Université Paris-Dauphine
École doctorale de Dauphine

N° attribué par la bibliothèque

Thèse

Pour l'obtention du grade de
Docteur en informatique

Coordination of production and distribution scheduling

Coordination d'ordonnancement de production et de distribution

Présentée par : **Liangliang FU**

Soutenue le 2 décembre 2014 devant le jury composé de :

Directeur de thèse :

Mohamed Ali ALOULOU	Maître de conférences (HDR), Université Paris-Dauphine

Rapporteurs :

Jean-Charles BILLAUT	Professeur, Ecole Polytechnique de l'Université de Tours
Safia KEDAD-SIDHOUM	Maître de conférences (HDR), Université Pierre et Marie Curie

Examinateurs :

Stéphane DAUZÈRE-PÉRÈS	Professeur, Ecole des Mines de Saint-étienne
Ammar OULAMARA	Professeur, Université de Lorraine
Daniel VANDERPOOTEN	Professeur, Université Paris-Dauphine

Abstract

In this dissertation, we aim at investigating three supply chain scheduling problems in the make-to-order business model. The first problem is a production and interstage distribution scheduling problem in a supply chain with a manufacturer and a third-party logistics (3PL) provider. The second problem is a production and outbound distribution scheduling problem with release dates and deadlines in a supply chain with a manufacturer, a 3PL provider and a customer. The third problem is a production and outbound distribution scheduling problem with setup times and delivery time windows in a supply chain with a manufacturer, a 3PL provider and several customers. For the three problems, we study their individual scheduling problems and coordinated scheduling problems. We propose polynomial-time algorithms or prove the intractability of these problems, and develop exact algorithms or heuristics to solve the NP-hard problems. We establish mechanisms of coordination and evaluate the benefits of coordination.

Keywords: Supply chain scheduling, Coordination, Production and distribution scheduling, Dynamic programming, Branch-and-bound algorithm, Heuristic.

Résumé

Dans cette thèse, nous étudions trois problèmes d'ordonnancement de la chaîne logistique dans le modèle de production la demande. Le premier problème est un problème d'ordonnancement de production et de distribution intermédiaire dans une chaîne logistique avec un producteur et un prestataire logistique. Le deuxième problème est un problème d'ordonnancement de production et de distribution aval avec des dates de début au plus tôt et des dates limites de livraison dans une chaîne logistique avec un producteur, un prestataire logistique et un client. Le troisième problème est un problème d'ordonnancement de production et de distribution aval avec des temps de réglage et des fenêtres de temps de livraison dans une chaîne logistique avec un producteur, un prestataire logistique et plusieurs clients. Pour les trois problèmes, nous étudions les problèmes d'ordonnancement individuels et les problèmes d'ordonnancement coordonnés. Nous proposons des algorithmes polynomiaux ou prouvons la NP-complétude de ces problèmes, et développons des algorithmes exacts ou heuristiques pour résoudre les problèmes NP-difficiles. Nous proposons des mécanismes de coordination et évaluons le bénéfice de la coordination.

Mots clés: Ordonnancement de la chaîne logistique, Coordination, Ordonnancement de production et de distribution, Programmation dynamique, Algorithme B&B, Heuristique.

Acknowledgements

I would like to express my sincere gratitude to my supervisor Mohamed Ali Aloulou for his guidance, technical advice and enthusiastic support throughout my dissertation research. His wonderful patience and friendship encourage constantly my work. I am also extremely grateful to Daniel Vanderpooten for bringing me to the research and cultivating my interest in optimization. I would like to thank the members of jury, Jean-Charles Billaut, Stéphane Dauzère-pérès, Safia Kedad-sidhoum, Ammar Oulamara and Daniel Vanderpooten, for having accepted to evaluate the quality of my dissertation.

Next, my thanks are extended to Alessandro Agnetis, for his help on my research, invitation and especially the Italian dishes. I also want to thank Christian Artigues, Mikhail Y. Kovalyov and Chefi Triki, for their support on my research. A special thank you to François Herbinet for supporting my research on the real industry scheduling problem.

I wish to thank Florian, Renaud, Lyes, Edouard, Amine, Lydia and all other friends and colleagues at LAMSADE, for friendship, support and smile. Your kindness made my three years at Dauphine enjoyable and memorable.

I would like to thank my family with love. The deepest appreciation to my wife for her continued support and encouragement in the difficult periods. A kiss to my son for bringing me happy time everyday. Finally, I want to thank my parents for their patience, support and encouragement all the way.

Contents

Abstract i

Résumé ii

Acknowledgements iii

1 Introduction 1

1.1 Background 1

1.2 Contribution 4

1.3 Organization of the dissertation 6

2 Literature Review 7

2.1 Production scheduling 7

2.2 Distribution scheduling 11

2.3 Integrated production and distribution scheduling 14

2.4 Supply chain scheduling 19

I Production and Interstage Distribution Scheduling 23

3 Individual Production and Interstage Distribution Scheduling Problems 25

3.1 Introduction 25

3.2 Problems and Notations 27

3.3 General properties of optimal delivery schedules 33

3.4 Manufacturer Dominates, 3PL Provider Adjusts - scenario 1 36

3.4.1 Manufacturer's problem 36

3.4.2 3PL provider's problem 37
3.5 3PL Provider Dominates, Manufacturer Adjusts - scenario 2 58
3.5.1 3PL provider's problem 58
3.5.2 Manufacturer's problem 68
3.6 Conclusions 69

4 Coordinated Production and Interstage Distribution Scheduling Problems 71
4.1 Introduction 71
4.2 Problems and Notations 72
4.3 Manufacturer Dominates, 3PL Provider Negotiates - scenario 3 74
4.4 Manufacturer and 3PL Provider Coordinate - scenario 4 76
4.4.1 Mixed Integer Linear Programing 76
4.4.2 Complexity 78
4.4.3 Special cases 81
4.4.4 Mechanism of coordination 84
4.5 Computational Results 85
4.6 Conclusions 91

II Production and Outbound Distribution Scheduling 93

5 Production and Outbound Distribution Scheduling Problems with Release Dates and Deadlines 95
5.1 Introduction 95
5.2 Problems and Notations 96
5.3 Individual Scheduling Problems 101
5.3.1 Manufacturer's Problem 101
5.3.2 3PL Provider's Problem 104
5.4 Coordinated Scheduling Problems 107
5.4.1 SP-NSD Problem and SP-SD Problem 107
5.4.2 NSP-NSD Problem 112
5.5 Computational Results 122

5.6 Conclusions 125

6 Production and Outbound Distribution Scheduling Problems with Setup Times and Delivery Time Windows 127

6.1 Introduction 127

6.2 Problems and Notations 129

6.3 Individual Scheduling Problems 133

6.3.1 Manufacturer's Problem 133

6.3.2 3PL Provider's Problem 134

6.4 Coordinated Scheduling Problems 137

6.4.1 Nonlinear programming model 138

6.4.2 Two-phase iterative heuristic 138

6.5 Computational Results 142

6.6 Conclusions 145

7 Conclusions and Perspectives 147

Bibliography 151

List of Figures

3.1 Production-distribution schedule when manufacturer dominates, 3PL provider adjusts . . . 32
3.2 Production-distribution schedule when 3PL provider dominates, Manufacturer adjusts . . . 33
3.3 Illustration of property 4(a) of Lemma 3.1. . . . 35
3.4 Illustration of Theorem 3.5. . . . 50
3.5 Graphical representation of a delivery schedule . . . 51
3.6 Illustration of property 1 of Lemma 3.8. . . . 62

4.1 Production-distribution schedule when manufacturer dominates, 3PL provider negotiates. . . . 73
4.2 Production-distribution schedule when manufacturer and 3PL provider coordinate . . . 74

5.1 Optimal schedules for the integrated problems . . . 99
5.2 Schedules for the individual problems and the integrated problem . . . 100
5.3 Illustration of property 1 of Lemma 5.3 . . . 108
5.4 Illustration of property 2 of Lemma 5.3 . . . 108
5.5 Illustration of branch-and-bound algorithm B5.1 . . . 117
5.6 An optimal solution for NSP-NSD problem . . . 119

6.1 Optimal production schedule without coordination . . . 132
6.2 Optimal production schedule with coordination . . . 132

List of Tables

3.1 Example for problem 1 32
3.2 Complexity of 3PL provider's problem when manufacturer dominates and 3PL provider adjusts. 38
3.3 Complexity of 3PL provider problems when 3PL provider dominates and manufacturer adjusts. 59

4.1 Average computational times of execution of algorithms. 88
4.2 Failure rates and relative gaps of MILP. 88
4.3 Benefits of coordination in scenario (4) 88
4.4 Price of dominance for 3PL provider. 89
4.5 Benefits of coordination in scenario (4) under various experiment aggregations. 89
4.6 Benefit of relaxing responsiveness constraints in scenario (3). 90

5.1 Example for the integrated problems 99
5.2 Example for evaluation of the benefit of coordination 100
5.3 Example for branch-and-bound algorithm B5.1 116
5.4 Performance of branch-and-bound algorithm B5.1. 123
5.5 Performance of two MILP models. 123
5.6 Gaps of solutions of branch-and-bound algorithm B5.1. 124
5.7 Gaps of solutions of two MILP models. 124

6.1 Average computational times of execution of heuristic 143
6.2 Benefit of coordination 143

Chapter 1

Introduction

1.1 Background

A supply chain involves a set of organizations, including suppliers, manufacturers, logistics providers, distributors and retailers, who work together to satisfy customers' demands. The cost of a product includes the cost of resources at all stages, such as procurement of raw materials, production, distribution of finished products to customers. The objective of supply chain management is to incorporate activities across organizations for adding value, reducing cost and increasing customer service quality. Thomas and Griffin (1996) provided a literature review on supply chain management.

In recent decades, globalization expands supply chain over national boundaries and brings a fierce competition market. In order to satisfy customers' heightened expectations, the enterprises increasingly find that they must rely on effective supply chains. A non-efficient supply chain may carry a high cost. For example, the logistics market volume in Europe accounted in 2012 for 930 billion euros (Kille and Schwemmer 2013). The weight of transportation sector is around 44% of added value and 48% of total employment. According to Eurostat data 2012 (Palmer et al. 2012), third party logistics (3PL) providers fail to consolidate their customers' transport orders: about 24% of all road freight kilometers driven in Europe are empty vehicles and the average vehicle is loaded to 56% of its capacity in terms of weight.

As production and distribution are the main business processes in supply chain, the coordination of production and distribution issue is crucial in supply chain management.

In traditional supply chain, production and distribution are separated by a large intermediate inventory and are planned independently. This independence can simplify decision-making but increases the holding inventory cost. Facing the fierce competition at current internal market and the expectations of customers, many enterprises adopt the make-to-order (a.k.a. assemble-to-order, build-to-order) business model. These enterprises include the ones with highly configured product as automobiles, computers, or with expensive inventory as aircraft. In this context, a product starts to be built after the order is received and there is a small or zero intermediate inventory between production and distribution. Consequently, the coordination of production and distribution is required in this business model. This coordination is also essential in supply chains with time-sensitive products as food, ready-mix concrete paste and newspapers. These products should be delivered to customers immediately or a short time after their production.

In the research literature on supply chain management, coordination issues at the strategic and tactical levels have attracted an extensive research. The issues at the strategic level focus on long-term decision-making, such as allocation of manufacturing equipment, plant opening, selection of distribution centers, etc. Research at the tactical level is targeted at medium-term decision-making, such as planning of production, inventory and distribution in a time period like one year, etc. The issues at the operational level have been investigated during the last decade and are always under developing. They focus on the order-by-order scheduling decision-making, such as machine scheduling, batch delivery, vehicle routing, etc. My thesis addresses the need of research at the operational level pointed out by Thomas and Griffin (1996) .

The coordination model varies with the supply chain models. With the development of the data exchange technology, especially the introduction of enterprise resource planning (ERP) systems and Internet-based collaborative systems, the supply chain can integrate the key business processes for adding value and saving cost. In this integrated supply chain, the involved organizations often belong to one corporation and work in collaborative relationship. In this model, the coordination is controlled by the corporation and the goal is to optimize the performance of the global supply chain. From 1990, some enterprises abandoned the integration and focused on their core competencies and

specialization. They outsource the non-core operations to other enterprises for improving their efficiency. According to the European commission 2011, in 2010, the share of own-account transport is around 15% of the tonne-km generated in road freight transport. This means that transport is mostly outsourced to independent partners like Third Party Logistics (3PL) providers. In this non-integrated supply chain, the independent enterprises have their own objectives and accept the coordination only if they can benefit from it. A negotiation-based mechanism is necessary to motivate the coordination.

The integrated production and distribution scheduling (IPDS) issue, motivated by the integrated supply chain, has been investigated from 1980. This issue investigates the integration of production scheduling decision-making and distribution decision-making at the operational level. Chen (2010) provided an extensive review of the literature on the integrated production and outbound distribution scheduling (IPODS) problems. Outbound distribution deals with a manufacturer shipping his products to the next stage of the supply chain, that typically belongs to another company. As a consequence, the receiving firm may set due dates or deadlines that will constrain the production/distribution problem. The focus of the analysis is on coordinating production decisions (typically, sequencing) and distribution decisions (typically, batching). These two aspects are often conflicting, and require a careful consideration of objectives and roles of the subjects involved. A few articles address the integrated production and interstage distribution scheduling (IPIDS) issues. In most of papers studying IPDS issue, they did not evaluate the benefit of coordination by comparing the integrated solution with the non-coordinated solution. Since the solution of IPDS problems can also be used in the non-integrated supply chain with a compensation mechanism, the IPDS issue is also importance for the non-integrated supply chain.

The term supply chain scheduling was mentioned, by Dawande et al. (2006) , to define the coordination of scheduling decisions at the operational level. Several subproblems are investigated in this respect:

- the individual scheduling problems without coordination, where the decision maker optimizes his individual schedule subject to the constraints imposed by the other decision maker in the supply chain;

- the coordinated scheduling problem, where the decision makers coordinate to decide

jointly their schedules;

- the mechanism of coordination explaining how the decision makers coordinate their activities;
- the evaluation of the benefit of coordination.

Some researches addressing this need have been made in the last decade. For example, Hall and Potts (2003) investigated coordinated scheduling problems between the suppliers and the manufacturers in a three stage supply chain. Dawande et al. (2006) studied the coordination between a manufacturer and a distributor in different bargaining powers scenarios.

1.2 Contribution

In this dissertation, we aim at investigating three supply chain scheduling problems in the make-to-order business model. The research objectives are to:

- study the individual scheduling problems and coordinated scheduling problems:
  - propose polynomial-time algorithms for some polynomial-time solvable problems,
  - prove the intractability for some NP-hard problems,
  - develop exact algorithms or heuristics to solve the NP-hard problems;
- establish mechanisms of coordination;
- evaluate the benefit of coordination.

We consider the following scheduling problems:

***Problem 1: production and interstage distribution scheduling problem* (Agnetis et al. 2014a, 2014b, 2014c)**

In this problem, we consider a supply chain with a manufacturer and a 3PL provider. The manufacturer has to process a set of orders on one machine at upstream and downstream stages. We consider the permutation flow shop environment in production. The 3PL provider is in charge of transportation of semi-finished products from the upstream

stage to the downstream stage. A batch cannot be delivered until all orders of the batch are completed at the upstream stage. Two transportation modes are considered: regular transportation, for which delivery departure times are fixed, and express transportation, for which delivery departure times are flexible. The manufacturer's objective is to minimize the makespan and the 3PL provider's objective is to minimize the transportation cost. We investigate four scenarios: (1) manufacturer dominates, 3PL provider adjusts; (2) 3PL provider dominates, manufacturer adjusts; (3) manufacturer dominates, 3PL provider negotiates; (4) manufacturer and 3PL provider coordinate. For the scheduling problems in each scenario, we provide polynomial-time algorithms or prove their NP-completeness. We provide two mechanisms of coordination for scenarios (3) and (4) and evaluate the benefit of coordination using numerical experiments.

***Problem 2: production and outbound distribution scheduling problem with release dates and deadlines* (Fu et al. 2014)**

In this problem, we consider a supply chain with a manufacturer, a 3PL provider and a customer. The manufacturer has to process a set of orders on one machine, then the 3PL provider delivers them in batches to the customer. Each order has a release date and a delivery deadline fixed by the customer. The manufacturer's objective is to ensure that all orders are delivered before or at their deadline and the 3PL provider's objective is to minimize the transportation cost. We first investigate individual scheduling problems. Then we consider three coordinated scheduling problems with different ways how an order can be produced and delivered: non-splittable production and delivery (NSP-NSD) problem, splittable production and non-splittable delivery (SP-NSD) problem and splittable production and delivery (SP-SD) problem. For these scheduling problems, we provide a polynomial-time algorithm for some restricted versions of SP-NSD and SP-SD problems and a branch-and-bound algorithm for NSP-NSD problem which is NP-hard. We evaluate the performance of branch-and-bound algorithm using numerical experiments.

Problem 3: production and outbound distribution scheduling problem with setup times and delivery time windows

This problem is a real problem proposed by a company working in the packaging industry. We consider a supply chain with a manufacturer, a 3PL provider and several customers.

The manufacturer has to process a set of orders on unrelated parallel machines and splitting of order is allowed in production. A sequence-dependent setup time and a setup cost occur when production changes from one order to another order. Then the 3PL provider delivers orders in batches to the customers with heterogeneous vehicles subject to delivery time windows. The manufacturer's objective is to minimize the total setup cost and the 3PL provider's objective is to minimize the transportation cost. We propose mathematical models for individual scheduling problems and coordinated scheduling problem. We develop a first decomposition approach to solve the coordinated scheduling problem using a commercial solver. we evaluate the feasibility of the approach and the potential benefit of coordination using numerical experiments for small instances. Finally, we propose some directions of improvement for further research.

1.3 Organization of the dissertation

This dissertation is organized as follows. Chapter 2 is dedicated to a literature review on production scheduling, distribution scheduling, integrated production and distribution scheduling, and supply chain scheduling. The three investigated problems are presented in two parts. In part I, we investigate a production and interstage distribution scheduling problem, i.e., the problem 1, which is divided to be presented in chapter 3 and chapter 4. In chapter 3, we study the individual scheduling problems, i.e., scenarios (1) manufacturer dominates, 3PL provider adjusts and (2) 3PL provider dominates, manufacturer adjusts. In chapter 4, we study the coordinated scheduling problems in different scenarios, i.e., scenarios (3) manufacturer dominates, 3PL provider negotiates and (4) manufacturer and 3PL provider coordinate. Then, we evaluate the benefit of coordination using numerical experiments. In part II, we investigate the two production and outbound distribution scheduling problems, i.e., problem 2 and problem 3, which are studied respectively in chapter 5 and chapter 6. Chapter 7 contains conclusions and perspectives of future research.

Chapter 2

Literature Review

In this chapter, we provide a literature review on the following problems: production scheduling, distribution scheduling, integrated production and distribution scheduling, and supply chain scheduling.

2.1 Production scheduling

Production scheduling problem can be presented generally as follows: supposing that a set of orders have to be processed on a set of machines, the problem is how to allocate one or more time intervals for each order to one or more machines while optimizing one or several objective functions.

Graham et al. (1979) introduced a three-field classification $\alpha|\beta|\gamma$ for production scheduling problem, where α, β and γ specify respectively machine environment, order characteristics and optimality criteria. This classification was extended by Brucker (2007).

The machine environment is specified by a string $\alpha = \alpha_1\alpha_2$. α_2 denotes the number of machines. If $\alpha_1 \in \{\circ, P, Q, R, PMPM, QMPM\}$, then each order consists of a single operation. If $\alpha_1 \in \{G, J, F, O, X\}$, then each order consists of a set of operations. If $\alpha_1 \in \{MPT\}$, then each order requires one or more processors at a time. The values are characterized as:

- $\alpha_1 = \circ$: single machine;

- $\alpha_1 = P$: identical parallel machines, i.e., processing speeds of machines are identical;
- $\alpha_1 = Q$: uniform parallel machines, i.e., processing speeds of machines are machine-dependent;
- $\alpha_1 = R$: unrelated parallel machines, i.e., processing speeds of machines are machine-dependent and order-dependent;
- $\alpha_1 = PMPM$: multi-purpose identical parallel machines, i.e., an operation can be processed on any machine equipped with the appropriate tool, and the machines are identical;
- $\alpha_1 = QMPM$: multi-purpose identical parallel machines, i.e., an operation can be processed on any machine equipped with the appropriate tool, and the machines are uniform;
- $\alpha_1 = G$: general shop;
- $\alpha_1 = J$: job shop;
- $\alpha_1 = F$: flow shop;
- $\alpha_1 = O$: open shop;
- $\alpha_1 = X$: mixed shop;

The order characteristics are specified by $\beta \subseteq \{\beta_1, \beta_2, \ldots, \beta_7\}$. The values are characterized as:

- $\beta_1 \in \{pmtn, split\}$, i.e., preemption or splitting of order is allowed, and in the splitting environment the split order can be processed simultaneously on several machines, which is different from the preemption environment;
- $\beta_2 = prec$, i.e., precedence relations exist between the orders;
- $\beta_3 = r_j$, i.e., orders may have different release dates from which their production can start;

- β_4 specifies the restrictions on processing times or number of operations;

- $\beta_5 = \overline{d}_j$, i.e., orders may have different deadlines;

- $\beta_6 \in \{p-batch, s-batch\}$, i.e., orders can be scheduled in batches, the length of a batch is equal to the maximum (sum) of processing times of all orders in the batch for p-batching (s-batching) problem;

- $\beta_7 \in \{s_j, s_{ij}, s_{ijk}\}$, i.e., a setup time occurs when the production changes from a family (or order) to another family (or order), s_j and s_{ij} represent sequence-independent setup time and sequence-dependent setup time respectively, and s_{ijk} represent sequence-dependent and machine-dependent setup time.

