the mathematics of the job-shop. Chichester, Horwood.
- Fry, T.D., Armstrong, R.D. and Blackstone, J.H. (1987). Minimizing weighted absolute deviation in a single machine scheduling. IIE Transactions, 19, 445–450.
- Fumero, F. and Vercellis, C. (1994). Capacity analysis in repetitive assemble-to-order manufacturing systems. European Journal of Operational Research, 78, 204–215.
- Fumero, F. and Vercellis, C. (1999). Synchronized development of production, inventory and distribution schedules. Transportation science, 33, 330-340.

- Garcia, J.M., Lozano, S. and Canca, D. (2004). Coordinated scheduling of production and delivery from multiple plants. Robotics and Computer-Integrated Manufacturing, 20(3), 191-198.
- Garey, M.R. and Johnson, D.S. (1979). Computers and intractability: a guide to the theory of NP-completeness. San Francisco: Freeman.
- Garey, M.R. Tarjan, R.E and Wilfong, G.T. (1988). One-Processor scheduling with symmetric earliness and tardiness penalties. Mathematics of Operations Research, 13(2), 330-348.
- Gendreau, M., Laporte, G. and Potvin, J. (1997). Vehicle routing: Modern heuristics. In Aarts, E. and Lenstra, J., editors, Local Search in Combinational Optimization. Wiley, Chichester.
- Gerchak, Y. and Henig, M.(1989). Material commonality in assemble-to-order systems: models and properties. Naval Research Logistics, 36, 61–68.
- Gilbo, E.P. (1993). Airport capacity: Representation, estimation, optimization. IEEE Transactions on Control Systems Technology, 1, 144-154.
- Glover, F. (1989). Tabu search: part I. INFORMS Journal on Computing, 1, 190-206.
- Glover, F. (1990). Tabu search: A tutorial. Interfaces, 20, 74-94.
- Golden, B.L. and Assad, A.A., editors. (1998). Vehicle routing: methods and studies. Amsterdam: North-Holland.
- Guerrero, H.H. (1991). Demand management strategies for assemble-to-order production environments. International Journal of Production Research, 29(1), 39–51.
- Gupta, S. and Sen, T. (1983). Minimizing a quadratic function of job lateness on a single machine. Engineering Costs and Production Economics, 7, 181-194.

- Hall, N. (1986). Single- and multiple-processor models for minimizing completion-time variance. Naval Research Logistics Quarterly, 33, 49-54.
- Hall, N., Kubiak, G. W. and Sethi, S. P. (1991). Earliness–tardiness scheduling problems, II: Deviation of completion times about a restrictive common due date. Operations Research, 39, 847–856.
- Hall, N.G. and Posner, M.E. (1991). Earliness-tardiness scheduling problems, I: Weighted deviation of completion times about a common due date. Operations Research, 39, 836–846.
- Hall, N.G. and Potts, C.N. (2003). Supply chain scheduling: Batching and delivery. Operations Research, 51(4), 566-584.
- Hall, R.W. (1996). On the integration of production and distribution: Economic order and production quantity implications. Transportation Research Part B: Methodological, 30(5), 387-403.
- Han, S.H., Yun, M.H., Kwahk, J. and Hong, S.W. (2001). Usability of consumer electronic products. International Journal of Industrial Ergonomics, 28, 143-151.
- Haq, A.N., Vrat, P. and Kanda, A. (1991). An integrated production-inventory-distribution model for manufacture of urea: a case. International Journal of Production Economics, 39, 39-49.
- Harris, F.H. and Pinder, J.P. (1995). A revenue management approach to demand management and order booking in assemble-to-order manufacturing. Journal of Operations Management, 13, 299–309.
- Heady, R.B. and Zhu, Z. (1998). Minimizing the sum of job earliness and tardiness in a multimachine system. International Journal of Production Research, 36, 1619–1632.