The optimality criteria are commonly specified by the total cost objectives depending on completion times of orders, and the customer service quality objectives depending on due dates of orders. The objectives are characterized by two types of functions: the bottleneck objectives, as makespan $C_{\max}$, maximum lateness $L_{\max}$; the sum objectives, as (weighted) total flow time, (weighted) total tardiness, (weighted) total earliness, etc. The objective of scheduling problem is to minimize one or several objective functions.

Production scheduling problem has been extensively investigated from the mid 1950s. In the book of Brucher (2007), he discussed the classical scheduling algorithms for solving single machine scheduling problems, parallel machine scheduling problems, shop scheduling problems, due dates scheduling problems, batching problems, scheduling problems with setup times, multi-purpose machines problems and multiprocessor tasks scheduling problems.

In the following, we focus on the literature of some problems linked to our research: flow shop scheduling problems, single machine scheduling problems with release dates, and scheduling problems with setup times.

In flow shop scheduling problems, each order has to be processed in a fixed sequence of machines, i.e., the first operation of each order is performed on the first machine, the second operation on the second machine, and so on. For regular objective functions, i.e., functions that are non-decreasing in completion times of orders, the problem is to find a processing sequence of orders for each machine. We focus on some flow shop scheduling

problems with makespan objective function $C_{\max}$. Johnson (1954) provided a polynomial-time algorithm to solve the 2-machine flow shop scheduling problem $F2||C_{\max}$. Garey et al. (1976) proved the NP-hardness of the m-machine flow shop scheduling problem $Fm||C_{\max}$ with $m \geq 3$.

Concerning the single machine scheduling problems with release dates, we focus on the problems with maximum lateness $L_{\max}$ objective function. The problem without release dates $1||L_{\max}$ can be solved by Jackson's earliest due date (EDD) rule introduced by Jackson (1955). This problem is a special case of the problem $1|prec|L_{\max}$ solved by a polynomial-time algorithm provided by Lawler (1973). The problem with release dates and preemption $1|r_j, pmtn|L_{\max}$ can be solved by Jackson's preemptive earliest due date (EDD-preemptive) rule introduced by Jackson (1955). This problem is a special case of the problem $1|prec, r_j, pmtn|L_{\max}$ solved by a polynomial-time algorithm provided by Baker et al. (1983). Lenstra et al. (1977) proved the NP-hardness of the problem $1|r_j|L_{\max}$. Carlier (1982) provided the first efficient branch-and-bound algorithm to solve this problem.

Allahverdi et al. (2008) provided a survey of scheduling problems with setup times or costs. They discussed the problems with sequence-independent setup time (s_j) or sequence-dependent setup time (s_{ij}). s_j depends upon only order j processed after the changeover, while s_{ij} depends upon both orders i and j processed before and after the changeover respectively. We focus on the problems with the objective of minimizing makespan $C_{\max}$ or sum of setup cost. The single machine problem $1|s_j|C_{\max}$ is polynomial-time solvable. The single machine problem $1|s_{ij}|C_{\max}$ is NP-hard (Bruno and Downey 1978) and can be reformulated as a Traveling Salesman Problem (TSP). The two parallel machine problem with unit processing times and unit setup times $P2|p_j = 1; s_j = 1|C_{\max}$ is NP-hard (Brucker et al. 1998). The special case of this problem where all families have equal sizes can be solved in polynomial time (Brucker 2007). Several heuristics and meta-heuristics were provided for the parallel machine problem $P2|s_{ij}|C_{\max}$: a divide and merge heuristic by Gendreau et al. (2001), a heuristic and a tabu search algorithm by Mendes et al. (2002), and a hybrid meta-heuristic by Behnamian et al. (2009). For the same problem with splitting of order, in that orders can be split and processed simultaneously on different machines, some heuristics were

provided by Tahar et al. (2006) and Yalaoui and Chu (2003). Concerning the problems with the sum of setup cost objective function, few papers have investigated this problem. Miller et al. (1999) provided a hybrid genetic algorithm for a single machine scheduling problem with sequence-dependent setup time minimizing the sum of setup cost, inventory cost, and backlog cost. Vignier et al. (1999) considered a parallel machines scheduling problem with sequence-dependent setup time, release dates and deadline. The objective function is to first find a feasible schedule and then to minimize the cost due to assignment and setup time costs. They proposed a hybrid method that consists of an iterative heuristic, a genetic algorithm, and a branch-and-bound algorithm. Anglani et al. (2005) proposed a fuzzy mathematical programming approach to solve a parallel machines scheduling problem with sequence-dependent setup time, uncertain processing time and the objective of minimizing the total setup costs.

Some other scheduling problems with setup times have been investigated, such as flow shop scheduling problems with setup times (Cheng et al. 1999, Brucker et al. 2005, etc.), open shop scheduling problems with setup times(Averbakh et al. 2005, Billaut et al. 2008, etc.), job shop scheduling problems with setup times (Cheung and Zhou 2001, Artigues and Roubellat 2002, etc.).

2.2 Distribution scheduling

Distribution scheduling problem is a central problem in distribution management and is faced by a lot of enterprises every day. There are three types of delivery (Chen 2010): individual delivery, i.e., each order is shipped individually; direct batch delivery, i.e., different orders of a customer can be delivered together in a shipment; routing batch delivery, also named as vehicle routing delivery, i.e., orders of different customers can be delivered together in a shipment. The individual delivery and direct batch delivery are used to deliver the time-sensitive products. The vehicle routing delivery is extensively adopted by many enterprises to reduce distribution cost. In the literature on distribution management, the vehicle routing problem has attracted an extensive research since the first study provided by Dantzig and Ramser (1959). In fact, the first two types of delivery can be seen as two special cases of the vehicle routing delivery. There are few

articles investigating only the distribution scheduling problem with the first two types of delivery. In many cases, these problems are simple and have been discussed in the integrated production and distribution scheduling problems (Chen 2010). However, it is interesting to investigate these problems with some adding characteristics, such as a limited number of vehicles, vehicles with fixed departure dates, heterogeneous vehicles, release dates, delivery deadlines, etc. In this section, we focus on vehicle routing problem.

The classical vehicle routing problem (VRP) is to determine a set of routes for a fleet of vehicles, each of which starts and ends at its own depot, to serve a set of customers on minimizing the total travel cost subject to a set of constraints. The VRP is one of the most popular combinatorial optimization problems and is NP-hard because it generates the traveling salesman problem (TSP) (Dantzig and Ramser 1959). A lot of exact algorithms and heuristics are provided to solve the VRP. Toth and Vigo (2002) surveyed the variants of the VRP. The capacitated vehicle routing problem (CVRP) considers the vehicle capacity. In the distance-constrained vehicle routing problem (DVRP), the length of each each route cannot exceed a preset limit. The capacitated distance-constrained vehicle routing problem (DCVRP) considers both the vehicle capacity and the constraint of distance. In the vehicle routing problem with time windows (VRPTW), the service at each customer must start within a given time window and the arrival of vehicles after time windows are prohibited. In the vehicle routing problem with backhauls (VRPB), each customer location may act as a pickup or a delivery node and all deliveries must be performed before any pickup. In the vehicle routing problem with pickup and delivery (VRPPD), the passengers or goods are transported between pickup and delivery locations. All the above problems are NP-hard because they generate the classical VRP.

Toth and Vigo (2002) provided a survey of exact algorithms and heuristics for all the above vehicle routing problems. Similar surveys were provided in chapter 6 (Cordeau et al. 2007) and chapter 7 (Cordeau et al. 2007) of the handbook of operations research and management science. Other surveys have been provided for one or some of the above problems, such as VRPTW by Bräysy and Gendreau (2005), VRPB and VRPPD by Parragh et al. (2008), large-scale VRPTW by Gendreau and Tarantilis (2010), CVRP and VRPTW by Kumar and Panneerselvam (2012). There are other variations of the VRP, such as the dynamic vehicle routing problem (DVRP) where part or all of the

input is unknown before the start of working day, the heterogeneous fleet vehicle routing problem (HVRP) where a fleet of vehicles is characterized by different capacities and costs, and the split delivery vehicle routing problem (SDVRP). Surveys on DVRP, HVRP and SDVRP were provided respectively by Pillac et al. (2013), Baldacci et al. (2008) and Archetti and Speranza (2008).

In the following, we focus on the exact algorithms and heuristics for the VRPTW. The VRPTW can be defined on a directed graph and formulated as a multicommodity network model with time windows and capacity constraints (Desrochers et al. 1988). There are four main exact algorithms for the VRPTW: Lagrangian relaxation based branch-and-bound algorithm where the lower bound is obtained by Lagrangian relaxation (Fishier 1994, Fishier et al. 1997, Kohl and Madsen 1997, Kallehauge et al. 2006); column generation based branch-and-bound algorithm where the linear relaxations are solved by column generation (Desrochers et al. 1992, Kohl et al. 1999, Cook and Rich 1999); branch-and-cut algorithm where the upper bound is obtained by a greedy randomized adaptive search procedure (Bard et al. 2002); genetic and set partitioning two-phase approach (Alvarenga et al. 2007). Because of the NP-hardness of the VRPTW, the research has concentrated on heuristics, such as construction heuristics where at a time one customer is inserted into partial routes until a feasible solution is obtained (Solomon 1987, Potvin and Rousseau 1993, Ioannou et al. 2001, Nagata and Bräysy 2009, Pang 2011), improvement heuristics where a feasible solution is improved iteratively with an exchange mechanism (Russell 1977, Baker and Schaffer 1986, Potvin and Rousseau 1995, etc.), combination of construction and improvement heuristics (Russell 1995, Cordone and Wolfler Calvo 2001, Bräysy 2002), tabu search heuristics (Taillard et al. 1997, Chiang and Russell 1997, Cordeau et al. 2001, Lau et al. 2003, etc.), genetic algorithms (Gehring and Homberger 2002, Berger et al. 2003, Mester and Bräysy 2005, etc.), two-phase greedy randomized adaptive search procedure (Kontoravdis and Bard 1995), guided local search algorithm (Kilby et al. 1999), ant colony optimization algorithm (Gambardella et al. 1999), four-phase metaheuristic (Bräysy 2003), two-stage hybrid algorithm (Bent and Van Hentenryck 2004), improved multi-objective evolutionary algorithm (Garcia-Najera and Bullinaria 2011), etc.

2.3 Integrated production and distribution scheduling

In the literature, two types of integrated production and distribution scheduling problems have been investigated: the integrated production and interstage distribution scheduling (IPIDS) problem involving the distribution of orders between manufacturing stages in the shop production, such as flow shop, job shop, open shop, etc; the integrated production and outbound distribution scheduling (IPODS) problem involving the distribution of orders to the customers.

In the IPIDS problem, the production and transportation of orders between the stages are taken into account. The problem can be specified by the characteristics of machine environment, i.e., flow shop, job shop, open shop and so on, by the characteristics of transportation time, i.e., order-dependent, order-independent, machine-dependent, machine-independent and constant, by the characteristics of transporters, i.e., sufficient number, limited number, unlimited capacity and limited capacity. Brucker et al. (2004) provided a survey on the IPIDS problem in the flow shop and open shop environments. There are other surveys in the literature on the IPIDS, such as Hurinka and Knustb (2001), and Lee and Chen (2001).

We focus on the IPIDS problems with order-independent transportation time in 2-machine flow shop. Here, we use the three-field classification $\alpha|\beta|\gamma$ of production scheduling problem for the IPIDS problems. Maggu and Das (1980) developed a polynomial time algorithm based on well-known Johnson's rule (Johnson 1954) for the problem of minimizing the makespan subject to an unlimited number of transporters with order-dependent transportation time, denoted by $F2|t_j, v \geq n|C_{\max}$, where t_j, v and n represent respectively order-dependent transportation time, number of transporters and number of orders. Since the problem with order-independent transportation time can be seen as a special case of the problem with order-dependent transportation time, the similar problem with order-independent transportation time $F2|t_j \in \{t_1, t_2\}, v \geq n|C_{\max}$ can be solved by the same algorithm, where t_1 represents the transportation time from the first stage to the second while t_2 represents the returning time. The problem minimizing the makespan subject to one transporter with capacity of one and order-independent

transportation time, denoted by $F2|t_j \in \{t_1, t_2\}, v = 1, c = 1|C_{\max}$ was proved to be strongly NP-hard even if $t_1 = t_2$ (Hurinka and Knustb 2001). Tang et al. (2010) developed an approximation algorithm with worst case ratio of 2 for this problem. Lee and Chen (2001) proved the NP-hardness of a similar problem with a vehicle capacity more than 2, denoted by $F2|t_j \in \{t_1, t_2\}, v = 1, c \geq 3|C_{\max}$, and provided a polynomial-time dynamic programming algorithm for the special case with equal processing times on one machine and a fixed number of vehicles. Lee and Strusevich (2005) proved that for the problem $F2|t_j \in \{t_1, t_2\}, v = 1, c \geq n|C_{\max}$, finding the best schedule in class $SF(2)$ of schedules with at most two shipments is NP-hard even if $t_1 = t_2$, and provided an approximation algorithm with worst case ratio of 3/2, which is the best possible algorithm in the class of heuristics that construct schedules with at most two shipments. Gong and Tang (2011) developed an approximation algorithm with worst case ratio of 2 for the problem $F2|t_j \in \{t_1, t_2\}, v = 1, c \geq n|C_{\max}$ and an approximation algorithm with worst case ratio of 7/3 for a similar problem with the orders having different sizes of physical storage space in the transporter. There are also other related problems with buffer space constraints studied by Stern and Vitner (1990), Panwalkar (1991).

While the above articles considered transportation capacity and transportation time, few articles consider transportation cost. As mentioned in previous chapter, in a global supply chain, a product can be processed at different plants located at different geographic locations. So the transportation cost is not negligible. Aloulou et al. (2014) considered a bicriteria 2-machine flow shop serial-batching scheduling problem with a sufficient number of transporters with limited capacity. They developed two approximation algorithms and provided polynomial-time algorithms for some special cases. They considered two criteria: number of production batches and makespan. This problem is equivalent to the problem $F2|t_j \in \{t_1, t_2\}, v \geq n, c \geq 1|TC, C_{\max}$, where TC represents the trip-based transportation cost, where the cost of one delivery batch is order-independent.

Chen (2010) surveyed the IPODS problems and introduced a five-field notation, $\alpha|\beta|\pi|\delta|\gamma$, to represent the IPODS models. α, β and γ specify respectively the machine environment, the order characteristics and the optimality criterion as the classical three-field classification (Graham et al. 1979, Pinedo 2002). Some new objective func-

tions of transportation are introduced, such as maximum delivery time denoted by $D_{\max}$, total trip-based transportation cost denoted by TC, etc. π and δ specify respectively the characteristics of delivery process and the number of customers. The number of customers is specified by one value of $\{1, k, n\}$, where $\delta = k \geq 2$ means there are multiple customers, and $\delta = n$ means that each order belongs to a different customer. The characteristics of delivery process include vehicle characteristics (number and capacity of vehicles) and delivery methods. The vehicle characteristics are specified by $V(x, y)$, where:

- $x \in \{1, v, \infty\}$ represents the number of vehicles.
- $y \in \{1, c, \infty, Q\}$ represents the capacity of vehicles. $\{1, c, \infty\}$ and Q distinguish, respectively, the possible capacities of vehicles when orders have equal size, and the limited capacity of vehicles when orders have general size.

The delivery methods include:

- iid: individual and immediate delivery.
- direct: direct batch delivery.
- routing: batch routing delivery.
- fdep: shipping with fixed delivery departure dates.
- split: splittable delivery, i.e., an order can be split and delivered by several vehicles.

The IPODS problems are classified by delivery methods: (i) models with individual and immediate delivery; (ii) models with direct batch delivery to a single customer; (iii) models with batch routing delivery to multiple customers; (iv) models with batch routing delivery to multiple customers; (v) models with fixed delivery departure dates. Chen (2010) surveyed the algorithms, heuristics, and complexity for the IPODS problems in each class.

We focus on the IPODS problems related to our considered problems: the IPODS problems with release dates r_j; the IPODS problems with maximum lateness $L_{\max}$ or delivery deadline $\overline{d}_j$, and transportation cost TC; the IPODS problems with setup times s_j, s_{ij}; the IPODS problems with routing delivery and time windows.

IPODS problems with release dates: The research of the IPODS problems with release dates concentrates on the models with individual and immediate delivery, and direct delivery. As proved by Chen (2010), the problems with individual and immediate delivery, (i) $1|r_j|V(\infty,1),iid|n|D_{\max}$, (ii) $1|r_j,prec|V(\infty,1),iid|n|\ D_{\max}$, (iii) $Pm|r_j|V(\infty,1),iid|n|D_{\max}$, (iv) $Fm|r_j|V(\infty,1),iid|n|D_{\max}$ are strongly NP-hard. Liu and Cheng (2002) proved the NP-hardness of the problem (v) $1|r_j,s_j,pmtn|V(\infty,1),iid|n|D_{\max}$. In these problems, the orders are delivered individually and immediately to the customers upon their completion while minimizing the maximum delivery time. For problems (i), (ii), (iii) and (v), approximation algorithms and (or) polynomial-time approximation schemes were provided by Potts (1980), Hall and Shmoys (1989, 1992), Mastrolilli (2003), Zdrzalka (1994), Liu and Cheng (2002). Gharbi and Haouari (2002) developed a branch-and-bound algorithm for problem (iii). Kaminsky (2003) proposed an asymptotic optimality of the longest delivery time algorithm for problem (iv). Few articles consider direct or routing delivery. Lu et al. (2008) provided a polynomial-time algorithm for the problem $1|r_j,pmtn|V(1,c),direct|1|D_{\max}$. For the problem $1|r_j|V(1,c),direct|1|D_{\max}$ they proved its NP-hardness and proposed an approximation algorithm with worst case ratio of 5/3. Mazdeh et al. (2008) provided a branch-and-bound algorithm for a special case of the NP-hard problem $1|r_j|V(\infty,\infty),direct|n|\sum F_j+TC$, where $\sum F_j$ represents the total flow time. Mazdeh et al. (2012) provided a branch-and-bound algorithm for a special case of the similar problem with sum of weighted flow time, $1|r_j|V(\infty,\infty),direct|n|\sum w_jF_j+TC$. Selvarajah et al. (2013) provided an evolutionary meta-heuristic for the same problem in the general case and a polynomial-time algorithm for the special case with common weight and preemption in production, $1|r_j,pmtn|V(\infty,\infty),direct|n|\sum wF_j+TC$ and $1|r_j,pmtn|\ V(\infty,\infty),direct|n|\sum wC_j+TC$. There are some articles considering the on-line problem, i.e., the information related to an order becomes known when this order is released. Ng and Lu (2012) investigated the problems of Lu et al. (2008) in on-line environment. Averbakh and Xue (2007) provided an on-line two-competitive algorithm for the on-line problem $1|r_j,pmtn|V(\infty,\infty),direct|k|\sum D_j+TC$, where $\sum D_j$ represents the total delivery time.

IPODS problems with $L_{\max}$ and TC: Some IPODS problems with maximum lateness $L_{\max}$ or delivery deadline $\overline{d}_j$, and transportation cost TC have been investigated in the literature. Polynomial-time algorithms were provided for the problems $1||V(\infty,\infty),direct|k|L_{\max}+TC$ with a fixed k by Hall and Potts (2003), $1||V(\infty,c),direct|k|L_{\max}+TC$ with a fixed k by Pundoor and Chen (2005), $1||V(1,\infty)$, $direct|1|L_{\max}+TC$ by Hall and Potts (2005), $1||V(v,\infty),direct|1|L_{\max}+TC$ with a fixed v and $1||V(1,\infty),routing|k|L_{\max}+TC$ with a fixed k by Chen (2010), $1|pmtn,\overline{d}_j|V(\infty,Q),direct|1|TC$ by Chen and Pundoor (2009). Wang and Lee (2005) proved the NP-hardness of the problem $1|\overline{d}_j|V(\infty,1),iid|n|TC$ with two types of vehicles and provided a pseudo-polynomial time dynamic programming algorithm for a special case. Chen and Pundoor (2009) proved the NP-hardness of the problems without preemption of production, $1|\overline{d}_j|V(\infty,Q)|1|TC$ and $1|\overline{d}_j|V(\infty,Q),direct|1|TC$, and provided approximation algorithms with worst-case ratio of 2.

IPODS problems with setup times: Few articles investigated the IPODS problem with sequence-independent setup times (s_j) or sequence-dependent setup times (s_{ij}). The problem with sequence-independent setup time and individual and immediate delivery, $1|s_j|V(\infty,1),iid|n|\ D_{\max}$, is strongly NP-hard (Chen 2010). Zdrzalka (1991) provided an approximation algorithm with worst case ratio of 5/3 for the problem in the case with unit setup times, and Zdrzalka (1995) provided an approximation algorithm with worst case ratio of 3/2 for the problem in the case with general setup times. Woeginger (1998) developed a polynomial-time approximation scheme for this problem. Liu and Cheng (2002) proved the NP-hardness and provided a polynomial-time approximation scheme for the similar problem with release dates and preemption of production, $1|r_j,s_j,pmtn|V(\infty,\ 1),iid|n|D_{\max}$. Zdrzalka (1994) provided an approximation with worst case ratio of 3/2 for this problem. Van Buer, et al. (1999) proved the NP-hardness and provided a heuristic for the problem with sequence-dependent setup time and routing delivery, $1|s_{ij},\overline{d}_j|V(\infty,Q),routing|n|TC+VC$, where TC and VC represent respectively the trip-based transportation cost and the vehicle-based transportation cost. Wang and Cheng (2009) provided three heuristics for the NP-hard problem with sequence-dependent setup times and direct delivery, $P2|s_j|V(\infty,c),direct|k|D_{\max}+TC$.

IPODS problems with routing delivery and time windows: While the VRPTW has been well studied, few articles investigated the IPODS problem with routing delivery and time windows. Ullrich (2013) investigated the problem $Pm, r_m|[a_j, b_j]|V(v, Q), routing|k|\sum T_j$, i.e., a set of orders of general size is processed on parallel machines subject to the machine release time r_m, and delivered to customers within the time windows $[a_j, b_j]$ by a fleet of heterogeneous vehicles on minimizing the sum of tardiness $\sum T_j$. They provided a genetic algorithm for the integrated problem and evaluated its performance by comparing with two classical decomposition approaches. Low et al. (2014) provided an integer nonlinear programming model and two adaptive genetic algorithms for the problem $1|[a_j, b_j]|V(\infty, Q), routing|n|TC + \sum E_j + \sum T_j$, where retailers' orders are processed in a distribution center and delivered to customers by a fleet of heterogeneous vehicles within the soft time windows, i.e., the violation of time windows incurs a penalty. The objective is to minimize the total cost including the transportation cost, the penalty cost of earliness $\sum E_j$ and the penalty cost of tardiness $\sum T_j$. Low et al. (2013) investigated the problem with the same model on minimizing the time required to complete producing the product, delivering it to retailers and returning to the distribution center. Chen (2009) investigated an IPODS problem with routing delivery and time windows for perishable food products, which cannot be represented by the five-field notation of Chen (2010). In this problem, the products of each delivery batch are produced continuously on a single machine and are delivered to customers within soft time windows. The orders are assumed stochastic and the deterioration of products throughout their lifetime is considered. The objective is to maximize the expected total profit of the supplier. He proposed an algorithm composed of the constrained Nelder-Mead method (Nelder and Mead 1965) and a heuristic for the VRPTW.

2.4 Supply chain scheduling

As mentioned in the previous chapter, the supply chain scheduling problem focuses on the coordination of the scheduling decisions (Dawande et al. 2006). Generally, the following problems are considered: the individual scheduling problems at the conflict models, the

coordinated scheduling problem and the mechanisms of coordination, and the evaluation of the benefit of coordination. Since it is difficult to consider the coordination in the whole supply chain, the coordination between two or three stages has been considered in several articles.

Hall and Potts (2003) were the first to study supply chain scheduling problems. In their model, a three-stage supply chain is formed by suppliers, manufacturers, and customers. A supplier makes deliveries to several manufacturers who also make deliveries to customers. The problems in each stage have been studied. From the viewpoint of the supplier, the problem is to minimize the sum of production and delivery costs, where three production costs are considered respectively: sum of flow times, maximum lateness and number of late orders. From the viewpoint of the one manufacturer, the problem is to minimize the sum of production and delivery costs subject to the release dates of orders imposed by the supplier. The coordinated scheduling problem is to minimize the overall cost. The authors proposed polynomial-time dynamic programming algorithms or proved the NP-completeness for the supplier's scheduling problem, special cases of the manufacturer's scheduling problems and special cases of the coordinated scheduling problems. They also provided mechanisms of coordination and evaluated the benefit of coordination for two examples.

Agnetis et al. (2006) studied a coordinated scheduling problem between a supplier and several manufacturers, taking into consideration an intermediate storage buffer. They considered the inventory cost and the interchange cost representing the distances between the actual schedules at the various stages of supply chain and their respective ideal schedules. They investigated the individual scheduling problems and a special case of the coordinated problem. They provided a polynomial-time algorithm for each problem minimizing the interchange cost or both costs.

Dawande et al. (2006) analyzed the conflict and coordination issues between a manufacturer and a distributor. In their considered two distribution systems, a manufacturer makes products which are delivered to customers by a distributor. In the first system, the manufacturer focuses on minimizing the makespan and the distributor minimizes the maximum lateness. In the second system, the manufacturer focuses on minimizing the total setup cost and the distributor minimizes the inventory cost. They introduced the

cost of conflict, which is the additional cost when the other decision maker imposes his optimal schedule. They provided polynomial-time algorithms for the individual scheduling problems when one decision maker dominates and imposes the requirement to the other decision maker. They developed a polynomial-time algorithm for the coordinated scheduling problem. They also proposed mechanisms of coordination and evaluated the benefit of coordination.

Chen and Hall (2007) studied the conflict and coordination issues in an assembly system, where several suppliers deliver the parts of orders to a manufacturer. They considered two objectives, the makespan and the maximum lateness, for each decision maker. They evaluated the cost of conflict. They investigate the problems in four scenarios: manufacturer dominates, suppliers adjust; suppliers dominate, manufacturer negotiates; manufacturer dominates, suppliers negotiate; manufacturer and suppliers coordinate. They provided either a polynomial-time algorithm or a proof of intractability for the scheduling problems in the above scenarios, and developed heuristics for NP-hard problems. They provided mechanisms of coordination and evaluated the benefit of coordination.

Hall and Liu (2010) studied a coordinated scheduling problem between a manufacturer and several distributors. The manufacturer allocates the capacity of production to satisfy all or a set of orders among the distributors. The distributors may share their allocated capacity among themselves before submitting revised orders. Finally, the manufacturer schedules the revised orders to minimize his cost. They considered three mechanisms of coordination: the manufacturer considers the scheduling costs and constraints in making capacity and order allocation decisions; the distributors share their allocated capacity; the manufacturer and the distributors coordinate. They provided optimal algorithms for the scheduling problems and mechanisms of coordination. They evaluated the benefit of coordination.

Aydinliyim and Vairaktarakis (2010) studied a coordinated scheduling problem in a supply chain where several manufacturers outsource certain operations to a single third part. The objective of each manufacturer is to minimize the total cost consisting of the booking cost and the sum of weighted flow time cost. The objective of the third part is to minimize the total cost incurred by all manufacturers, and develop a saving

sharing scheme which motivates the coordination. They proved that the scheduling problem of each manufacturer and the third part centralized scheduling problem are NP-hard, for which they provided three heuristics. They presented a savings-sharing scheme of coordination and evaluated the benefit of coordination. Cai and Vairaktarakis (2012) investigated the same problem with the objective of minimizing the booking, overtime, and tardiness costs. They designed a truth-telling mechanism of coordination and evaluated the benefit of coordination.

Part I

Production and Interstage Distribution Scheduling

Chapter 3

Individual Production and Interstage Distribution Scheduling Problems

3.1 Introduction

In part I, we consider a production and interstage distribution scheduling problem in a permutation flow shop environment. A set of orders are processed by a manufacturer at the upstream facility, and delivered to the downstream facility belonging to the same manufacturer. The distribution is outsourced to a third-party logistics (3PL) provider. The manufacturer may impose his requirement to the 3PL provider, for example, each order should be delivered within a certain time T from its release at the upstream stage. Small values of T indicate high responsiveness of the 3PL provider, which is desirable for the manufacturer. However, this may entail higher costs for the 3PL provider. The 3PL provider may impose his delivery schedule to the manufacturer, for example, the delivery schedule fixes the number of available vehicles for some fixed departure dates. In order to decease the transportation cost, the 3PL provider decreases the frequency of delivery and offers the vehicles of large capacity. However, large batch deliveries may increase the makespan at the downstream stage. These conflicts motivate the coordination between the production scheduling and the interstage distribution scheduling.

In the literature, it is commonly assumed that all orders are produced at a single

plant. However, in our globalized world, many firms have now multiple plants spread over different countries. Even national firms adopt a multiple plants scheme to reduce production costs and to expand their production capacity. Therefore, in a global supply chain, a product can be processed at different plants located at different geographic locations. So there is a need to analyze the process of planning, scheduling, and distribution in the multiple plants environment.

Another feature considered in our model is the presence of multiple transportation resources. As observed by Chen (2010) in his survey on integrated production and distribution scheduling problems, research has not yet adequately addressed problems with *fixed delivery departure times*. These refer to the fact that transportation resources are available to depart from the upstream stage only at fixed, known departure times. While a handful of papers (Li et al. 2005, 2006; Stecke and Zhao 2007; Wang et al. 2005) address coordination between production and outbound distribution with fixed delivery departure times, we are not aware of articles on production and interstage distribution scheduling problems with fixed delivery departure times. In this problem, we consider a general setting in which transportation can be provided by means of resources with fixed departure times as well as *dedicated* resources which are always available. We will refer to the two transportation modes above as *regular* and *express* respectively. In general, deciding which transportation resource is the most appropriate to perform each delivery is part of the 3PL provider decision problem. Koc et al. (2013) considered the similar transportation setting for an integrated production and outbound delivery planning problem. In our problem, the cases with only one transportation mode and with both transportation modes are considered.

We investigate four scenarios specified by the bargaining powers and the relationship of decision makers: (1) manufacturer dominates, 3PL provider adjusts; (2) 3PL provider dominates, manufacturer adjusts; (3) manufacturer dominates, 3PL provider negotiates; (4) manufacturer and 3PL provider coordinate. For the scheduling problems in each scenario, we provide polynomial-time algorithms or prove their NP-hardness. We provide two mechanisms of coordination for scenarios (3) and (4). We evaluate the benefit of coordination using numerical experiments. The most related research was provided by Dawande et al. (2006). They analyzed the conflict and coordination issues

between a manufacturer and a distributor. In their considered two distribution systems, a manufacturer makes products which are delivered to customers by a distributor. In the first system, the manufacturer focuses on minimizing the makespan and the distributor minimizes the maximum lateness. In the second system, the manufacturer focuses on minimizing the total setup cost and the distributor minimizes the inventory cost. They introduced the cost of conflict, which is the additional cost when the other decision maker imposes his optimal schedule. They provided polynomial-time algorithms for the individual scheduling problems when one decision maker dominates and imposes the requirement to the other decision maker. They developed a polynomial-time algorithm for the coordinated scheduling problem. They also proposed mechanisms of coordination and evaluated the benefit of coordination. The literature of the IPIDS (integrated production and interstage distribution scheduling) problem can be found in section 2.3 of chapter 2.

In chapter 3, we study the individual scheduling problems, i.e. scenarios (1) and (2). In section 3.2, we formally describe the problems and introduce notations and terminology. Section 3.3 illustrates some general properties of the optimal delivery schedules which are common to scenarios (1) and (2). Section 3.4 is devoted to scenario (1), section 3.5 to scenario (2). Section 3.6 contains some conclusions and perspectives. Chapter 4 is dedicated to scenarios (3) and (4), to the mechanisms of coordination, and to the evaluation of the benefit of coordination.

3.2 Problems and Notations

In this section, we formally define the problems addressed in this chapter. We refer to the upstream and downstream stages of a supply chain as machine M_1 and M_2 respectively. The context is specified by the following points.

- A manufacturer has to process a set of n orders, $1, \ldots, n$. Each order $j = 1, \ldots, n$ is first processed on machine M_1, with processing time p_j^1, then on machine M_2, with processing time p_j^2.

- We consider the permutation flow shop context, i.e. the orders are processed within the same sequence on both machines.

- After processing on M_1, the orders have to be shipped from M_1 to M_2 (typically, located in a different city) for further processing.

- Denote by C_j^i the completion time of order $j = 1, \ldots, n$, on machine $M_i, i = 1, 2$. In particular, C_j^1 is the release time of order j from the upstream stage. The quantities C_j^i specify a *production schedule*. When it is necessary to denote a production schedule as σ, the related quantities will be indicated as $C_j^1(\sigma)$. We denote by $\sigma(j)$ the j-th order in the production schedule σ.

- A 3PL provider is in charge of transportation of semi-finished product from M_1 to M_2.

- Transportation must comply with a certain *responsiveness* requirement, i.e., each order j has to be delivered to M_2 within a time T from its completion on M_1. In other words, for each order j, a *deadline* $\overline{d}_j = C_j^1 + T$ is specified. The uniform setting of T can be adopted in practice for homogeneous semi-finished products. We let D_j denote the actual *delivery time* of order j to M_2, so in a feasible solution $D_j \leq \overline{d}_j$ for all j. This responsiveness requirement can be relaxed in the scenarios with coordination.

- The vehicles used by the 3PL provider have a certain *capacity*, given by the maximum number of orders a vehicle can carry (i.e., orders are supposed to have equal size).

- The 3PL provider uses two *transportation modes*.

  - Type-1 transportation is *regular transportation*. There are V_1 identical vehicles of capacity $c_1 \leq n$ each of which has a *fixed departure time*. The transportation time from M_1 to M_2 is denoted by ℓ_1. We assume that each vehicle can be used for at most one trip. So we ignore the return of vehicle. The transportation cost per delivery is denoted by h_1. We assume we have L fixed delivery departure times $t_1 < \ldots < t_L$, $v_s \leq n$ vehicles for departure time t_s, $s = 1, \ldots, L$, and consequently $V_1 \leq Ln$.

  - Type-2 transportation is *express transportation*. There are V_2 identical vehicles of capacity $c_2 \leq n$, each of which can depart at any time, i.e., there are no

fixed departure times. The transportation time from M_1 to M_2 is denoted by ℓ_2 and the return time by ℓ_2'. The transportation cost per delivery is denoted by h_2.

For both modes, the transportation cost of one shipment depends on the assigned resource by the 3PL provider and does not depend on the quantity of products delivered.

Clearly, if either $V_1 = 0$ or $V_2 = 0$, only one transportation mode exists.

- For both transportation modes, orders in the same delivery form a *batch*. A batch B is available for delivery when all orders belonging to B are completed at M_1. The *delivery time* of a batch B, denoted by D_B, is the time at which the batch reaches M_2. We let a *regular batch* or an *express batch* be a batch for which regular or express transportation, respectively, is used.

- Given a production schedule, a *delivery schedule* θ is a partition of the orders into b batches, along with the specification of a transportation mode and the departure time for each batch.

- A *production-distribution schedule* is specified by a production schedule σ and a delivery schedule θ. We indicate a production-distribution schedule by the pair (σ, θ). Given a production-distribution schedule, we let $B(j)$ denote the batch of order j. The delivery time of an order j is therefore denoted by $D_{B(j)}$.

- In all scenarios, the objective of the 3PL provider is to minimize the transportation cost $TC = h_1 n_1 + h_2 n_2$, where n_1 and n_2 are the number of batches of type 1 (regular) and type 2 (express), respectively. If only one transportation mode is present, the objective is to minimize the number of batches. The objective of the manufacturer is to minimize the makespan $C_{\max} = \max_{j=1,\ldots,n} C_j^2$. The values of makespan $C_{\max}$ and transportation costs TC associated with the production-distribution schedule (σ, θ) are denoted as $C_{\max}(\sigma, \theta)$ and $TC(\sigma, \theta)$ respectively.

Scenarios In this chapter we consider two *scenarios*: (1) manufacturer dominates, 3PL provider adjusts; (2) 3PL provider dominates, manufacturer adjusts. These are formally defined in the following.

1. **Manufacturer dominates, 3PL provider adjusts - scenario 1.**

   (a) **Manufacturer's problem.** The manufacturer determines an optimal production schedule in two steps. In the first step, because of the dominance, the manufacturer can plan his schedule disregarding the role of the 3PL provider, i.e., assuming that each order will be transported to M_2 immediately after release from M_1. Hence, in the first step, the manufacturer faces a 2-machine permutation flow shop scheduling problem with a constant transportation time-lag. Following the three-field notation $\alpha|\beta|\gamma$ for machine scheduling problems (Graham et al. 1979), this problem is denoted by $F2|time-lag|C_{\max}$. In the second step, the manufacturer first imposes the constraints, i.e. the release time C_j^1 and the deadline $\overline{d}_j$ of order j, $j = 1, \ldots, n$, to the 3PL provider. Then, the manufacturer adjusts the starting time of the orders on M_2 on minimizing $C_{\max}$ subject to the delivery times D_j of each order $j = 1, \ldots, n$ offered by the 3PL provider and the sequence of production on M_1.

   (b) **3PL provider's problem.** Given the release time C_j^1 and the deadline $\overline{d}_j$ of each order $j, j = 1, \ldots, n$, the 3PL provider aims at finding a delivery schedule that minimizes the transportation cost TC and such that each order is delivered to M_2 within $\overline{d}_j$, i.e., it must hold $D_j \leq C_j^1 + T$. The 3PL provider's problem can be seen as an IPODS (Integrated production and outbound distribution scheduling) problem with a single machine, deadlines and direct batch delivery to a single customer, in which the sequence on the machine is fixed. Following the five-field notation $\alpha|\beta|\pi|\delta|\gamma$ of Chen (2010), the problem can therefore be denoted as $1|fseq, \overline{d}_j = C_j^1 + T|\pi|1|TC$, where:

      α: 1 refers to a single machine (M_1) and *fseq* indicates that the sequence on M_1 is fixed,

      β: deadlines are defined as $C_j^1 + T$,

      π: this field may contain the following values:

      - *direct* indicate the direct batch delivery
      - $V^1(x_1, y_1)$, where $x_1 \in \{V_1, \infty\}$ is the total number of vehicles for reg-

ular transportation (either limited or unlimited) and $y_1 \in \{c_1, \infty\}$ is the capacity of each vehicle for regular transportation (either limited or unlimited).

- $V^2(x_2, y_2)$, where $x_2 \in \{V_2, \infty\}$ represents the total number of vehicles for express transportation (either limited or unlimited) and $y_2 \in \{c_2, \infty\}$ represents the capacity of each vehicle for express transportation(either limited or unlimited).

δ: this field is equal to 1, which refers to a single customer (M_2)

γ: this field contains the objective function, which is the total cost to the 3PL provider (TC).

2. **3PL provider dominates, Manufacturer adjusts - scenario 2.**

   (a) **3PL provider's problem.** In this scenario, on the one hand the 3PL provider wants to relax the responsiveness by increasing the value of T, on the other hand he want to find the best constraints of release time by imposing the delivery schedule to the manufacturer. With the necessary information of production, i.e. the processing time of orders on M_1, the 3PL provider aims at determining a production-distribution schedule (hence, both a production schedule on M_1 and a delivery schedule) such that the transportation cost TC is minimized, while delivering each order within T from its release by M_1 (i.e., $\overline{d}_j = C_j^1 + T$ for each order j). We consider production schedules in which M_1 continuously processes orders, with no idle time. This constraint enforces a minimum productivity requirement. This problem can be seen as an IPODS problem, denoted by $1|no-idle, \overline{d}_j = C_j^1 + T|\pi|1|TC$. In the β field we specify that M_1 must never be idle, while π is as described in the previous scenario. Then according to the obtained schedule, the 3PL provider offers a limited set of regular vehicles and express vehicles with given capacities and transportation times to the manufacturer.

   (b) **Manufacturer's problem.** The manufacturer determines a production schedule on both machines minimizing the $C_{\max}$ subject to a limited set of regular vehicles and express vehicles with given capacities and transportation

times. The responsiveness constraints are not considered.