- Ho, S.C. and Haugland, D. (2004). A tabu search heuristic for the vehicle routing problem with time windows and split deliveries. Computers & Operations Research, 31(12), 1947-1964.
- Holland, J.H. (1975). Adaptation in natural and artificial systems. Ann Arbor: The University of Michigan Press.
- Hoogeveen, J.A. and van de Velde, S.L. (1991). Scheduling around a small common due date. European Journal of Operational Research, 55, 237–242.
- Hoogeveen, J.A., Oosterhout, H. and van de Velde, S.L. (1994). New lower and upper bounds for scheduling around a small common due date. Operations Research, 42, 102–110.
- Hsu, H.M. and Wang, W.P. (2001). Possibilistic programming in production planning of assemble-to-order environments, Fuzzy Sets and Systems, 119, 59-70.
- Ishii, K., Takahashi, K. and Muramatsu, R. (1988). Integrated production, inventory and distribution systems. International Journal of Production Research, 26(3), 473-482.
- James, R.J.W. (1997). Using tabu search to solve the common due date early/tardy machine scheduling problem. Computers & Operations Research, 24, 199–208.
- James, R.J.W. and Buchanan, J.T. (1997) A neighbourhood scheme with a compressed solution space for the early/tardy scheduling problem. European Journal of Operational Research, 102(3), 513-527.
- James, R.J.W. and Buchanan, J.T. (1998). Robustness of single machine scheduling problems to earliness and tardiness penalty errors. Annals of Operations Research, 76, 219-232.

- James, R.J.W. and Buchanan, J.T. (1998). Performance enhancements to tabu search for the early/tardy scheduling problem. European Journal of Operational Research, 106(2-3), 254-265.
- Johnson, D.S, Aragon, C. R., McGeoch, L.A, and Schevon, C. (1989). Optimization by simulated annealing: An experimental evaluation; Part I, Graph Partitioning. Operations Research, 37(6), 865-892.
- Jolayemi, J.K., Olorunniwo, F.O. (2004). A deterministic model for planning production quantities in a multi-plant, multi-warehouse environment with extensible capacities. International Journal of Production Economics, 87(2), 99-113.
- Jorge, J.D. and Rus, G.D. (2004). Cost–benefit analysis of investments in airport infrastructure: a practical approach. Journal of Air Transportation Management, 10(5), 311-326.
- Kanet, J. (1981). Minimizing the average deviation of job completion times about a common due date. Naval Research Logistics Quarterly, 28, 643-651.
- Karageorgos, A., Mehandjiev, N., Weichhart, G. and Hammerle, A. (2003). Agent-based optimisation of logistics and production planning. Engineering Applications of Artificial Intelligence, 16, 335–348.
- Kim, Y.D. and Yano, C.A. (1994). Minimizing mean tardiness and earliness in single-machine problems with unequal due dates. Naval Research Logistics, 41(7), 913–933.
- Kirkpatrick, S. Gelatt, C.D. and Vecchi, M.P. (1983). Optimization by simulated annealing. Science, 220, 671-680.

- Koulamas, C. (1996). Single-machine scheduling with time windows and earliness/tardiness penalties. European Journal of Operational Research, 91(1), 190-202.
- Kovalyov, M.Y., Kubiak, W. (1999). A fully polynomial approximation scheme for the weighted earliness-tardiness problem. Operations Research, 47(5), 757-761.
- Kramer, F.J. and Lee, C.Y. (1993). Common due-window scheduling. Production and Operations Management, 2, 262-275.
- Kubiak, W., Lou, S. and Sethi, S. (1990). Equivalence of mean flow time problems and mean absolute deviation problems. Operations Research Letters, 9, 371-374.
- Laguna, M., and Velarde, J.L.G. (1991). A search heuristic for just-in-time scheduling in parallel machines. Journal of Intelligent Manufacturing, 2, 253-260.
- Laporte, G. (1992). The vehicle routing problem: An overview of exact and approximate algorithms. European Journal of Operational Research, 59, 213-247.
- Lee, C.Y. and Chen, Z.L. (2001). Machine scheduling with transportation considerations. Journal of scheduling, 4, 3-24.
- Lee, C.Y. and Choi, J.Y. (1995). A genetic algorithm for job sequencing problems with distinct due-dates and general early-tardy penalty weights. Computers & Operations Research, 22(8), 857-869.
- Lee, C.Y. and Kim, S.J. (1995). Parallel genetic algorithms for the earliness tardiness job scheduling problem with general penalty weights. Computers & Industrial Engineering, 28(2), 231-243.
- Lee, Y.H., Kim S.H. and Moon C. (2002). Production-distribution planning in supply chain using a hybrid approach. Production Planning & Control, 13(1), 35-46.