Example We consider the following instance. The manufacturer has to process a set of $n = 5$ orders. Table 3.1 shows their processing time on the two machines. The 3PL provider has two types of vehicles, $V_1 = 2$ and $V_2 = 5$. The capacities of vehicles are $c_1 = 3$ and $c_2 = 1$. The transportation times are $\ell_1 = 5$ and $\ell_2 = \ell'_2 = 5$. The transportation costs are $h_1 = 10$ and $h_2 = 7$. For regular transportation, there are $L = 2$ fixed departure dates, $t_1 = 15$ and $t_2 = 30$. For each fixed departure date, there is one regular vehicle, i.e. $v_s = 1$ for $s = 1, 2$. The responsiveness parameter is $T = 12$.

Table 3.1: Example for problem 1

Order j	1	2	3	4	5
p_j^1	2	8	10	4	6
p_j^2	3	4	5	2	3

1. **Manufacturer dominates, 3PL provider adjusts - scenario 1.** Figure 3.1 illustrates the optimal production schedule $\sigma_1 = \{1, 3, 2, 5, 4\}$ and the adjusted delivery schedule θ_1. In θ_1, there are four batches {1},{3},{2} and $\{5, 4\}$, where $\{5, 4\}$ is a regular batch departing at time 30, while {1},{3} and {2} are three express batches, departing at times 2, 12 and 20 respectively. We have therefore $C_{\max}(\sigma_1, \theta_1) = 40$ and $TC(\sigma_1, \theta_1) = h_1 + 3h_2 = 31$.

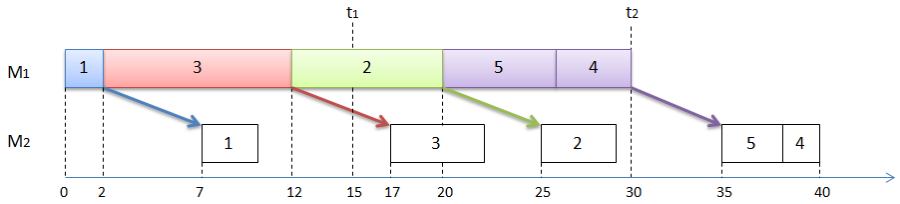


Figure 3.1: Production-distribution schedule when manufacturer dominates, 3PL provider adjusts

2. **3PL provider dominates, Manufacturer adjusts - scenario 2.** Figure 3.2 illustrates the optimal delivery schedule θ_2 with only two regular batches $\{2, 1, 4\}$ and $\{3, 5\}$, departing at times 15 and 30 respectively. The adjusted production

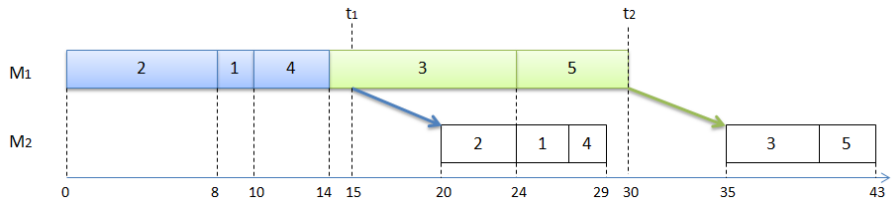


Figure 3.2: Production-distribution schedule when 3PL provider dominates, Manufacturer adjusts

schedule $\sigma_2 = \{2, 1, 4, 3, 5\}$. We have therefore $C_{\max}(\sigma_2, \theta_2) = 43$ and $TC(\sigma_2, \theta_2) = 2h_1 = 20$.

3.3 General properties of optimal delivery schedules

In this section we give some general properties of the optimal delivery schedules that apply to scenarios (1) and (2), hence to all problems denoted by $1|\cdot, \overline{d}_j = C_j^1 + T|\pi|1|TC$, where $\cdot$ may actually be either $fseq$ (in the 3PL provider's problem when the manufacturer dominates) or $no-idle$ (in the 3PL provider's problem when the 3PL provider dominates).

In what follows, we say that a batch B *is split by another batch* B' if B contains some orders that are processed *before* some orders of B', and some orders which are processed *after* some orders of B' on M_1. Also, given a delivery schedule, we say that we *swap* two orders i and j to mean that we move order i to $B(j)$ and order j to $B(i)$, without changing the production schedule σ on M_1 nor the departure time of each batch.

Lemma 3.1 There exists an optimal delivery schedule for the problem $1|\cdot, \overline{d}_j = C_j^1 + T|\pi|1|TC$, such that the following properties hold :

1. If orders i and j are delivered by the same transportation mode and $C_i^1 < C_j^1$, then $D_i \leq D_j$.

2. An express batch can be only split by regular batches and a regular batch can be only split by express batches.

3. If there exists one transportation mode only, the orders of each batch are processed consecutively on machine M_1.

4. If there exist two transportation modes,

   (a) If $\ell_1 \geq \ell_2$, no regular batch is split.

   (b) If $\ell_1 \leq \ell_2$, no express batch is split.

5. If there exists an unlimited number of vehicles for express transportation, each express batch departs at the completion time of the last order in this batch.

6. If there exists a limited number of vehicles for express transportation, each express batch departs either at the completion time of the last order in this batch or as soon as a vehicle for express transportation is available.

Proof. In what follows, we denote by $Se(i, j)$ the subsequence of orders from $\sigma(i)$ through $\sigma(j)$ on M_1. We do not change the production schedule σ.

Property 1: Suppose that there is an optimal delivery schedule such that order i is delivered after order j by the same transportation mode although $C_i^1 < C_j^1$, i.e., $D_{B(j)} < D_{B(i)}$ and $\overline{d}_i < \overline{d}_j$. Since $D_{B(i)} \leq \overline{d}_i$, one has $D_{B(j)} < D_{B(i)} \leq \overline{d}_i < \overline{d}_j$. We can then swap i and j obtaining a new delivery schedule which is still feasible and has the same transportation cost. This argument can be repeated until the delivery schedule satisfies Property 1.

Property 2: Suppose there is an optimal delivery schedule which does not respect Property 2, i.e., assume that a regular batch B is split by some express batches and some regular batches. Consider the subsequence $Se(i, j)$, where $\sigma(i)$ and $\sigma(j)$ are respectively the first and the last order of batch B. Due to Property 1, $D_{B(\sigma(i))} \leq D_{B'} \leq D_{B(\sigma(j))}$, where B' is any regular batch in $Se(i, j)$. Since $D_{B(\sigma(i))} = D_{B(\sigma(j))}$, all regular batches in $Se(i, j)$ have the same departure time. In the subsequence $Se(i, j)$, we do the following: as long as there is an h and an k, $i \leq h < k \leq j$, such that $\sigma(h) \in B$ and $\sigma(k) \in B'$, where B' is a regular batch different from B, swap $\sigma(h)$ and $\sigma(k)$. When no further such swap is possible, we stop. In the resulting delivery schedule, batch B is split only by express batches. By a very similar argument, one can prove that there is an optimal schedule in which any express batch can be only split by regular batches.

Property 3: This is a direct consequence of Property 2. If only one transportation mode exists (i.e., either $V_1 = 0$ or $V_2 = 0$), no batch can be split.

Property 4(a): Suppose that in an optimal delivery schedule θ, the orders of a regular batch B are not processed consecutively on machine M_1 when $\ell_1 \geq \ell_2$. According to Property 2 above, batch B can be only split by express batches. Let again $\sigma(i)$ and $\sigma(j)$ be the first and last order in B, consider the subsequence $Se(i,j)$, and let $\sigma(u)$ be the *first* order in $Se(i,j)$ belonging to an express batch such that $D_{\sigma(u)} - \ell_2 \geq C_{\sigma(j)}$ (see Figure 3.3, in which $\sigma(i) = 4$, $\sigma(j) = 8$ and $\sigma(u) = 6$). Note that order $\sigma(u)$ is the first order of an express batch. Hence, in the subsequence $Se(i, u-1)$, we do the following: as long as there is an h, $i \leq h \leq u-2$, such that $\sigma(h) \in B$ and $\sigma(h+1) \in B'$, where B' is an express batch, swap $\sigma(h)$ and $\sigma(h+1)$. When no further such swap is possible, we stop. In the resulting delivery schedule $\hat{\theta}$, batch B is scheduled after all express batches preceding order u. In $\hat{\theta}$, the deadline of the first order of batch B has been delayed, so the orders of batch B are not late. For each order $\sigma(k)$ from any express batch, recalling that $D_{\sigma(k)} - \ell_2 < C_{\sigma(j)}$, $C_{\sigma(j)} + \ell_1 \leq \overline{d}_{\sigma(i)}$ and $\ell_2 \leq \ell_1$, one has that $D_{\sigma(k)} \leq C_{\sigma(j)} + \ell_2 \leq \overline{d}_{\sigma(i)} \leq \overline{d}_{\sigma(k)}$. So no order from any express batch is late. Symmetrically, starting now from $\hat{\theta}$, consider the subsequence $Se(u,j)$, and do the following: as long as there is an h such that $\sigma(h) \in B'$ and $\sigma(h+1) \in B$, where B' is an express batch, then swap $\sigma(h)$ and $\sigma(h+1)$. When no further such swap is possible, we stop. In the resulting delivery schedule $\overline{\theta}$, batch B terminates before all express batches following order u. Note that, since neither the leftmost order of B nor D_B have changed, no order of batch B is late. Also, the deadline of the first order of no express batch has been anticipated, so again no order from any express batch is late. In conclusion, $\overline{\theta}$ is optimal and the orders of batch B are processed consecutively on machine M_1 (see Figure 3.3).

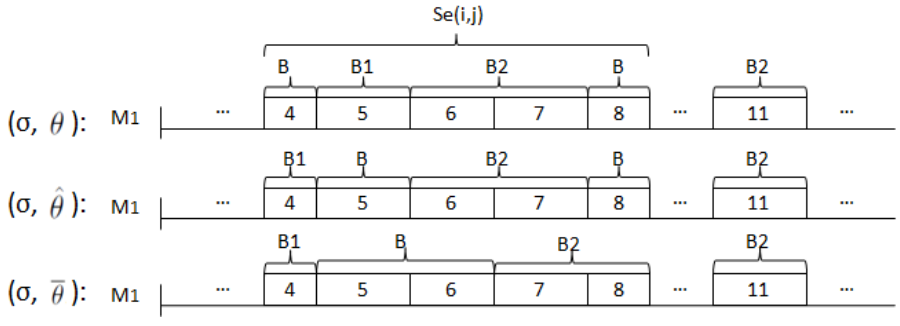


Figure 3.3: Illustration of property 4(a) of Lemma 3.1.

Applying this argument to each regular batch B which is split, we can obtain a new optimal production-distribution schedule in which no regular batch is split.

Property 4(b): The proof is symmetrical to that of Property 4(a).

Property 5: Simply observe that if, in an optimal delivery schedule, the departure time of an express batch B is later than the completion time of the last order in B, one can simply anticipate the departure time of batch B to the completion time of the last order in batch B. If there is an unlimited number of vehicles for express transportation, this does not affect the subsequent schedule.

Property 6: Similar to that of Property 5, but since the number of vehicles is limited, now we anticipate the departure time of a batch B either to the completion time of the last order in B or to the time the vehicle transporting B becomes available. □

3.4 Manufacturer Dominates, 3PL Provider Adjusts - scenario 1

In this section, we address the first scenario, i.e., the manufacturer defines a schedule on M_1, and the 3PL provider has to comply with such schedule. Thereafter, the manufacturer defines a schedule on M_2 accounting for the delivery times of the orders. We address the manufacturer's problem $F2|time-lag|C_{\max}$ and the 3PL provider's problem $1|fseq, \overline{d}_j = C_j^1 + T|\pi|1|TC$ separately. In the latter problem, we consider the cases with different delivery characteristics π, i.e. one transportation mode or both transportation modes, unlimited or limited number of vehicles.

3.4.1 Manufacturer's problem

The manufacturer determines an optimal production schedule in two steps. In the first step, because of the dominance, the manufacturer can plan his schedule disregarding the role of the 3PL provider, i.e., assuming that each order will be transported to M_2 immediately after release from M_1. As observed in the previous section, the problem faced by the manufacturer is a 2-machine permutation flow shop scheduling problem, minimizing $C_{\max}$, with the consideration of a constant transportation time-lag. Fol-

lowing the commonly used three-field notation $\alpha|\beta|\gamma$ for machine scheduling problems (Graham et al. 1979), this problem is denoted by $F2|time-lag=\tau|C_{\max}$. This problem without permutation, is solved in $O(n\log n)$ by the well known Johnson's algorithm (Johnson 1954). Since the Johnson's schedule is also a solution for permutation problem, the manufacturer's problem can be solved by the Johnson's algorithm. In fact, $F2|time-lag|C_{\max}$ and $F2||C_{\max}$ are equivalent problems. Indeed, we can delay an optimal schedule of $F2||C_{\max}$ on M_2 by $time-lag$ time units to obtain an optimal schedule for $F2|time-lag|C_{\max}$, and viceversa, given an optimal schedule for $F2|time-lag|C_{\max}$, by removing the delays of $time-lag$ time units before machine M_2 one gets an optimal schedule for $F2||C_{\max}$.

In the second step, the manufacturer first imposes the constraints, i.e. the release time C_j^1 and the deadline $\overline{d}_j$ of order j, $j=1,\ldots,n$, to the 3PL provider, then requires the information about the delivery time $\overline{d}_j=C_j^1+T$ of each order $j=1,\ldots,n$. In this step, the objective of the manufacturer is to adjust the starting time of orders on M_2 while minimizing $C_{\max}$ subject to the delivery times D_j and the sequence of production on M_1. The sequence of production is not allowed to be changed after it is imposed to the 3PL provider. Hence, we can obtain the optimal solution as follows: from the beginning to the end of sequence, schedule the orders one by one as early as possible, i.e. the starting time of order j is the minimum time between its release time and the end of the previous scheduled order on M_2.

3.4.2 3PL provider's problem

The 3PL provider's problem is to determine an optimal delivery schedule minimizing the transportation cost TC, subject to the release time C_j^1 and the deadline $\overline{d}_j$ of each order $j, j=1,\ldots,n$. As mentioned in the precedent section, this problem can be denoted as $1,fseq|\overline{d}_j=C_j^1+T|\pi|1|TC$. We consider the cases with different delivery characteristics π, i.e. one transportation mode or both transportation modes, unlimited or limited number of vehicles. Throughout Section 3.4.2, since the production schedule is fixed, we can simplify the notation assuming that the orders are numbered by increasing completion times, i.e. $C_1^1<\ldots<C_n^1$.

In the following subsections, we analyze the complexity of the 3PL provider's problem

in different cases. We consider the problems with both transportation modes (subsections (i) and (ii)), the problems with regular transportation only (subsection (iii)), and the problems with express transportation only (subsection (iv)). Table 3.2 summarizes the obtained complexity results.

Table 3.2: Complexity of 3PL provider's problem when manufacturer dominates and 3PL provider adjusts.

Vehicle		**Compl. status**	**Algorithm complexity**	**Subsection**
Regular	**Express**			
$V^1(V_1, c_1)$	$V^2(\infty, c_2)$	P	$O((n(L+V_1))^2)$ when $\ell_1 = \ell_2$, $O(n^3((L+V_1)^2c_2 + (L+V_1)c_2^2))$ when $\ell_1 > \ell_2$, $O(n^3((L+V_1)c_1)^2)$ when $\ell_1 < \ell_2$	(i)
$V^1(\infty, c_1)$	$V^2(\infty, c_2)$	P	$O((nL)^2)$ when $\ell_1 = \ell_2$, $O(n^3(L^2c_2 + Lc_2^2))$ when $\ell_1 > \ell_2$, $O(n^3(Lc_1)^2)$ when $\ell_1 < \ell_2$	(i)
$V^1(V_1, c_1)$	$V^2(V_2, c_2)$	P when V_2 is fixed, Open when V_2 is arbitrary	$O(n^{2V_2+2}(L+V_1)^2 + n^{2V_2+4}(L+V_1))$ when $\ell_1 = \ell_2$, $O(n^{2V_2+3}(L+V_1)^2c_2 + n^{2V_2+5}(L+V_1)c_2^2)$ when $\ell_1 > \ell_2$, $O(n^{2V_2+3}((L+V_1)c_1)^2 + n^{2V_2+5}(L+V_1)c_1)$ when $\ell_1 < \ell_2$	(ii)
$V^1(\infty, c_1)$	$V^2(V_2, c_2)$	P when V_2 is fixed, Open when V_2 is arbitrary	$O(n^{2V_2+2}L^2 + n^{2V_2+4}L)$ when $\ell_1 = \ell_2$, $O(n^{2V_2+3}L^2c_2 + n^{2V_2+5}Lc_2^2)$ when $\ell_1 > \ell_2$, $O(n^{2V_2+3}(Lc_1)^2 + n^{2V_2+5}Lc_1)$ when $\ell_1 < \ell_2$	(ii)
$V^1(\infty, c_1)$	-	P	$O(n+L)$	(iii)
$V^1(V_1, c_1)$	-	P	$O(n + V_1 \min\{V_1, n\})$	(iii)
-	$V^2(\infty, c_2)$	P	$O(n)$	(iv)
-	$V^2(V_2, c_2)$	P	$O(n)$	(iv)

(i) Problems with two transportation modes and an unlimited number of vehicles for express transportation

We first address the problem $1|fseq, \overline{d}_j = C_j^1 + T|V^1(V_1, c_1), V^2(\infty, c_2), \ell_1 > \ell_2, direct|1|TC$, in which there is a limited number V_1 of vehicles for regular trans-

portation and an unlimited number of vehicles for express transportation, and $\ell_1 > \ell_2$. We present a polynomial-time dynamic programming algorithm to solve this problem. Recall that according to Property 4(a) of Lemma 3.1, the orders of each regular batch are processed consecutively on M_1, according to Property 2 of Lemma 3.1 each express batch can be only split by regular batches, and according to Property 5 of Lemma 3.1 each express batch departs at the completion time of the last order in this batch.

Algorithm DP3.1

State and Value Function

The dynamic program considers state $(i, j_1, j_2, s, v, g_2, m)$, in which:

- the first i orders have been delivered
- order j_1 is the first order of the last regular batch scheduled so far, order j_2 is the first order of the last express batch scheduled so far
- parameter m indicates the transportation mode of the last batch, namely $m = 1$ stands for regular transportation, and $m = 2$ for express transportation
- the current last regular batch departs at time t_s
- at time t_s, $v \leq v_s$ vehicles for regular transportation have been used
- the current last *express* batch consists of g_2 orders. If $m = 2$, order i is the current last order of such batch, thus it departs at time C_i^1.

We let $f(i, j_1, j_2, s, v, g_2, m)$ denote the minimum transportation cost of all schedules in state $(i, j_1, j_2, s, v, g_2, m)$.

Boundary Condition

$f(0, 0, 0, 0, 0, 0, 1) = 0,\ f(0, 0, 0, 0, 0, 0, 2) = 0.$

Optimal Solution Value

$\min_{(j_1,j_2,s,v,g_2,m)\in A} f(n, j_1, j_2, s, v, g_2, m)$,

where $A = \{(j_1, j_2, s, v, g_2, m) | 1 \leq j_1 \leq n, 1 \leq j_2 \leq n, 0 \leq s \leq L, 0 \leq v \leq v_s, 0 \leq g_2 \leq c_2, m = 1, 2\}$.

Recurrence Relation

We next consider the recurrence relation for $f(i, j_1, j_2, s, v, g_2, m)$. Actually, such state is only defined for values of the indices such that:

- $i \in \{1, \ldots, n\}$, $j_1 \in \{1, \ldots, i\}$, $j_2 \in \{1, \ldots, i\}$, $s \in \{0, \ldots, L\}$, $v \in \{0, \ldots, v_s\}$, $g_2 \in \{0, \ldots, c_2\}$, $m \in \{1, 2\}$
- if $m = 1$, then $s > 0$, $v > 0$ and:
  - $C_i^1 \leq t_s$, i.e., order i must be completed before the batch departs
  - $t_s + \ell_1 \leq C_{j_1}^1 + T$, i.e., the vehicle must arrive in M_2 within the most restrictive due date of an order in the batch
  - $i - j_1 + 1 \leq c_1$, i.e., the batch size cannot exceed the vehicle capacity
- if $m = 2$, then
  - $C_i^1 + \ell_2 \leq C_{j_2}^1 + T$, i.e., the vehicle must arrive in M_2 within the most restrictive due date of an order in the batch
  - $0 < g_2 \leq c_2$
  - if $i = j_2$, then $g_2 = 1$ since the current last order only consists of order i

If a state does not satisfy all these conditions, we let $f(i, j_1, j_2, s, v, g_2, m) = +\infty$. We can now express the recursive relation as:

$$
f(i, j_1, j_2, s, v, g_2, m) = \min \begin{cases} f(i-1, j_1, j_2, s, v, g_2, m), & \text{if } i > j_1, m = 1 \\ \min_{(j_1', m', s', v') \in A_1} f(i-1, j_1', j_2, s', v', g_2, m') + h_1, & \text{if } i = j_1, m = 1 \\ \min\{f(i-1, j_1, j_2, s, v, g_2 - 1, 1), & \\ \quad f(i-1, j_1, j_2, s, v, g_2 - 1, 2)\}, & \text{if } i > j_2, m = 2 \\ \min_{(j_2', g_2', m') \in A_2} f(i-1, j_1, j_2', s, v, g_2', m') + h_2, & \text{if } i = j_2, m = 2 \end{cases}
$$

where

$A_1 = \{(j_1', m', s', v') | 1 \leq j_1' \leq i-1, m' = 1, 2$, if $v > 1$, then $s' = s$ and $v' = v - 1$, otherwise $0 \leq s' < s$ and $0 \leq v' \leq v_{s'}\}$,

$A_2 = \{(j_2', g_2', m') | 1 \leq j_2' \leq i-1, 0 \leq g_2' \leq c_2, m' = 1, 2\}$.