- Lee, W.S., Kim, C.H. and Sox, C.R. (2003). A dynamic production and transportation model with heterogeneous vehicle types. International Journal of Industrial Engineering-Theory Applications and Practice, 10(4), 420-426.
- Leinbach, T.R. and Bowen, J.T. (2004). Air cargo services and the electronics industry in Southeast Asia. Journal of Economic Geography, 4(3), 299-321.
- Li, G. (1997). Single machine earliness and tardiness scheduling. European Journal of Operational Research, 96(3), 546-558.
- Li, K.P., Sivakumar, A.I., Mathirajan, M. and Ganesan, V.K. (2004). Solution methodology for synchronizing assembly manufacturing and air transportation of consumer electronics supply chain. International Journal of Business, 9(4), 361-380.
- Liaw, C.F. (1999). A branch-and-bound algorithm for the single machine earliness and tardiness scheduling problem. Computers & Operations Research, 26(7), 679-693.
- Mannur, N.R. and Addagatla, J.B. (1993). Heuristic algorithms for solving earliness-tardiness scheduling problem with machine vacations. Computers and Industrial Engineering, 25(1-4), 255-258.
- Maruchek, A. and McClelland, M. (1992). Planning capacity utilization in an assemble-to-order environment. International Journal of Operations & Production Management, 12(9), 18-38.
- Mazzini, R. and Armentano, V.A. (2001). A heuristic for single machine scheduling with early and tardy costs. European Journal of Operational Research, 128, 129-146.
- Mittenthal, J. and Raghavachari, M. (1993). Stochastic single machine scheduling with quadratic early-tardy penalties. Operations Research, 41(4), 786-796.

- Mondal, S.A. (2002). Minimization of squared deviation of completion times about a common due date. Computers & Operations Research, 29(14), 2073-2085.
- Mondal, S.A. and Sen, A.K. (2001). Single machine weighted earliness-tardiness penalty problem with a common due date. Computers & Operation Research, 28(7), 649–69.
- Mosheiov, G. (1996). A note on pre-emptive scheduling with earliness and tardiness costs. Production Planning & Control, 7(4), 401-406.
- Motevalli, V. and Stough, R. (2004). Aviation safety and security: reaching beyond borders. Journal of Air Transportation Management, 10(4), 223-224.
- Nagar, A., Haddock, J. and Heragu, S. (1995). Multiple and bicriteria scheduling – a literature survey. European Journal of Operational Research, 81(1), 88-104.
- Nagy, G. and Salhi, S. (2004). Heuristic algorithms for single and multiple depot vehicle routing problems with pickups and deliveries. European Journal of Operational Research, In Press.
- Nandkeolyar, U., Ahmed, M.U. and Sundararaghavan, P.S. (1993). Dynamic single-machine weighted absolute deviation problem: Predictive heuristics and evaluation. International Journal of Production Research, 31, 1453–1466.
- Oum, T.H. and Lee, Y.H. (2002). The Northeast Asian air transport network: is there a possibility of creating Open Skies in the region? Journal of Air Transportation Management, 8(5), 325-337.
- Ow, P.S. and Morton P.E. (1989). The single machine early/tardy problem. Management Science, 35(2), 177-191.

- Özdamar, L. and Yazgac, T. (1999). A hierarchical planning approach for a production-distribution system. International Journal of Production Research, 37(16), 3759-3772.
- Panwalkar, S. S, Smith, M. L., and Seidmann, A. (1982). Common due-date assignment to minimize total penalty for the one machine scheduling problem, Operations Research, 30 , 391-399.
- Pathumnakul, S. and Egbelu, P.J. (2005). Algorithm for minimizing weighted earliness penalty in single-machine problem. European Journal of Operational Research, 161(3), 780-796.
- Rabadi, G., Mollaghasemi, M. and Anagnostopoulos G.C. (2004). A branch-and-bound algorithm for the early/tardy machine scheduling problem with a common due-date and sequence-dependent setup time. Computers & Operations Research, 31(10), 1727-1751.
- Radhakrishnan, S and Ventura, J.A. (2000). Simulated annealing for parallel machine scheduling with earliness-tardiness penalties and sequence-dependent set-up times. International Journal of Production Research, 38(10), 2233-2252.
- Rajagopalan S. (2002). Make to order or make to stock: model and application. Management Science, 48, 241-256.
- Rardin, R.L. and Uzsoy, R. (2001). Experimental evaluation of heuristic optimization algorithms: a tutorial. Journal of Heuristics, 7(3), 261-304.
- Roslöf, J., Harjunkoski, I., Björkqvist, J, Karlsson, S. and Westerlund, T. (2001). An MILP-based reordering algorithm for complex industrial scheduling and rescheduling. Computers & Chemical Engineering, 25(4-6), 821-828.