The four terms in the recurrence relation have the following meaning.

- The first term corresponds to the case in which order i is added to the current last batch, and such batch is regular. According to Property 4(a) of Lemma 3.1, such batch is not split by other batches, so it contains orders $j_1, \ldots, i$.
- In the second term, order i is the first order of a *new* regular batch. In this case, we must consider all possible states of the first $i-1$ orders, in which the last *express* batch is the same as state $(i, j_1, j_2, s, v, g_2, m)$. All these states are described by set A_1. Note that if $v = 1$, the new regular batch is indeed the first being shipped at time t_s, whereas if $v > 1$ other vehicles have been already planned to depart at t_s, so we must only consider states of type $(i-1, j_1', j_2, s, v-1, g_2, m')$
- The third term considers the case in which order i is added to the current last batch, and such batch is express. Note that order i and j_2 are both in such batch, while order $i-1$ can either be in a regular batch (so that we consider $f(i-1, j_1, j_2, s, v, g_2 - 1, 1)$) or in the same express batch as order i (and hence we consider $f(i-1, j_1, j_2, s, v, g_2 - 1, 2)$). In the former case, the regular batch splits the current last batch.
- The fourth term is similar to the second. In this term, order i is the first order of a *new* express batch. This term considers all possible states of the first $i-1$ orders, in which the last regular batch is the same as state $(i, j_1, j_2, s, v, g_2, m)$. These states are described by set A_2.

Theorem 3.1 Algorithm DP3.1 finds an optimal delivery schedule for problem $1|fseq, \overline{d}_j = C_j^1 + T|V^1(V_1, c_1), V^2(\infty, c_2), \ell_1 > \ell_2, direct|1|TC$ in polynomial time $O(n^3((L+V_1)^2 c_2 + (L+V_1)c_2^2))$.

Proof. There exists $O(n^3(L+V_1)c_2)$ states. In fact, for each $s \in \{0, ..., L\}$, v can assume $v_s + 1$ distinct values. Since $\sum_s v_s = V_1$, one has $O(L+V_1)$ possible pairs (s, v). In the recurrence relation, the first and third term require $O(1)$ time and are applied respectively for $O(n^3(L+V_1)c_2)$ states. The second term requires $O(n(L+V_1))$ time

and is applied for $O(n^2(L+V_1)c_2)$ states. The fourth term requires $O(nc_2)$ time and is applied for $O(n^2(L+V_1)c_2)$ states. Therefore, the complexity of algorithm DP3.1 is $O(n^3((L+V_1)^2c_2+(L+V_1)c_2^2))$. Note that since $V_1 \leq Ln$ and $c_2 \leq n$, this problem is solvable in polynomial time. □

Then we provide a similar dynamic programming algorithm for the problem $1|fseq, \overline{d}_j = C_j^1 + T|V^1(V_1, c_1), V^2(\infty, c_2), \ell_1 < \ell_2, direct|1|TC$. Recall that according to Property 4(b) of Lemma 3.1, the orders of each express batch are processed consecutively on M_1, according to Property 2 of Lemma 3.1 each regular batch can be only split by express batches, and according to Property 5 of Lemma 3.1 each express batch departs at the completion time of the last order in this batch.

Algorithm DP3.2

State and Value Function, Boundary Condition, Optimal Solution Value:

From the algorithm DP3.1, we replace g_2 by g_1 which represents the number of orders in the current last *regular* batch and c_2 by c_1.

Recurrence Relation

We next consider the recurrence relation for $f(i, j_1, j_2, s, v, g_1, m)$. Actually, such state is only defined for values of the indices such that:

- $i \in \{1, \ldots, n\}$, $j_1 \in \{1, \ldots, i\}$, $j_2 \in \{1, \ldots, i\}$, $s \in \{0, \ldots, L\}$, $v \in \{0, \ldots, v_s\}$, $g_1 \in \{0, \ldots, c_1\}$, $m \in \{1, 2\}$
- if $m = 1$, then $s > 0$, $v > 0$, $g_1 > 0$ and:
  - $C_i^1 \leq t_s$, i.e., order i must be completed before the batch departs
  - $t_s + \ell_1 \leq C_{j_1}^1 + T$, i.e., the vehicle must arrive in M_2 within the most restrictive due date of an order in the batch
  - if $i = j_1$, then $g_1 = 1$ since the current last order only consists of order i
- if $m = 2$, then $C_i^1 + \ell_2 \leq C_{j_2}^1 + T$, i.e., the vehicle must arrive in M_2 within the most restrictive due date of an order in the batch, and $i - j_2 + 1 \leq c_2$, i.e., the batch size cannot exceed the vehicle capacity.

If a state does not satisfy all these conditions, we let $f(i, j_1, j_2, s, v, g_1, m) = +\infty$. We can now express the recursive relation as:

$$f(i, j_1, j_2, s, v, g_1, m) =$$
$$\min \begin{cases} \min\{f(i-1, j_1, j_2, s, v, g_1 - 1, 1), & \\ \qquad f(i-1, j_1, j_2, s, v, g_1 - 1, 2)\}, & \text{if } i > j_1, m = 1 \\ \min_{(j_1', m', s', v', g_1') \in A_1} f(i-1, j_1', j_2, s', v', g_1', m') + h_1, & \text{if } i = j_1, m = 1 \\ f(i-1, j_1, j_2, s, v, g_1, m), & \text{if } i > j_2, m = 2 \\ \min_{(j_2', m') \in A_2} f(i-1, j_1, j_2', s, v, g_1, m') + h_2, & \text{if } i = j_2, m = 2 \end{cases}$$

where

$A_1 = \{(j_1', m', s', v', g_1') | 1 \le j_1' \le i-1, m' = 1, 2,\ 0 \le g_1' \le c_1$, if $v > 1$, then $s' = s$ and $v' = v - 1$, otherwise $0 \le s' < s$ and $0 \le v' \le v_{s'}\}$,

$A_2 = \{(j_2', m') | 1 \le j_2' \le i-1, m' = 1, 2\}$.

The interpretation of the recurrence relation of algorithm DP3.3 is similar to that of algorithm DP3.1.

Theorem 3.2 Algorithm DP3.2 finds an optimal delivery schedule for problem $1|fseq, \overline{d}_j = C_j^1 + T|V^1(V_1, c_1), V^2(\infty, c_2), \ell_1 < \ell_2, direct|1|TC$ in polynomial time $O(n^3((L + V_1)c_1)^2)$.

Proof. The proof is similar to that of algorithm DP3.1. □

Finally, we provide a similar polynomial-time dynamic programming algorithm for the problem $1|fseq, \overline{d}_j = C_j^1 + T|V^1(V_1, c_1), V^2(\infty, c_2), \ell_1 = \ell_2, direct|1|TC$. Recall that according to Property 4 of Lemma 3.1, the orders of each batch are processed consecutively on M_1. Hence we can simplify the dynamic programming as follows.

Algorithm DP3.3

State and Value Function, Boundary Condition, Optimal Solution Value, Current relation:

From the algorithm DP3.1, we remove g_2, replace two parameters j_1 and j_2 by one

parameter j which represents the first order of the last batch.

Recurrence Relation

$$f(i,j,s,v,m) = \min \begin{cases} f(i-1,j,s,v,m), & \text{if } i > j, m = 1 \\ \min_{(j',m',s',v') \in A_1} f(i-1,j',s',v',m') + h_1, & \text{if } i = j, m = 1 \\ f(i-1,j,s,v,m), & \text{if } i > j, m = 2 \\ \min_{(j',m') \in A_2} f(i-1,j',s,v,m') + h_2, & \text{if } i = j, m = 2 \end{cases}$$

where

$A_1 = \{(j',m',s',v') | 1 \leq j' \leq i-1, m' = 1,2,$ if $v > 1$, then $s' = s$ and $v' = v-1$, otherwise $0 \leq s' < s$ and $0 \leq v' \leq v_{s'}\}$,

$A_2 = \{(j',m') | 1 \leq j' \leq i-1, m' = 1,2\}$.

The interpretation of the recurrence relation of algorithm DP3.3 is similar to that of algorithm DP3.1.

Theorem 3.3 Algorithm DP3.3 finds an optimal delivery schedule for problem $1|fseq, \overline{d}_j = C_j^1 + T|V^1(V_1, c_1), V^2(\infty, c_2), \ell_1 = \ell_2, direct|1|TC$ in polynomial time $O((n(L+V_1))^2)$.

Proof. The proof is similar to that of algorithm DP3.1. □

For the other problems with an unlimited number of vehicles for regular transportation, i.e. $1|fseq, \overline{d}_j = C_j^1 + T|V^1(\infty, c_1), V^2(\infty, c_2), \ell_1 > \ell_2, direct|1|TC$, $1|fseq, \overline{d}_j = C_j^1 + T|V^1(\infty, c_1), V^2(\infty, c_2), \ell_1 < \ell_2, direct|1|TC$, $1|fseq, \overline{d}_j = C_j^1 + T|V^1(\infty, c_1), V^2(\infty, c_2), \ell_1 = \ell_2, direct|1|TC$, we can give similar dynamic programming algorithms by removing the parameter v from the algorithms DP3.1, DP3.2 and DP3.3, because we do not need to consider the number of used regular vehicles. The complexity results are shown in Table 3.2.

(ii) Problems with two transportation modes and a limited number of vehicles for express transportation

Let us consider the problem $1|fseq, \overline{d}_j = C_j^1 + T|V^1(V_1, c_1), V^2(V_2, c_2), \ell_1 \geq \ell_2, direct|1|TC$, where both the number of vehicles for regular transportation and that

for express transportation are limited and $\ell_1 > \ell_2$. According to Property 4(a) of Lemma 3.1, the orders of each regular batch are processed consecutively on machine M_1. From Property 2 of Lemma 3.1, each express batch can be only split by regular batches. Since, according to Property 6 of Lemma 3.1, each express batch departs either at the completion time of the last order in this batch or when a vehicle for express transportation becomes available, the possible departure times for vehicles for express transportation are $C_j, C_j^1 + (\ell_2 + \ell_2'), \ldots, C_j^1 + u(\ell_2 + \ell_2')$, for some $u = 0, \ldots, n - j$ and $j = 1, \ldots, n$. Assuming that there are $L' + 1$ distinct candidate departure times, we denote them as $t_0, t_1, t_2, \ldots, t_{L'}$, where $t_0 = 0$. Note that $L' \leq n(n+1)/2$, hence the possible departure times for vehicles for express transportation are $O(n^2)$ (Lee and Chen 2001). We assume that V_2 is fixed. The problem can be solved by a dynamic program which is similar to DP3.1.

Algorithm DP3.4

State and Value Function

The dynamic program considers state $(i, j_1, j_2, s, v, g_2, m, s_1, \ldots, s_{V_2})$, where parameters i, j_1, j_2, s, v, g_2 and m are defined exactly as algorithm DP3.1 (see subsection (i)), and the last V_2 express batches depart at times $t_{s_1}, \ldots, t_{s_{V_2}}$, where $s_k \in \{0, 1, \ldots, L'\}$ for $k = 1, \ldots, V_2$ and $0 \leq s_1 \leq \ldots \leq s_{V_2}$. We let $f(i, j_1, j_2, s, v, g_2, m, s_1, \ldots, s_{V_2})$ denote the minimum transportation cost of all schedules in state $(i, j_1, j_2, s, v, g_2, m, s_1, \ldots, s_{V_2})$.

Boundary Condition

$f(0, 0, 0, 0, 0, 0, 1, 0, \ldots, 0) = 0,\ f(0, 0, 0, 0, 0, 0, 2, 0, \ldots, 0) = 0.$

Optimal Solution Value

$\min_{(j_1, j_2, s, v, g_2, m, s_1, \ldots, s_{V_2}) \in A} f(n, j_1, j_2, s, v, g_2, m, s_1, \ldots, s_{V_2})$,

where $A = \{(j_1, j_2, s, v, g_2, m, s_1, \ldots, s_{V_2}) | 1 \leq j_1 \leq n, 1 \leq j_2 \leq n, 0 \leq s \leq L, 0 \leq v \leq v_s, 0 \leq g_2 \leq c_2, m = 1, 2, 0 \leq s_k \leq L', k = 1, \ldots, V_2\}$.

Recurrence Relation

Considering the recurrence relation for $f(i, j_1, j_2, s, v, g_2, m, s_1, \ldots, s_{V_2})$, we have that this is defined only for values of the indices such that:

- $i \in \{1, \ldots, n\}$, $j_1 \in \{1, \ldots, i\}$, $j_2 \in \{1, \ldots, i\}$, $s \in \{0, \ldots, L\}$, $v \in \{0, \ldots, v_s\}$, $g_2 \in \{0, \ldots, c_2\}$, $m \in \{1, 2\}$

- $s_k \in \{0, \ldots, L'\}$,$k = 1, \ldots, V_2$, $s_u \leq s_{u+1}$, $u = 1, \ldots, V_2 - 1$
- if $m = 1$, then $s > 0$, $v > 0$ and:
  - $C_i \leq t_s$, i.e., order i must be completed before the (regular) batch departs
  - $t_s + \ell_1 \leq C^1_{j_1} + T$, i.e., the vehicle must arrive in M_2 within the most restrictive due date of an order in the batch
  - $i - j_1 + 1 \leq c_1$, i.e., the batch size cannot exceed the vehicle capacity
- if $m = 2$, then
  - $C_i \leq t_{s_{V_2}}$, i.e., order i must be completed before the (express) batch departs
  - $t_{s_{V_2}} + \ell_2 \leq C^1_{j_2} + T$, i.e., the vehicle (departing at $t_{s_{V_2}}$) must arrive in M_2 within the most restrictive due date of an order in the batch
  - $0 < g_2 \leq c_2$
  - if $i = j_2$, then $g_2 = 1$ since the current last order only consists of order i

If a state does not satisfy all these conditions, we let $f(i, j_1, j_2, s, v, g_2, m, s_1, \ldots, s_{V_2}) = +\infty$. We can now express the recursive relation as:

$$f(i, j_1, j_2, s, v, g_2, m, s_1, \ldots, s_{V_2}) =$$

$$\min \begin{cases} f(i-1, j_1, j_2, s, v, g_2, m, s_1, \ldots, s_{V_2}), & \text{if } i > j_1, m = 1 \\ \min_{(j_1', m', s', v') \in A_1} f(i-1, j_1', j_2, s', v', g_2, m', s_1, \ldots, s_{V_2}) + h_1, & \text{if } i = j_1, m = 1 \\ \min\{f(i-1, j_1, j_2, s, v, g_2 - 1, 1, s_1, \ldots, s_{V_2}), & \\ \quad f(i-1, j_1, j_2, s, v, g_2 - 1, 2, s_1, \ldots, s_{V_2})\}, & \text{if } i > j_2, m = 2 \\ \min_{(j_2', g_2', m', s_1') \in A_2} f(i-1, j_1, j_2', s, v, g_2', m', s_1', s_1, \ldots, s_{V_2-1}) & \\ \quad + h_2, & \text{if } i = j_2, m = 2 \end{cases}$$

where

$A_1 = \{(j_1', m', s', v') | 1 \leq j_1' \leq i-1, m' = 1, 2$, if $v > 1$, then $s' = s$ and $v' = v - 1$, otherwise $0 \leq s' < s$ and $0 \leq v' \leq v_{s'}\}$,

$A_2 = \{(j_2', g_2', m', s') | 1 \leq j_2' \leq i-1, 0 \leq g_2' \leq c_2, m' = 1, 2, 0 \leq s_1' \leq s_1, t_{s_1'} + \min\{1, s_1'\} * (\ell_2 + \ell_2') \leq t_{s_{V_2}}\}$.

The four terms in the recurrence relation have the following meaning.

- As in DP3.1, the first term corresponds to the case in which order i is added to the current last batch, and such batch is regular. Due to Property 4(a) of Lemma 3.1, such batch is not split by other batches, so it contains orders $j_1, \ldots, i$.

- Also the second term is very similar to the corresponding term in DP3.1. Order i is the first order of a *new* regular batch. In this case, we must consider all possible states of the first $i-1$ orders, in which the last *express* batch is the same as state $(i, j_1, j_2, s, v, g_2, m, s_1, \ldots, s_{V_2})$. All these states are described by set A_1. If $v = 1$, the new regular batch is indeed the first being shipped at time t_s, whereas if $v > 1$ other vehicles have been already planned to depart at t_s, so we must only consider states of type $(i-1, j_1', j_2, s, v-1, g_2, m')$.

- The third term considers the case in which order i is added to the current last batch, and such batch is express. Note that order i and j_2 are both in such batch, while order $i-1$ can either be in a regular batch (so that we consider $f(i-1, j_1, j_2, s, v, g_2 - 1, 1, s_1, \ldots, s_{V_2})$) or in the same express batch as order i (and hence we consider $f(i-1, j_1, j_2, s, v, g_2 - 1, 2, s_1, \ldots, s_{V_2})$). In the former case, the regular batch splits the current last batch.

- The fourth term is similar to the second. In this term, order i is the first order of a *new* express batch. This term considers all possible states of the first $i-1$ orders, in which the last regular batch is the same as state $(i, j_1, j_2, s, v, g_2, m, s_1, \ldots, s_{V_2})$. These states are described by set A_2.

Theorem 3.4 Algorithm DP3.4 finds an optimal delivery schedule for problem $1|fseq, \overline{d}_j = C_j^1 + T|V^1(\infty, c_1), V^2(\infty, c_2), \ell_1 > \ell_2, direct|1|TC$ in $O(n^{2V_2+3}(L+V_1)^2 c_2 + n^{2V_2+5}(L+V_1)c_2^2)$ time, which is polynomial if V_2 is fixed.

Proof. The proof is similar to that of algorithm DP3.1. □

For the other problems with two transportation modes and limited number of vehicles for express transportation, similar dynamic programming algorithms can be given by using the same method of transformation as the subsection (i). The results are shown

in Table 3.2. If V_2 is fixed, all of the above problems can be solved in polynomial time. Otherwise, the complexities of problems are open.

(iii) Problems with regular transportation only

We next consider problem $1|fseq, \overline{d}_j = C_j^1 + T|V^1(\infty, c_1), direct|1|TC$, which can be seen as a special case of problem $1|fseq, \overline{d}_j = C_j^1 + T|V^1(\infty, c_1), V^2(V_2, c_2), \ell_1 = \ell_2, direct|1|TC$ (see subsection (ii)), with $V_2 = 0$. We can give a similar dynamic programming algorithm that runs in $O((nL)^2)$ time. Actually, we can give a polynomial-time greedy algorithm having smaller complexity. Hereafter we assume, together with the assumption at the beginning of section 3.4.2, that orders and departure times are numbered in increasing order of completion times on M_1 and departure times respectively. In fact, for each order i, we let $t(i)$ denote the first departure time after the completion of order i on M_1, i.e.:

$$t(i) = \min\{t_s : t_s \geq C_i^1, s = 1, \ldots, L\}$$

Note that all values $t(i)$ can be computed in $O(n + L)$, simply scanning the two ordered sets.

The algorithm GA3.1 greedily scans the orders in increasing order of completion times, assigning orders to the current batch as long as no deadline is violated. When the addition of the next order would make the first order late, a new batch is started.

The algorithm exploits the fact that there exists an optimal delivery schedule in which no batch is split (Property 3 of Lemma 3.1). In Algorithm GA3.1, i is the index of the last considered order, j is the index of the last considered batch, P contains the delivery schedule and t_{B_j} is the departure time of batch B_j.

Theorem 3.5 Algorithm GA3.1 finds an optimal delivery schedule for problem $1|fseq, \overline{d}_j = C_j^1 + T|V^1(\infty, c_1)\ , direct|1|TC$ in polynomial time $O(n + L)$.

Proof. Let ϕ be the schedule generated by Algorithm GA3.1, and consider an optimal delivery schedule θ. From Property 3 of Lemma 3.1, no batch can be split in θ. Suppose that θ and ϕ coincide up to the first $k - 1$ batches. According to Algorithm GA3.1, one has $D_{B_k}(\theta) \geq D_{B_k}(\phi)$ and $|B_k(\phi)| \geq |B_k(\theta)|$, where $B_k(\phi)$ and $B_k(\theta)$ denote the k-th batch in ϕ and θ respectively, $|B_k(\phi)|$ denotes the number of orders in batch $B_k(\phi)$. We

Algorithm 1: Algorithm GA3.1

```
1  i = 0 ; j = 0 ; P = ∅; compute all t(i) ; t_s = t(1) ;
2  while i < n do
3      if C^1_{i+1} + T < t_s + ℓ_1 then
4          There is no feasible solution, STOP.
5      else
6          j = j + 1 ; B_j = {i + 1} ; //put order i + 1 in a new batch
7          t_{B_j} = t_s ; //initialize departure time of this new batch
8          i = i + 1 ; q = i ; //save index of the first order of this new batch
9          t_s = t(i + 1) ;
10         while (i < n)&&(|B_j| + 1 ≤ c_1)&&(C^1_q + T ≥ t_s + ℓ_1) do
11             B_j = B_j ∪ {i + 1} ;
12             t_{B_j} = t_s; //update departure time of the current batch
13             i = i + 1; t_s = t(i + 1) ;
14 for w = 1 to j do
15     P = P ∪ B_w;//form delivery schedule by combining all the batches
```

distinguish three cases.

- $|B_k(\phi)| > |B_k(\theta)|$ and $D_{B_k}(\theta) = D_{B_k}(\phi)$. In this case, we add to $B_k(\theta)$ the orders in $B_k(\phi) \setminus B_k(\theta)$, so that $|B_k(\phi)| = |B_k(\theta)|$ without changing $D_{B_k}(\theta)$, which does not increase the transportation cost.

- $|B_k(\phi)| = |B_k(\theta)|$ and $D_{B_k}(\theta) > D_{B_k}(\phi)$. In this case, we anticipate $D_{B_k}(\theta)$, so that $D_{B_k}(\theta) = D_{B_k}(\phi)$, and this does not increase the transportation cost.

- $|B_k(\phi)| > |B_k(\theta)|$ and $D_{B_k}(\theta) > D_{B_k}(\phi)$. In this case one can perform both previous transformations, to get both $|B_k(\phi)| = |B_k(\theta)|$ and $D_{B_k}(\theta) = D_{B_k}(\phi)$, which does not increase the transportation cost.

Applying the same rule, we can transform the subsequent batches in θ as in ϕ without increasing the transportation cost. So schedule ϕ produced by Algorithm GA3.1 is optimal.

For each order, Algorithm GA3.1 checks (at line 13) whether its addition to the current batch violates any deadline or the batch size. This is done in constant time. Since there are n orders, lines 2–13 are executed in $O(n)$, and the complexity of Algorithm GA3.1 is therefore dominated by the computation of values $t(i)$, i.e., $O(n + L)$. □

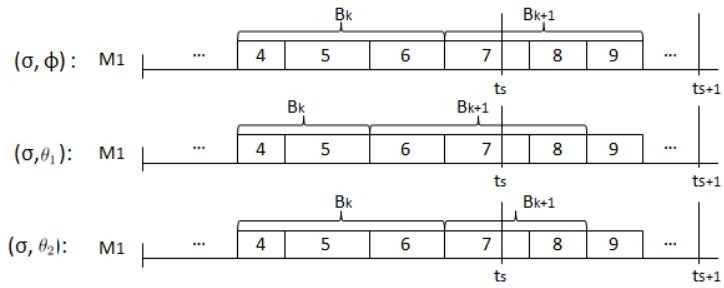


Figure 3.4: Illustration of Theorem 3.5.

Figure 3.4 shows an example of the third case in the proof of Theorem 3.5, where ϕ, θ_1 and θ_2 represent respectively the schedule created by Algorithm GA3.1, the optimal schedule before change and that after change. In this example, $D_{B_k}(\phi) = t_s + l_1$ is earlier than $D_{B_k}(\theta_1) = t_{s+1} + l_1$. $B_k(\phi) = \{4, 5, 6\}$ has 3 orders while $B_k(\theta_1) = \{4, 5\}$ has 2 orders. This accords with the condition of the third case. After change as the proof, we obtain a new optimal schedule θ_2, such that $D_{B_k}(\theta_2) = t_s + l_1$ and $B_k(\theta_2) = \{4, 5, 6\}$.

Similarly, viewing problem $1|fseq, \overline{d}_j = C_j^1 + T|V^1(V_1, c_1), direct|1|TC$ as a special case of $1|fseq, \overline{d}_j = C_j^1 + T|V^1(V_1, c_1), V^2(V_2, c_2), \ell_1 = \ell_2, direct|1|TC$ with $V_2 = 0$, one can solve it by dynamic programming algorithm in time $O((n(L + V_1))^2)$. Actually, we can give a polynomial-time dynamic algorithm having smaller complexity $O(n + V_1 \min\{V_1, n\})$.

In what follows, given two integers A and E, $E \geq A$, we let $[A, E]$ denote the closed interval of integers $A, A + 1, \ldots, E$. Recall that $C_1^1 \leq \cdots \leq C_n^1$ and $t_1 < \cdots < t_L$.

Given an instance of the 3PL provider's problem, consider the i-th departure time t_i. Denote by $[A_i, E_i] \subseteq N$ the maximal interval of orders such that any order from this interval can be feasibly delivered if its delivery starts at time t_i, $i = 1, \ldots, L$. Observe that order j can be feasibly delivered by a vehicle departing at time t_i if and only if $t_i - T + \ell_1 \leq C_j^1 \leq t_i$, $j = 1, \ldots, n$. All the intervals $[A_i, E_i]$, $i = 1, \ldots, L$, can be constructed in $O(n + L)$ time.

The following two lemmas are trivial.

Lemma 3.2 $A_1 \leq \cdots \leq A_L$ and $E_1 \leq \cdots \leq E_L$.

Lemma 3.3 Problem $1|fseq, \overline{d}_j = C_j^1 + T|V^1(V_1, c_1), direct|1|TC$ has a solution only if $\cup_{i=1}^{L}[A_i, E_i] = \{1, \ldots, n\}$.

Note that if an instance of the 3PL provider's problem has a solution, then $A_1 = 1$, $E_L = n$ and $A_{i+1} \leq E_i$, $i = 1, \ldots, n-1$. Moreover, a simple order interchange argument can be used to prove the following:

Lemma 3.4 If the problem $1|fseq, \overline{d}_j = C_j^1 + T|V^1(V_1, c_1), direct|1|TC$ has a solution, then there exists an optimal solution, in which each batch consists of consecutively indexed orders and a batch with smaller order indices is delivered earlier.

We will consider solutions which satisfy Lemma 3.4. In what follows, we conveniently represent an instance and a feasible solution by means of the diagram shown in Figure3.5. There are L rows and n columns. Row i corresponds to departure time t_i, and circles in these row correspond to the orders that can be feasibly delivered by a vehicle departing at t_i. Note that Lemma 3.4 implies that, if a feasible solution exists, no column can be empty. We represent a feasible solution by framing the orders in the same batch. Products assigned to a batch are marked in bold. For the example in Figure 3.5, there are $L = 4$ delivery times, batch sizes are bounded by $c_1 = 4$, and the numbers of vehicles are $v_1 = 4$, $v_2 = 3$, $v_3 = 3$ and $v_4 = 2$.

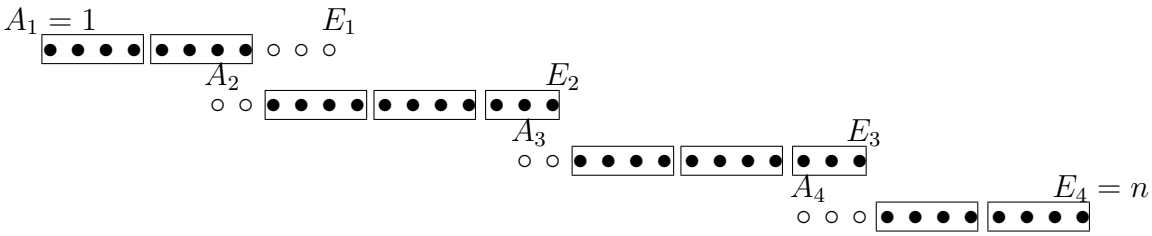


Figure 3.5: Graphical representation of a delivery schedule

We call order $j \in [A_i, E_i]$ an *order in row i*. Note that an order is in at least one row (because of Lemma 3.3) and, in general, is in several rows. Given a feasible solution, we call a batch *full* if it contains exactly c_1 orders. In a feasible solution, we call *residual*

orders the last orders in row i which are not assigned to a vehicle departing at t_i (and, hence, will be delivered later).

Lemma 3.5 If the problem $1|fseq, \overline{d}_j = C_j^1 + T|V^1(V_1, c_1), direct|1|TC$ has a solution, then there exists an optimal solution which satisfies Lemma 3.4 and in which batches in the same row i, $i = 1, \ldots, L$, satisfy one of the following four properties:

(1) all batches are full and their number is v_i,

(2) all batches are full, their number is less than v_i, and the number of residual orders in row i is at most $c_1 - 1$,

(3) row i contains no batch and the number of residual orders in rows $1, \ldots, i$ is at most $c_1 - 1$,

(4) all but one batches are full, the non-full batch is last in row i and it includes order E_i.

Proof. Given an optimal solution which satisfies Lemma 3.4, consider row 1 of the diagram. We will modify it according to the following two cases (a) and (b).

Case (a): $v_1c_1 \leq E_1$. In this case, re-assign orders $1, \ldots, v_1c_1$ to v_1 full batches in row 1. It can be easily seen that the solution remains optimal and satisfying Lemma 3.4. In the obtained solution, the batches in row 1 satisfy property (i).

Case (b): $v_1c_1 > E_1$. In this case, calculate number x, $0 \leq x \leq v_1 - 1$, such that $xc_1 \leq E_1$ and $(x+1)c_1 > E_1$. Re-assign orders $1, \ldots, xc_1$ to x full batches in row 1. If, after performing such reassignment, some orders in $xc_1 + 1, xc_1 + 2, \ldots, E_1$ are still assigned to some batch in row 1, then reassign all orders $xc_1 + 1, xc_1 + 2, \ldots, E_1$ to a last non-full batch in row 1. Otherwise, leave orders $xc_1 + 1, xc_1 + 2, \ldots, E_1$ unassigned (residual). It can be easily seen that the solution remains optimal and satisfying Lemma 3.4. For this solution, if $x \geq 1$, then the batches in row 1 satisfy property (ii) or (iv), and if $x = 0$, then they satisfy property (iii) or (iv).

After the above modification has been done, remove all assigned orders of row 1 from the obtained solution. Also, renumber remaining orders to start from number 1, renumber departure times to start from number 1, and consider the reduced solution in the new notation. Now, one has only $L - 1$ departure times and a smaller number of orders.

All the above arguments can be applied for the reduced solution. Repetition of this argumentation at most L times completes the proof. □

Note that the rows 1, 2, 3 and 4 in Figure 3.5 satisfy properties (ii), (iv), (iv) and (i), respectively.

The above properties justify the following dynamic programming algorithm, denoted as DP3.5. In this algorithm, partial batch sequences satisfying Lemmas 3.4 and 3.5 are constructed. A batch sequence is extended by appending a new batch to its end. A state (i, a, k) is associated with each partial batch sequence, where i is the current row, a is number of batches in row i and k is the total number of batches considered so far. We denote by $G(i, a, k)$ the maximum number of orders that can be feasibly delivered according to the partial batch sequences in the state (i, a, k). It is easy to see that, if there exists a batch sequence in a state (i, a, k) that can be extended to an optimal batch sequence, then the batch sequence corresponding to the value $G(i, a, k)$ can be extended to an optimal batch sequence as well. A batch sequence with the smallest number of batches k, which corresponds to the value $G(L, a, k) = n$, is optimal. A formal description of algorithm DP3.5 is given below.

As a preprocessing step, compute

$$v'_i := \min\{v_i, \lceil (E_i - A_i + 1)/c_1 \rceil\},\ i = 1, \ldots, L, \text{ and } V' = \sum_{i=1}^{L} v'_i.$$

In fact, in view of Lemma 3.5, no more than v'_i vehicles will be used in an optimal solution departing at time t_i.

The initialization is

$$G(1, 0, 0) = 0,\ G(1, a, a) = \min\{ac_1, E_1\},\ a = 1, \ldots, v'_1,$$

and $G(i, a, k) = -\infty$, $i = 2, 3, \ldots, L$, $a = 0, 1, \ldots, v'_i$, $k = 0, 1, \ldots, \min\{V', n\}$.

The recursion for $i = 2, 3, \ldots, L$ is the following.

For $k = 0, 1, \ldots, \min\{V', n\}$,

$$G(i, 0, k) = \max_{0 \le b \le v_{i-1}} \{G(i - 1, b, k)\}.$$

Note that this represents the maximum number of orders which can be feasibly delivered in k batches, when no batch departs at t_i.

For $k = 1, \ldots, \min\{V', n\}$,

$$G(i, 1, k) = \min\{E_i, \max_{0 \le b \le v_{i-1}} \{G(i-1, b, k-1\} + c_1\}\}.$$

Note that this represents the maximum number of orders which can be feasibly delivered in k batches, when exactly one batch departs at t_i. Let the above maximum be attained at $b = b^*$. If $G(i-1, b^*, k-1) + c_1 \le E_i$, then the first batch in row i is full and it includes orders $G(i-1, b^*, k-1) + 1, G(i-1, b^*, k-1) + 2, \ldots, G(i-1, b^*, k-1) + c_1$ (case (i) or (ii) of Lemma 3.5). Otherwise, this batch is non-full and it includes orders $G(i-1, b^*, k-1) + 1, G(i-1, b^*, k-1) + 2, \ldots, E_i$ (case (iv) of Lemma 3.5).

Similarly, we have the general formula for $k = 1, \ldots, \min\{V', n\}$ and $a = 2, 3, \ldots, \min\{k, v_i\}$:

$$G(i, a, k) = \min\{E_i, G(i, a-1, k-1) + c_1\}.$$

This is the maximum number of orders which can be feasibly delivered in k batches, when exactly a batches depart at t_i ($a \ge 2$). If $G(i, a-1, k-1) + c_1 \le E_i$, then the a-th batch is full and it includes orders $G(i, a-1, k-1)+1, G(i, a-1, k-1)+2, \ldots, G(i, a-1, k-1) + c_1$ (case (i) or (ii)). Otherwise, this batch is non-full and it includes orders $G(i, a-1, k-1) + 1, G(i, a-1, k-1) + 2, \ldots, E_i$ (case (iv)).

If, for some (i, a, k), one has $G(i, a, k) \le A_i - 1$, then it is not possible to feasibly accommodate all orders of rows $1, \ldots, i-1$ in k batches, a of which consist of orders in row i. In this case, no feasible solution can be obtained from such partial solution, and hence we reset $G(i, a, k) = -\infty$, $i = 2, 3, \ldots, L$, $k = 0, 1, \ldots, \min\{V', n\}$, $a = 0, 1, \ldots, \min\{k, v_i\}$.

If

$$\max\Big\{G(L, a, k) \mid a = 0, 1, \ldots, v_L,\ k = 1, \ldots, \min\{V', n\}\Big\} \le n - 1,$$

then the 3PL provider's problem has no solution. Otherwise, the minimum number of batches is equal to

$$k^* = \min\Big\{k \mid G(L, a, k) = n,\ a = 0, 1, \ldots, v_L,\ k = 1, \ldots, \min\{V', n\}\Big\}, \qquad (3.4.1)$$

and the corresponding solution can be found by backtracking.

We are now in the position of proving the final result.

Theorem 3.6 Problem $1|fseq, \overline{d}_j = C_j^1 + T|V^1(V_1, c_1), direct|1|TC$ can be solved in time $O(n + V_1 \min\{V_1, n\})$.

Proof. The running time of algorithm DP3.5 is determined by the recursive computations of the values $G(i, 0, k)$, $G(i, 1, k)$ and $G(i, a, k)$ for $i = 2, 3 \ldots, L$, $a = 2, 3, \ldots, \min\{k, v_i'\}$, and $k = 1, \ldots, \min\{V', n\}$. For fixed arguments, values $G(i, 0, k)$ can be computed in $O(v_{i-1}')$ time, and hence all values $G(i, 0, \cdot)$ can be computed in $O(v_{i-1}' \min\{V', n\})$. As i ranges from 1 to L, one has $O((v_1' + v_2' + \cdots + v_L') \min\{V', n\})$, i.e., $O(V' \min\{V', n\})$. The same discussion holds for all values $G(i, 1, k)$, while all values $G(i, a, k)$ can be computed in $O(1)$ time. In conclusion, all G-values can be computed in $O(V' \min\{V', n\})$ time. Finally, the optimal solution can be computed in $O(n)$ by the (3.4.1). Since $V' \leq V_1$, the 3PL provider's problem can be solved in time $O(n + V_1 \min\{V_1, n\})$. □

(iv) Problems with express transportation only

Let us now turn to problems in which $V_1 = 0$. Again, from Property 3 of Lemma 3.1, there exists an optimal delivery schedule in which no batch is split.

If there is an unlimited number of vehicles, considering that $1|fseq, \overline{d}_j = C_j^1 + T|V^2(\infty, c_2), direct|1|TC$ is as a special case of $1|fseq, \overline{d}_j = C_j^1 + T|V^1(V_1, c_1), V^2(\infty, c_2), \ell_1 = \ell_2, direct|1|TC$ with $V_1 = 0$, one can give a dynamic programming algorithm having complexity $O(n^2)$. Indeed, also in this case the problem can be solved more efficiently by a greedy algorithm. Such algorithm is identical to Algorithm GA3.1, except that the departure time of a batch now coincides with the completion time of the last order in the batch, and, of course, there is no need to define $t(i)$. Hence, it suffices replacing the step at lines 9 and 13, "$t_s = t(i + 1)$", with "$t_s = C_{i+1}^1$", and the time complexity is therefore $O(n)$.

Let us now turn to the problem with a limited number of vehicles, i.e., $1|fseq, \overline{d}_j = C_j^1 + T|V^2(V_2, c_2), direct|1|TC$. The complexity of this problem is open if V_2 is not fixed. If V_2 is fixed, the problem can be seen as a special case of problem $1|fseq, \overline{d}_j = C_j^1 + T|V^1(V_1, c_1), V^2(V_2, c_2), \ell_1 = \ell_2, direct|1|TC$ with $V_1 = 0$, and hence one can give a similar dynamic programming algorithm that runs in $O(n^{2V_2+4})$ time. Indeed, in this case the problem can be solved more efficiently. We assume, together with the assumption at

the beginning of section 3.4.2, that orders and departure times are numbered in increasing order of completion times on M_1 and departure times respectively.

Consider the following *batch scheduling problem*. There are $\tilde{n}$ jobs and m parallel identical machines. All jobs have the same processing time $p_j = p$, $j = 1, \ldots, \tilde{n}$. Each job has a release date $r_j \geq 0$ and a deadline d_j, $d_j \geq r_j + p$, $j = 1, \ldots, \tilde{n}$. Jobs are performed in batches having bounded capacity q, i.e., a machine can handle up to q jobs in parallel. Jobs in a batch have the same starting and completion time (this is also known as *batch availability model with parallel job processing and bounded batch capacity*, see Potts and Kovalyov (2000) for the terminology.) Since all job processing times are identical and equal to p, the processing time of any batch equals p. The problem is to find a feasible batch schedule (i.e., a partition of $\tilde{N} = \{1, \ldots, \tilde{n}\}$ into batches and an assignment of batches to machines that respect all release dates and deadlines) with the minimum number of batches.

In what follows, we show that the 3PL provider's problem $1|fseq, \overline{d}_j = C_j^1 + T|V^2(V_2, c_2), direct|1|TC$ can be reduced to batch scheduling problem. In such reduction, express vehicles in 3PL provider's problem correspond to machines in batch scheduling problem, orders in the 3PL provider's problem to jobs in batch scheduling problem and the vehicle round trip, including batch delivery and return in the 3PL provider's problem, to the processing of the corresponding batch in batch scheduling problem. More formally, given an instance I of 3PL provider's problem, construct an instance I_B of batch scheduling problem as follows:

$$\begin{aligned} \tilde{n} &:= n; \\ m &:= V_2; \\ q &:= c_2; \\ r_j &:= C_j^1, \;\; d_j := r_j + T + \ell_2', \;\; p_j = p = \ell_2 + \ell_2', \;\; j = 1, \ldots, \tilde{n}. \end{aligned}$$

Lemma 3.6 Given a feasible solution for the instance I_B of batch scheduling problem, a feasible solution for the instance I of 3PL provider's problem $1|fseq, \overline{d}_j = C_j^1 + T|V^2(V_2, c_2), direct|1|TC$ can be built having the same value of objective function, and vice versa.