- Ruiz-Torres, A.J. and Tyworth, J.E. (1997). Simulation based approach to study the interaction of scheduling and routing on a logistic network. Proceeding of the 1997 Winter Simulation Conference.
- Sarmiento, A.M. and Nagi, R. (1999). A review of integrated analysis of production-distribution systems. IIE transactions, 31, 1061-1074.
- Sidney, J.B. (1977). Optimal single-machine scheduling with earliness and tardiness penalties. Operations Research, 25, 62-69.
- Simchi-Levi, D, Kaminsky, P. and Simchi-Levi, E. (2000). Designing and managing the supply chain: concepts, strategies and case studies. Singapore: McGraw-Hill.
- Sivrikaya-Serifoglu, F., Ulusoy G. (1999). Parallel machine scheduling with earliness and tardiness penalties. Computers & Operations Research, (8), 773-787.
- Sridharan, V. and Zhou, Z. (1996). A decision theory based scheduling procedure for single-machine weighted earliness and tardiness problems. European Journal of Operational Research, 94(2), 292-301.
- Sourd, F., Kedad-Sidhoum, S. (2003). The one-machine problem with earliness and tardiness penalties. Journal of Scheduling, 6(6), 533-549.
- Sun, H.Y., Wang, G.Q. (2003). Parallel machine earliness and tardiness scheduling with proportional weights. Computers & Operations Research, 30(5), 801-808.
- Sundararaghavan, P. and Ahmed, M. (1984). Minimizing the sum of lateness in single-machine and multimachine scheduling. Naval Research Logistics Quarterly, 31, 325-333.
- Szelke, E. and Markus G. (1997). A learning reactive scheduler using CBR/L. Computers in Industry, 33, 31-46.

- Szwarc, W. (1993). Adjacent ordering in single machine scheduling with earliness and tardiness penalties. Naval Research Logistics, 40, 229–243.
- Szwarc, W. (1996). The weighted common due date single machine scheduling problem revisited. Computers & Operations Research, 23(3), 255–262.
- Szwarc, W. and Mukhopadhyay, S.K. (1995). Optimal timing in tardiness and earliness single machine sequencing. Naval Research Logistics, 42, 1109–1114.
- Thomas, D.J. and Griffin, P.M. (1996). Coordinated supply chain management. European Journal of Operational Research, 94, 1-15.
- Toth, P. and Vigo, D. editors. (2002). The vehicle routing problem. Philadelphia, PA: SIAM.
- Tsai, W.H. and Kuo, L. (2004). Operating costs and capacity in the airline industry. Journal of Air Transportation Management, 10(4), 269-275.
- Tyan, J.C., Wang, F.K. and Du, T.C. (2003). An evaluation of freight consolidation policies in global third party logistics. Omega, 31(1), 55-62.
- Ventura, J.A., Kim, D. (2003). Parallel machine scheduling with earliness-tardiness penalties and additional resource constraints. Computers & Operations Research, 30(13), 1945-1958.
- Ventura, J.A., Kim, D. and Garriga, F. (2002). Single machine earliness-tardiness scheduling with resource-dependent release dates. European Journal of Operational Research, 142(1), 52-69.
- Ventura, J.A. and Radhakrishnan, S. (2003). Single machine scheduling with symmetric earliness and tardiness penalties. European Journal of Operational Research, 144(3), 598-612.