Proof. Consider an arbitrary batch G in a feasible solution of I_B. Let $\tilde{C}_G$ denote the

completion time of this batch. Due to the feasibility, the relation $\max_{j\in G}\{r_j\} + p \leq \tilde{C}_G \leq \min_{j\in G}\{d_j\}$ is satisfied. This relation can be written as

$$\max_{j\in G}\{C_j^1\} + \ell_2 + \ell_2' \leq \tilde{C}_G \leq \min_{j\in G}\{C_j^1\} + T + \ell_2',$$

which implies $\max_{j\in G}\{C_j^1\} \leq \tilde{C}_G - \ell_2' - \ell_2$ and $\tilde{C}_G - \ell_2' \leq \min_{j\in G}\{C_j^1\} + T$. Now, associate with batch G a batch H in the corresponding instance I of 3PL provider's problem, and let $\tilde{C}_G - \ell_2' - \ell_2$ and $\tilde{C}_G - \ell_2'$ be its start and delivery times, respectively. The latter two relations prove that the orders in the batch H are feasibly delivered for 3PL provider's problem. Also note that, according to reduction, no two batches in I will be assigned to the same vehicle at the same time and each vehicle returns empty to the upstream stage.

Conversely, let H be a batch in a feasible solution of I. The corresponding vehicle departs not earlier than $\max_{j\in H}\{C_j^1\}$ and delivers not earlier than $\max_{j\in H}\{C_j^1\} + \ell_2$. Let $\tau_{DEL,H}$ be the delivery time of batch H in the feasible solution to I. Due to feasibility, one must have

$$\tau_{DEL,H} - \min_{j\in H}\{C_j^1\} \leq T. \tag{3.4.2}$$

Now, define the completion time of the corresponding batch G in I_B as $\tilde{C}_G := \tau_{DEL,H} + \ell_2'$. Then, from (3.4.2), we obtain

$$\tilde{C}_G \leq \min_{j\in H}\{C_j^1\} + T + \ell_2' = \min_{j\in G}\{d_j\}.$$

Hence, all job deadlines in I_B are satisfied. Furthermore, batch G in I_B starts at time $\tilde{C}_G - p$, and

$$\tau_{DEL,H} + \ell_2' - p = \tau_{DEL,H} - \ell_2 \geq \max_{j\in H}\{C_j^1\} = \max_{j\in G}\{r_j\},$$

so batch G respects job release dates as well. Finally, each vehicle carries one batch and hence no machine will process more than one batch at a time. □

We are now in the position of proving the following result.

Theorem 3.7 Problem $1|fseq, \overline{d}_j = C_j^1 + T|V^2(V_2, c_2), direct|1|TC$ can be solved in $O(n)$ time.

Proof. Given an instance of 3PL provider's problem, it can be solved by constructing the corresponding instance of batch scheduling problem according to the reduction and solving it. Koehler and Khuller (2013) developed an $O(\tilde{n})$ time algorithm for batch scheduling problem in the special case in which release dates and deadlines are *agreeable*, i.e., $r_i < r_j$ implies $d_i \leq d_j$ for any jobs i and j. We observe that, due to the reduction, the instance of batch scheduling problem obtained is indeed agreeable, and as a consequence, the instance of 3PL provider's problem can be solved in $O(n)$ time. □

3.5 3PL Provider Dominates, Manufacturer Adjusts - scenario 2

In this section, we address the second scenario, i.e., the 3PL provider defines a delivery schedule, and the manufacturer has to adjust the production schedule subject to the fixed number of regular vehicles and express vehicles imposed by the 3PL provider. We address the 3PL provider's problem $1|no-idle, \overline{d}_j = C_j^1 + T|\pi|1|TC$ and the manufacturer's problem separately. For the 3PL provider's problem, we consider the cases with one transportation mode and other different delivery characteristics π, i.e. unlimited or limited number of vehicles.

3.5.1 3PL provider's problem

As discussed in Section 3.2, the 3PL provider's problem can be seen as an IPODS (Integrated production and outbound distribution scheduling) problem, where the orders are processed on a single machine and the batches are delivered to a single customer, with deadlines depending on the production schedule. We analyze the complexity of the problems in this scenario. Since in the literature the deadlines are independent of the production schedule, known complexity results do not in general apply to our problems.

In this section, after giving some properties, we address the case in which there are only vehicles for express transportation (subsections (a) and (b)) and the case in which there are only vehicles for regular transportation (subsection (c)). As shown in Table 3.3, the problem with express transportation is polynomially solvable only in the case

with an unlimited number of vehicles of capacity $c_2 \leq 3$. The problem with regular transportation is always strongly NP-hard. Hence the problem with both transportation modes is also strongly NP-hard.

Table 3.3: Complexity of 3PL provider problems when 3PL provider dominates and manufacturer adjusts.

Vehicle		**Compl. status**	**Algorithm complexity**	**Subsection**
Regular	**Express**			
-	$V^2(\infty, 3)$	P	$O(n^2)$	(a)
-	$V^2(\infty, 2)$	P	$O(n \log n)$	(a)
-	$V^2(\infty, 1)$	P	$O(n)$	(a)
-	$V^2(\infty, c_2 \geq 4)$	sNP		(a)
-	$V^2(1, 1)$	sNP		(b)
$V^1(\infty, c_1)$	-	sNP		(c)
$V^1(V_1, c_1)$	-	sNP		(c)

We can prove that the following property holds for the problem $1|no-idle, \overline{d}_j = C_j^1 + T|\pi|1|TC$ with all possible values of π.

Lemma 3.7 There exists an optimal production-distribution schedule for problem $1|no-idle, \overline{d}_j = C_j^1 + T|\pi|1|TC$ such that, if the orders of batch B are processed consecutively on M_1, then the order having longest processing time in the batch is in first position.

Proof. In a given optimal production-distribution schedule (σ, θ), consider a batch B consisting of orders consecutively processed on M_1, and in which the first order, say order j, is not the longest order. Here the batch B can be a regular batch or an express batch. Let k be the longest order in B. Due to feasibility, $D_B \leq \overline{d}_j = C_j^1(\sigma) + T$. Now consider the new production schedule σ' obtained by moving k in first position in its batch, and shifting all other orders forward, without changing the departure time of batch B. Since k is the longest order in B, $C_h^1(\sigma') > C_h^1(\sigma)$ for all $h \neq k$. Hence, since the delivery date D_B of the batch has not changed, all these orders are on time in σ'. The only order which has been moved backward is k. Since $p_k^1 > p_j^1$, then $C_k^1(\sigma') > C_j^1(\sigma)$, and since j was on time in σ, $D_B \leq C_j^1(\sigma) + T < C_k^1(\sigma') + T$. In conclusion, also σ' is feasible and optimal. By repeatedly applying this argument, one can find a production schedule

which respects the property and having the same transportation cost as (σ, θ). □

(a) Problems with express transportation only and an unlimited number of vehicles for express transportation

We address the case in which only express transportation exists and the number of vehicles is unlimited. We show that the complexity of the problem can be fully characterized and the complexity of the problem depends on the capacity of the vehicle.

We first show that if each vehicle can carry at least 4 orders, the problem is difficult. In what follows, we use the following strongly NP-complete problem (Garey and Johnson 1979):

3-PARTITION. Given $3n$ integers $a_1, \ldots, a_{3n}$, so that $\sum a_i = nW$, and such that $W/4 < a_i < W/2$ for all i, is it possible to partition them into n triples each summing up to W?

Theorem 3.8 Problem $1|no-idle, \overline{d}_j = C_j^1 + T|V^2(\infty, c_2), direct|1|TC$ where $c_2 \geq 4$ is strongly NP-hard.

Proof. We reduce 3-PARTITION to our problem. Given an instance of 3-PARTITION, define an instance of $1|no-idle, \overline{d}_j = C_j^1 + T|V^2(\infty, c_2), direct|1|TC$ as follows. There are $4n$ orders, namely n *long* orders and $3n$ *short* orders. Short orders correspond to the integers of 3-PARTITION, so $p_i^1 = a_i$, $i = 1, \ldots, 3n$. Each long order has processing time $Q >> W$. We let $T = Q+2W$, hence the deadline of an order is $C_j^1 + T = C_j^1 + (Q+2W)$. Transportation time is $\ell_2 = Q+W$. We want to know whether there exists a production-distribution schedule (σ, θ) consisting of n batches that respects all due dates.

Suppose that a solution Σ to 3-PARTITION exists. Then we can build a production-distribution schedule (σ, θ) for 3PL provider as follows. We put in each batch one long order, followed by one of the triples of Σ. Doing so, the total processing time of each batch is exactly $Q + W$, and so a long order j which completes at C_j^1 will be delivered exactly at $C_j^1 + Q + 2W$, hence on time. Of course, also the three short orders will be on time. So, the obtained schedule (σ, θ) is a solution to 3PL provider.

Now suppose that a solution (σ, θ) to 3PL provider exists, i.e., (σ, θ) consists of n batches and respects all due dates. The following facts concerning (σ, θ) hold.

Fact 1. Each batch contains exactly one long order. In fact, if in a batch there are two long orders, the first of them would be certainly delivered late.

Fact 2. Each batch contains exactly three short orders. If not, there is at least one batch with four short orders, let $\tilde{p}$ be their total processing time. From Lemma 3.7, the long order is the first in the batch, and it would be delivered after $\tilde{p} + Q + W$ from its completion time. Since $W/4 < a_i$, $\tilde{p} > W$. Hence, the long order would be late.

Fact 3. The total processing time of the short orders in each batch equals W. If this is not the case, there would be at least one batch in which the total processing time of the short orders would exceed W and hence, similarly to Fact 2, the long order in this batch would be delivered beyond its due date.

In conclusion, the short orders in each batch define a triple which constitutes a solution to 3-PARTITION. □

Now let us turn to the case $c_2 = 3$. For the sake of simplicity, we suppose that in a schedule with b batches, there are $3b - n$ dummy orders with zero processing time, so that we can regard each batch as consisting of exactly three orders.

We can prove that the following properties hold for the problem $1|no - idle, \overline{d}_j = C_j^1 + T|V^2(\infty, 3), direct|1|TC$. In what follows, we suppose the orders numbered in the longest precessing time (LPT) order, i.e., $p_1^1 \geq \ldots \geq p_n^1$.

Lemma 3.8 For problem $1|no-idle, \overline{d}_j = C_j^1 + T|V^2(\infty, 3), direct|1|TC$, there exists an optimal production-distribution schedule, consisting of b batches, such that the following properties hold:

1. The orders $1, \ldots, b$ are the first orders in their respective batch;

2. For each $i = 0, \ldots, b-1$, the orders $b + 1 + i$ and $3b - i$ are in the same batch.

Proof. Property 1: The proof of this property is illustrated by Figure 3.6. Consider an optimal schedule (σ, θ), and suppose that there is a batch B with an order i such that $1 \leq i \leq b$, and i is *not* the first order in B. Since there are b batches, there is at least one batch, say B', in which the first order has index $j > i$. Now swap the orders i and j among B and B' in θ, and update the departure time of each batch by the completion of each batch, hence obtaining a new schedule $(\hat{\sigma}, \hat{\theta})$ with the same number of batches.

Note that $(\hat{\sigma}, \hat{\theta})$ is feasible. In fact, the first order of B is the same in the two schedules, and the difference between the departure time of B and the completion time of the first order of B is smaller in $\hat{\theta}$, so B is on time in $(\hat{\sigma}, \hat{\theta})$. Concerning B', the first order is longer in $(\hat{\sigma}, \hat{\theta})$ than in (σ, θ) by $\eta = p_i^1 - p_j^1$, so the departure time increases by η. However, the deadline of the first order increases also by η, so also B' is on time in $(\hat{\sigma}, \hat{\theta})$. We can repeat the same argument for each order k such that $1 \leq k \leq b$ and which is not the first order in its batch, until an optimal schedule respecting Property 1 is obtained.



Figure 3.6: Illustration of property 1 of Lemma 3.8.

Property 2: Let (σ, θ) be an optimal production-distribution schedule which respects the previous Property 1. Suppose that $b+1+i$ and $3b-i$ are not in the same batch, and $b+1+i$ is the smallest-indexed order which is not in the same batch with $3b-i$, for $i \leq b-1$. Let k denote the order in the same batch with $b+1+i$, and k' the order in the same batch with $3b-i$. Note that $p_{b+1+i}^1 \geq p_k^1 \geq p_{3b-i}^1$ and $p_{b+1+i}^1 \geq p_{k'}^1 \geq p_{3b-i}^1$. We can exchange order k with order $3b-i$ in θ, and update the departure time of each batch by the completion of each batch, without changing the number of batches, hence obtaining a new schedule $(\hat{\sigma}, \hat{\theta})$. Again,$(\hat{\sigma}, \hat{\theta})$ is feasible. In fact, since $p_{b+1+i}^1 + p_{3b-i}^1 \leq p_{b+1+i}^1 + p_k^1 \leq T - \ell_2$, the batch containing $b+1+i$ and $3b-i$ is on time. The batch containing k and k' is also on time, since $p_{k'}^1 + p_k^1 \leq p_{b+1+i}^1 + p_k^1 \leq T - \ell_2$. One can repeatedly apply the same argument until a schedule with the same number of batches is obtained that respects Property 2. □

According to Lemma 3.7 and Lemma 3.8, we propose a greedy algorithm GA3.2 for the problem $1|no-idle, \overline{d}_j = C_j^1 + T|V^2(\infty, 3), direct|1|TC$.

Theorem 3.9 Algorithm GA3.2 finds an optimal production-distribution schedule for problem $1|no-idle, \overline{d}_j = C_j^1 + T|V^2(\infty, 3), direct|1|TC$ in $O(n^2)$ time.

Algorithm 2: Algorithm GA3.2

1 Index orders in LPT order, so that $p_1^1 \geq \ldots \geq p_n^1$;
2 $b = \lceil \frac{n}{3} \rceil$; //b is the initial number of batches
3 Generate $3b - n$ dummy orders with 0 processing time and add them at the end of the list of orders;
4 **for** $j = 1$ *to* b **do**
5 $\quad B_j = \{j\}$; //put order j in the batch B_j
6 $i = 0$;
7 **while** $i < b$ **do**
8 $\quad$ **if** $p_{b+1+i}^1 + p_{3b-i}^1 > T - \ell_2$ **then**
9 $\quad\quad B_{b+1} = \{b+1\}$; //there is no feasible solution with b batches, then we create a new batch with order $b+1$
10 $\quad\quad$ Generate 3 new dummy orders with 0 processing time and add them at the end of the list of orders;
11 $\quad\quad b = b + 1$; $i = 0$;
12 $\quad$ **else**
13 $\quad\quad i = i + 1$;
14 **for** $i = 0$ *to* $b - 1$ **do**
15 $\quad B_{i+1} = B_{i+1} \cup \{b + 1 + i, 3b - i\}$; //put orders $b + 1 + i$ and $3b - i$ in the batch B_{i+1}
16 Order the batches randomly to obtain a production-distribution schedule;

Proof. In the algorithm GA3.2, the first step (line 1) requires $O(n \log n)$ time. The second (lines 2-5) and the fourth step (line 16) require $O(n)$ time. The checking action in the third step (lines 6-13) requires $O(n)$ time and in the worst case we need to check $O(n)$ times. The assignment in the third step (lines 14-15) requires $O(n)$ time. So the algorithm GA3.2 finds an optimal schedule for problem $1|no - idle, \overline{d}_j = C_j^1 + T|(\infty, 3), direct|1|TC$ in $O(n^2)$ time. □

We next propose a greedy algorithm GA3.3 for the problem $1|no - idle, \overline{d}_j = C_j^1 + T|V^2(\infty, 2), direct|1|TC$. In this case, with a similar proof, the Property 1 of Lemma

3.8 holds for the problem with $c_2 = 2$.

Algorithm 3: Algorithm GA3.3

1 Index orders in LPT order, so that $p_1^1 \geq \ldots \geq p_n^1$;
2 $b = \lceil \frac{n}{2} \rceil$; //b is the initial number of batches
3 **for** $j = 1$ *to* b **do**
4 $\quad B_j = \{j\}$; //put order j in the batch B_j
5 $i = b + 1$;
6 **while** $i \leq n$ **do**
7 $\quad$ **if** $p_i^1 > T - \ell_2$ **then**
8 $\quad\quad B_{b+1} = \{b+1\}$; //there is no feasible solution with b batches, then we create a new batch with order $b+1$
9 $\quad\quad b = b + 1$; $i = b + 1$;
10 $\quad$ **else**
11 $\quad\quad i = n + 1$;
12 **for** $i = b + 1$ *to* n **do**
13 $\quad B_{i-b} = B_{i-b} \cup \{i\}$; //put order i in the batch B_{i-b}
14 Order the batches randomly to obtain a production-distribution schedule;

Theorem 3.10 Algorithm GA3.3 finds an optimal production-distribution schedule for problem $1|no-idle, \overline{d}_j = C_j^1 + T|V^2(\infty, 2), direct|1|TC$ in $O(n \log n)$ time.

Proof. In the algorithm GA3.3, the first step (line 1) requires $O(n \log n)$ time. The second step (lines 2-4) and the forth step (line 14) require $O(n)$ time. The third step (lines 5-13) requires $O(n)$ time. So the algorithm GA3.3 finds an optimal schedule for problem $1|no-idle, \overline{d}_j = C_j + T|V^2(\infty, 2), direct|1|TC$ in $O(n \log n)$ time. □

Finally, for $c_2 = 1$, the problem $1|no-idle, \overline{d}_j = C_j + T|V^2(\infty, 1), direct|1|TC$, is trivially solved in $O(n)$ by randomly ordering the orders on M_1, and delivering each order by a different vehicle as soon as it is finished.

(b) Problems with express transportation only and one vehicle for express transportation

Let us now consider the case in which there is only *one* vehicle which can carry only one order, i.e., problem $1|no-idle, \overline{d}_j = C_j^1 + T|V^2(1,1), direct|1|TC$. In this case, batches correspond indeed to orders, and each round trip of the vehicle concerns the delivery of a single order, i.e. the transportation cost of any feasible solution is $TC = h_2 n$. Hence, the only issue is to check whether there exists a feasible production-distribution schedule.

A schedule for this problem has a very distinctive structure, in which we call *offset* O_j the difference between the departure time of the vehicle that carries away order j (equal to $D_j - \ell_2$) and the completion time C_j^1 of the same order on M_1. Note that by definition the offset can never be negative. The offset is zero if an order is completed on M_1 after (or exactly at) the time the vehicle is back in M_1. It is easy to check that in general, if we let i be the order that immediately precedes j in a schedule:

$$O_j = \max\{0, O_i - p_j^1 + \ell_2 + \ell_2'\}. \tag{3.5.3}$$

Note that if $p_j^1 > \ell_2 + \ell_2'$, then the offset decreases (by $\ell_2 + \ell_2' - p_j^1$), while on the contrary, if $p_j^1 < \ell_2 + \ell_2'$, then the offset increases (by $\ell_2 + \ell_2' - p_j^1$). No order can be scheduled if the offset becomes larger than $T - \ell_2$, so in a feasible schedule all the offset values are below this value.

To establish the complexity of the problem, we use the following strongly NP-complete problem (Kellerer et al 1998):

Stock Size Problem (SSP). Given a set I of n nonzero integers $a_1, \ldots, a_n$, and a positive integer Q such that $\sum_{j \in I} a_j = Q$, is there an ordering σ of the n integers such that, denoting by $\sigma(i)$ the i-th integer in the ordering, for all $k = 1, \ldots, n$ it holds

$$0 \leq \sum_{i=1}^{k} a_{\sigma(i)} \leq Q? \tag{3.5.4}$$

Theorem 3.11 Problem $1|no-idle, \overline{d}_j = C_j^1 + T|V^2(1,1), direct|1|TC$ is strongly NP-hard.

Proof. We reduce SSP to our problem. Given an instance of SSP, we let $\mathcal{P}$ and $\mathcal{N}$ denote positive and negative integers in I respectively. We let $\ell_2 = \ell_2' = Q + 1$ and $T = 2Q + 1$. For each a_i, define an order i having processing time $p_i^1 = 2\ell_2 - a_i$ (note that $p_i^1 < 2\ell_2$ if

$i \in \mathcal{P}$ and $p_i^1 > 2\ell_2$ if $i \in \mathcal{N}$). We call $J_\mathcal{P}$ and $J_\mathcal{N}$ the order sets corresponding to $\mathcal{P}$ and $\mathcal{N}$ respectively. Finally, add a dummy order having $p_0^1 = \varepsilon > 0$, where ε is very small. Note that a schedule is feasible iff the offset never exceeds $T - \ell_2 = Q$. We want to show that a feasible schedule exists iff a solution to SSP exists.