- Ventura, J.A., Weng, M.X. (1995). An improved dynamic programming algorithm for the single machine mean absolute deviation problem with a restrictive common due date. Operations Research Letters, 17, 149-152.
- Ventura, J.A., Weng, M.X. (1996). Single machine scheduling with a common delivery window. Journal of Operational Research Society, 47(3), 424-434.
- Verma, S. and Dessouky, M. (1998). Single-machine scheduling of unit-time jobs with earliness and tardiness penalties. Mathematics of Operations Research, 23(4), 930-943.
- Wan, G.H. and Yen, B.M.P.C. (2002). Tabu search for single machine scheduling with distinct due windows and weighted earliness/tardiness penalties. European Journal of Operational Research, 142(2), 271-281.
- Webster, S., Jog, P.D. and Gupta, A. (1998). A genetic algorithm for scheduling job families on a single machine with arbitrary earliness/tardiness penalties and an unrestricted common due date. International Journal of Production Research, 36(9), 2543-2551.
- Webster, S.T. (1997). The complexity of scheduling job families about a common due date. Operations Research Letters, 20, 65-74.
- Weng, M.X. and Ventura, J.A. (1994). Scheduling about a large common due date with tolerance to minimize mean absolute deviation in completion times. Naval Research Logistics, 41, 843-851.
- Weng, M.X. and Ventura, J.A. (1996). Single-machine earliness-tardiness scheduling about a common due date with tolerances. International Journal of Production Economics, 42(3), 217-227.

- Williams, T.M. (1984). Special products and uncertainty in production/inventory systems. European Journal of Operational Research, 15, 46–54.
- Winston, W.L. (1994). Operations research: applications and algorithms (3rd edition). California: Duxbury.
- Xing, W.X. and Zhang, J.W. (2000). Parallel machine scheduling with split jobs. Discrete Applied Mathematics, 103, 259-269.
- Yano, C.A., Kim, Y.D. (1991). Algorithms for a class of single machine weighted tardiness and earliness problems, European Journal of Operational Research, 52, 167–178.
- Yi, Y., Wang, D.W. (2003). Soft computing for scheduling with batch setup times and earliness-tardiness penalties on parallel machines. Journal of Intelligent Manufacturing, 14(3-4), 311-322.
- Yu, G., Arguello, M., Song, G., McCowan, S.M. and White A. (2003). A new era for crew recovery at Continental Airlines. Interfaces, 33, 5-22.
- Zhu, Z.W., Heady, R.B. (2000). Minimizing the sum of earliness/tardiness in multi-machine scheduling: a mixed integer programming approach. Computers & Industrial Engineering, 38(2), 297-305.
- Zuo, M., Kuo, W. and McRoberts, K. (1991). Application of mathematical programming to a large-scale agricultural production and distribution system. Journal of the Operational Research Society, 42(8), 639-648.

Publications

- Li, K.P., Sivakumar, A.I., Mathirajan, M. and Ganesan, V.K. (2004). Solution Methodology for Synchronizing Assembly Manufacturing and Air Transportation of Consumer Electronics Supply Chain. International Journal of Business, 9(4), 361-380.
- Li, K.P., Ganesan, V.K. and Sivakumar, A.I. (2005). Synchronized scheduling of Assembly and Multi-Destination Air Transportation in Consumer Electronics Supply Chain. International Journal of Production Research , 43(13), 2671-2685.
- Li, K.P., Ganesan, V.K., Sivakumar, A.I. and Mathirajan, M. (2006). Methodologies for Synchronized Scheduling of Assembly and Air transportation in A Consumer Electronics Supply Chain. International Journal of Logistics Systems and Management, 2(1), 52-67.
- Li, K.P., Sivakumar, A.I. and Mathirajan, M. (2004). A Solution Framework for Synchronizing Logistics and Assembly Manufacturing of Consumer Electronics Supply Chain, Proceedings of International Conference on Responsive Supply Chain and Organizational Competitiveness, Coimbatore, India. (**Best Application Paper Award**)
- Li, K.P., Sivakumar, A.I. and Ganesan, V.K. (2004). Synchronized Scheduling of Assembly and Air Transportation in Consumer Electronics Supply Chain. Proceedings of the 33rd International Conference on Computers and Industrial Engineering, Jeju, Korea.
- Li, K.P., Ganesan, V.K. and Sivakumar, A.I. (2005). Algorithms and Complexities for Synchronized Scheduling of Parallel Machine Assembly and Air Transportation in Consumer Electronics Supply Chain. (Under second review with) European Journal of Operational Research.

Li, K.P., Ganesan, V.K. and Sivakumar, A.I. (2005). Scheduling of Single Stage Assembly with Air Transportation in A Consumer Electronics Supply Chain. (Under review with) Computers & Industrial Engineering.