Given a solution to SSP, a feasible schedule can be built as follows. The first order is the dummy order, which has therefore to be carried away. We then sequence the orders in the same order as in the solution to SSP. Each order $i \in J_\mathcal{P}$ increases the offset by a_i, while each order $i \in J_\mathcal{N}$ decreases it by a_i. After the k-th order, the value of the offset is precisely the value $\sum_{i=1}^{k} a_{\sigma(i)}$. From equation (3.5.4), for all k it does not exceed Q, and the sequence is therefore feasible.

Now viceversa, suppose that we have a feasible production-distribution schedule (σ, θ). First of all, the dummy order is certainly at the beginning of the schedule, and the initial offset is $O_{\sigma(0)} = 0$. Consider now the n equations (3.5.3):

$$\begin{aligned}
O_{\sigma(1)} &= \max\{0, 2\ell_2 - p^1_{\sigma(1)}\} = \max\{0, a_{\sigma(1)}\} \\
O_{\sigma(2)} &= \max\{0, O_{\sigma(1)} + 2\ell_2 - p^1_{\sigma(2)}\} = \max\{0, O_{\sigma(1)} + a_{\sigma(2)}\} \\
O_{\sigma(3)} &= \max\{0, O_{\sigma(2)} + 2\ell_2 - p^1_{\sigma(3)}\} = \max\{0, O_{\sigma(2)} + a_{\sigma(3)}\} \\
&\dots \\
O_{\sigma(n)} &= \max\{0, O_{\sigma(n-1)} + 2\ell_2 - p^1_{\sigma(n)}\} = \max\{0, O_{\sigma(n-1)} + a_{\sigma(n)}\}
\end{aligned}$$

Suppose that $O_{\sigma(i)} + a_{\sigma(i+1)} < 0$ for some i, $0 \le i \le n-1$. Summing up all equations, one would get that the final offset $O_{\sigma(n)} > \sum_{i=1}^{n} a_{\sigma(i)} = Q$, which would make it impossible to deliver $\sigma(n)$ within T. This would contradict the feasibility of (σ, θ). Hence, $O_{\sigma(i)} + a_{\sigma(i)} \ge 0$ for all i. This implies that the vehicle is never idle throughout the schedule. As a consequence, the value of the offset after each order k is given precisely by $\sum_{i=1}^{k} a_{\sigma(i)}$ which never exceeds Q, since (σ, θ) is feasible. Hence, equation (3.5.4) is satisfied, i.e., the corresponding sequence of integers is feasible for SSP. □

(c) Problems with regular transportation only

The last problem we consider concerns the cases in which only regular transportation exists. We next show that the problem is difficult, even if the number of vehicles v_s for each known fixed departure time t_s is unlimited.

Theorem 3.12 Problem $1|no-idle, \overline{d}_j = C_j^1 + T|V^1(\infty, c_1), direct|1|TC$ is strongly NP-hard, for any value of c_1.

Proof. Given an instance of 3-PARTITION, define an instance of 3PL provider's problem $1|no-idle, \overline{d}_j = C_j^1 + T|V^1(\infty, c_1), direct|1|TC$ as follows. There are $4n$ orders, namely n *long* orders and $3n$ *short* orders. The short orders correspond to the integers of 3-partition, so $p_i^1 = a_i$ for $i = 1, \ldots, 3n$. All long orders have the same processing time $Q >> W$. We let $T = Q + 2W$, so the deadline of a order is $C_j^1 + T = C_j^1 + (Q + 2W)$. There are n fixed departure times $Q+W, 2(Q+W), \ldots, n(Q+W)$. Transportation time is $\ell_1 = Q + W$. For different values of capacity, we want to establish whether there is a feasible production-distribution schedule σ with a number of batches given by n (if $c_1 \geq 4$), $2n$ (if $c_1 = 2$ or $c_1 = 3$), or $4n$ (if $c_1 = 1$).

Suppose that a solution Σ to 3-PARTITION exists. Then we can build a production schedule (σ, θ) starting with a long order and in which long orders and triples of small orders alternate. For what concerns the delivery schedule:

- $c_1 \geq 4$. In this case we define n batches, each consisting of one long order followed by the subsequent triple of Σ. Doing so, the total processing time of each batch is exactly $Q + W$, and each batch departs at its completion time, which is precisely one fixed departure time. Each long order is delivered exactly at its deadline.

- $c_1 = 2$ or $c_1 = 3$. In this case we define $2n$ batches, obtained by splitting in two each batch defined in the previous case. One has therefore that at each fixed departure time $i(Q+W)$, $i = 1, \ldots, n$, two vehicles start, each carrying two orders (one vehicle will transport the long order and one small order, the other vehicle will transport the other two small orders). Both batches are on time.

- $c_1 = 1$. In this case we simply define one batch for each order, so that at each time $i(Q + W)$, four vehicles depart from M_1.

In all cases, the obtained production-distribution schedule is a solution to the 3PL provider's problem.

Now suppose that a feasible production-distribution schedule to 3PL provider exists. The following facts concerning the corresponding production schedule (σ, θ) hold, for any value of the capacity c_1.

Fact 1. In (σ, θ), exactly one long order is entirely processed between $(i-1)(Q+W)$ and $i(Q+W)$, for $1 \leq i \leq n$. In fact, if a long order starts before some $i(Q+W)$ but ends after $i(Q+W)$ for some i, it would be certainly delivered late, since it could not depart before $(i+1)(Q+W)$, and therefore the long order should wait more than W before leaving. With no loss of generality, we can therefore suppose that the long order starts exactly at $i(Q+W)$, for $0 \leq i \leq n-1$.

Fact 2. In (σ, θ), each long order is followed by exactly three short orders, and the total processing time of these three short orders equals W. Suppose in fact that there exists a long order followed by a set of short orders whose total processing time is not W. Then, there exists a long order j followed by short orders whose total processing time *exceeds* W. As a consequence of Fact 1, the last of these short orders, say k, cannot depart in the same batch of the long order j. But the next departure time is too far and order k could not be delivered within its deadline. So, the total processing time of these orders is W, and since $W/4 < a_i < W/2$, there are exactly three short orders after each long order.

In conclusion, the short orders after each long order define a triple which constitutes a solution to 3-PARTITION. □

In theorem 3.12, we observe that in the instance of 3PL provider's problem for each case we consider the minimum number of vehicles. So this proof applies to the 3PL provider's problem with limited number of vehicles for regular transportation. That means the 3PL provider's problem with limited number of vehicles for regular transportation is also strongly NP-hard.

3.5.2 Manufacturer's problem

As discussed in section 3.2, the manufacturer determines a production schedule on both machines minimizing the makespan $C_{\max}$ subject to a limited set of regular vehicles and express vehicles with given capacities and transportation times. The responsiveness constraints are removed. The manufacturer's problem with only one express vehicle is strongly NP-hard if $c_2 \neq 2$ (Lee and Chen 2001). The complexity of the problem with $c_2 = 2$ is still open. The manufacturer's problems with only regular vehicles or both

modes of vehicle are also strongly NP-hard proved in section 4.4 when manufacturer and 3PL provider coordinate.

3.6 Conclusions

In this chapter, we investigated a collection of models for production and interstage batch delivery scheduling problems in a supply chain, where a manufacturer processes a set of orders in two production locations and a 3PL provider delivers the semi-finished products from the upstream production location to the downstream production location. We considered the two scenarios: (1) manufacturer dominates, 3PL provider adjusts; (2) 3PL provider dominates, manufacturer adjusts. For each scenario, we investigated the manufacturer's problem and the 3PL provider's problem separately. We focused on the 3PL provider's problem. We provided some polynomial-time algorithms for the 3PL provider's problem when the manufacturer dominates, where the 3PL provider's problem is to determine a delivery schedule minimizing transportation costs, while respecting the orders release times and the deadline of each order imposed by the manufacturer. In particular, we considered several models with different transportation modes and transportation time. When the 3PL provider dominates, the 3PL provider's problem is to determine an optimal production-distribution schedule minimizing transportation cost, while respecting the deadline of each order. We proposed some polynomial-time algorithms for 3PL provider's problems when the 3PL provider dominates in some particular cases and demonstrate the NP-hardness for other 3PL provider's problems. Here we also considered several models with different transportation modes.

A number of important issues remain open. First, the complexities of 3PL provider's problems are open when the manufacturer dominates and the limited number of vehicles for express transportation is not fixed. Second, most of the 3PL provider's problems with 3PL provider dominating, are shown to be intractable, which motivates the need to develop exact or approximate solution algorithms. The coordinated production and interstage distribution scheduling problems will be investigated in the next chapter.

Chapter 4

Coordinated Production and Interstage Distribution Scheduling Problems

4.1 Introduction

Recall that in this part, we consider a production and interstage distribution scheduling problem in a permutation flow shop environment. A set of orders are processed by a manufacturer at the upstream facility, and delivered to the downstream facility belonging to the same manufacturer. The distribution is outsourced to a third-party logistics (3PL) provider.

Our objective is to investigate the production and interstage distribution scheduling problem in four scenarios: (1) manufacturer dominates, 3PL provider adjusts; (2) 3PL provider dominates, manufacturer adjusts; (3) manufacturer dominates, 3PL provider negotiates; (4) manufacturer and 3PL provider coordinate.

In chapter 3, we studied the individual scheduling problems in scenarios (1) and (2). In this chapter, we investigate the coordinated scheduling problems in scenarios (3) and (4). For the scheduling problems in each scenario, we provide exact polynomial-time algorithms or prove their NP-hardness. We provide two mechanisms of coordination for scenarios (3) and (4) and evaluate the benefit of coordination using numerical experiments.

Chapter 4 is organized as follows. In section 4.2, we formally describe the problems and introduce notations and terminology. Section 4.3 is devoted to scenario (3), section 4.4 to scenario (4). In section 4.5, we evaluate the benefit of coordination in scenarios (3) and (4). Section 4.6 contains some conclusions and perspectives.

4.2 Problems and Notations

In chapter 3, we considered several models, i.e. one transportation mode or both transportation modes, limited or unlimited number of vehicle, unique or different transportation time. In this chapter, we consider only one model: there is a limited number of regular vehicles and an unlimited number of express vehicles, and the transportation times of the two transportation modes are identical.

The problem was formally described in section 3.2 of chapter 3. In this chapter, we change two notations:

- use τ to replace the transportation times of the two transportation modes, i.e. $\ell_1 = \ell_2 = \tau$.
- use V instead of V_1 to represent the number of regular vehicles.

We consider two *scenarios*: (3) manufacturer dominates, 3PL provider negotiates; (4) manufacturer and 3PL provider coordinate. These are formally defined in the following.

1. **Manufacturer dominates, 3PL provider negotiates - scenario 3.**
   (a) **Manufacturer's problem.** It is the same as the manufacturer's problem in scenario (1) (see section 3.2 of chapter 3).
   (b) **3PL provider's problem.** Given the completion times C_j^1 and the deadlines $\overline{d}_j = C_j^1 + T$, the 3PL provider determines a delivery schedule that minimizes the transportation cost TC. Unlike scenario (1), the 3PL provider is allowed to negotiate with the manufacturer. We assume that the sequence of production cannot be changed while responsiveness constraints $D_j \leq C_j^1 + T$ can be relaxed. The 3PL provider's problem includes: establishing a mechanism of coordination and determining a delivery schedule under the mechanism of coordination.

2. **Manufacturer and 3PL provider coordinate - scenario 4.** In this scenario we suppose that the production schedule and the delivery schedule are to be jointly decided by the two actors. The responsiveness constraints $D_j \leq C_j^1 + T$ are removed. The objective is to find *efficient* production-distribution schedules, i.e., such that no better solution for one actor can be obtained without increasing the cost to the other actor. In order to motivate the coordination, it is necessary to establish a mechanism of coordination.

Example We consider the same example of section 3.2. Recall that, in scenario (1) manufacturer dominates and 3PL provider adjusts, we find the schedule (σ_1, θ_1) with $C_{\max}(\sigma_1, \theta_1) = 40$ and $TC(\sigma_1, \theta_1) = h_1 + 3h_2 = 31$.

1. **Manufacturer dominates, 3PL provider negotiates - scenario 3.** In this scenario, if the responsiveness constraints are relaxed, a better solution for 3PL provider can be found while guaranteeing that the makespan does not exceed $C_{\max}(\sigma_1, \theta_1) = 40$. Figure 4.1 illustrates a schedule (σ_3, θ_3). The production sequence in σ_3 is $\{1, 3, 2, 5, 4\}$. The θ_3 is composed of three batches $\{1, 3\}$,$\{2\}$ and $\{5, 4\}$. In it, $\{2\}$ is an express batch departing at time 20, while $\{1, 3\}$ and $\{5, 4\}$ are two regular batches, departing at times 15 and 30 respectively. We have $C_{\max}(\sigma_3, \theta_3) = 40$ and $TC(\sigma_3, \theta_3) = 2h_1 + h_2 = 27$.

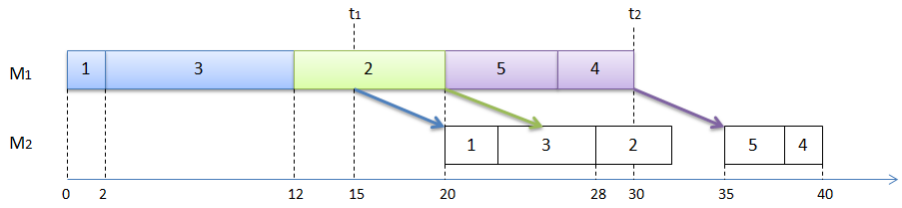


Figure 4.1: Production-distribution schedule when manufacturer dominates, 3PL provider negotiates.

2. **Manufacturer and 3PL provider coordinate - scenario 4.** When production and distribution schedules are decided by two actors together and the responsiveness constraints are removed, better solutions can be achieved for both actors. We find an efficient production-distribution schedules which are better than the bench-

mark schedule (σ_1, θ_1). This efficient solution (σ, θ) is found and depicted in Figure 4.2.

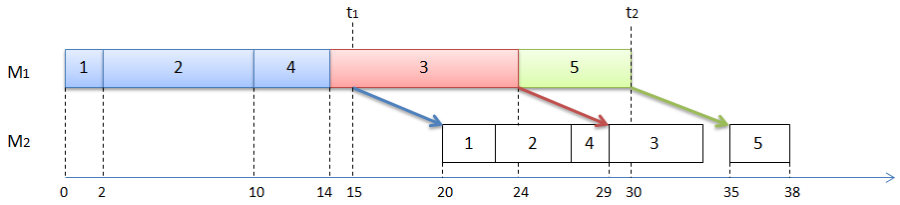


Figure 4.2: Production-distribution schedule when manufacturer and 3PL provider coordinate

The production sequence is $\{1, 2, 4, 3, 5\}$. There is one regular batch $\{1, 2, 4\}$, which departs at time 15, while the express batches $\{3\}$ and $\{5\}$ depart at time 24 and 30 respectively. Therefore, $TC(\sigma, \theta) = h_1 + 2h_2 = 24$ and $C_{\max}(\sigma, \theta) = 38$.

In this example, we observe that replacing responsiveness constraints with the makespan constraint is highly beneficial for the 3PL provider. In fact, comparing the solutions in scenario (1) and scenario (3), we get that the 3PL provider's benefit is $(TC(\sigma_1, \theta_1) - TC(\sigma_3, \theta_3))/TC(\sigma_1, \theta_1) \simeq 12.9\%$, while the manufacturer maintains the same makespan. Comparing (σ_1, θ_1) with (σ, θ), we get better solutions for both actors, since the 3PL provider's benefit is now $(TC(\sigma_1, \theta_1) - TC(\sigma, \theta))/TC(\sigma_1, \theta_1) \simeq 22.6\%$, and the manufacturer's benefit is $(C_{\max}(\sigma_1, \theta_1) - C_{\max}(\sigma, \theta))/C_{\max}(\sigma_1, \theta_1) \simeq 5.0\%$.

4.3 Manufacturer Dominates, 3PL Provider Negotiates - scenario 3

The manufacturer's problem is the same as that of scenario (1) (see section 3.2 of chapter 3), i.e. manufacturer dominates, 3PL provider negotiates. In the first step, the manufacturer adopts the Johnson's schedule on M_1 (Johnson 1954). In the second step, the manufacturer first imposes the constraints to the 3PL provider. Then the manufacturer adjusts the starting time of orders on M_2 while minimizing $C_{\max}$ subject to the delivery times D_j and the sequence of production on M_1. The schedule on M_2 is determined as follows: from the beginning to the end of sequence, schedule the orders one by one

as early as possible, i.e. the starting time of order j is the minimum time between its release date and the end of the previous scheduled order on M_2.

In scenario (1) the 3PL provider determines a distribution schedule subject to production schedule on M_1 (i.e. C_j^1) and deadlines $\overline{d}_j = C_j^1 + T$. In scenario (3), the actors can exchange necessary information to coordinate. We assume that the sequence of production cannot be changed while responsiveness constraints $D_j \leq C_j^1 + T$ can be relaxed. We provide a mechanism of coordination. Let $C'_{\max}$ denote the makespan obtained in the 3PL provider's problem of scenario (1).

Mechanism 1

1. The 3PL provider demands the manufacturer to use a performance guaranteeing constraint, i.e. $C_{\max} \leq C'_{\max}$, to replace the deadline constraints.

2. The 3PL provider determines a distribution schedule subject to the new constraint.

The 3PL provider's problem can be defined as: given the completion times C_j^1 for $j \in N$, the 3PL provider determines a distribution schedule that minimizes TC such that the makespan does not exceed the value obtained in scenario (1). This problem can still be solved through algorithm DP3.3 of chapter 3, provided that the responsiveness parameter is now different for each order. Precisely, numbering the orders according to the increasing completion time on M_1, i.e. $C_1^1 < \ldots < C_n^1$. we can define the deadline of each order as

$$\overline{d}_j = C'_{\max} - \sum_{u=j}^{n} p_u^2, \forall j = 1, \ldots, n. \tag{4.3.1}$$

It is easy to see that the makespan constraint is respected if and only if each order j is delivered to M_2 within the deadline $\overline{d}_j = C_j^1 + T_j$. In other words, the responsiveness parameter for order j is now $T_j = \overline{d}_j - C_j^1$ for $j = 1, \ldots, n$. This problem can be denoted by $1|fseq, \overline{d}_j = C_j^1 + T_j|V^1(V, c_1), V^2(\infty, c_2), \ell_1 = \ell_2 = \tau|1|TC$. Note that if $C_i^1 < C_j^1$, we have always $\overline{d}_i < \overline{d}_j$. As a consequence, we can prove that the properties of Lemma 3.1 in chapter 3 hold also for this 3PL provider's problem. Comparing this case with Scenario (1), we note that the only difference is that T is replaced by T_j in the recurrence relation of algorithm DP3.3 in chapter 3. So, such modified algorithm solves this 3PL provider's problem.

4.4 Manufacturer and 3PL Provider Coordinate - scenario 4

In this section we address the scenario in which the two parties can share information and define their schedules concurrently. Here we assume that the optimization criteria of manufacturer and 3PL provider are makespan and, respectively, total transportation cost. First we give a MILP(Mixed Integer Linear Programing) for the integrated scheduling problem that allows us generating *efficient* (or Pareto optimal) solutions, i.e., such that it cannot be improved on one criterion without being depreciated on the other one. These solutions can be used as input for negotiation between the manufacturer and the 3PL provider. Thereafter, we show that finding a production-distribution schedule is in general NP-hard, and we discuss some special cases, which are polynomially solvable. In order to motivate the coordination, a mechanism is established.

4.4.1 Mixed Integer Linear Programing

As discussed in Section 4.2, we consider the *permutation* problem, i.e., the production schedule has the same sequence on the two machines. We next give a MILP for the integrated scheduling problem that we used in our experiments (Section 4.5) to generate efficient solutions. Moreover, a simple order interchange argument can be used to change any efficient schedule to an efficient schedule such that the orders of each delivery batch are processed consecutively on machines. Hence, an efficient solution is completely characterized by a batching scheme, a transportation mode associated to each batch, and a starting time of each batch on the two machines. The sequence of orders within each batch is irrelevant.

We introduce the following notations.

$$u_{cs} = \begin{cases} 1, & \text{if the regular vehicle } c \text{ is available at the time } t_s, c = 1, \ldots, V, \\ & s = 1, \ldots, L \\ 0, & \text{otherwise} \end{cases}$$

Decision variables:

- t_k^e : starting time of batch k on machine M_e, $k = 1, \ldots, n, e = 1, 2$

- $x_{ik} = \begin{cases} 1, & \text{if order } i \text{ is in batch } k, i = 1, \ldots, n, k = 1, \ldots, n \\ 0, & \text{otherwise} \end{cases}$
- $w_k^m = \begin{cases} 1, & \text{if batch } k \text{ contains at least one order and is delivered by} \\ & \text{transport mode type-m}, k = 1, \ldots, n, m = 1, 2 \\ 0, & \text{otherwise} \end{cases}$
- $y_{kc} = \begin{cases} 1, & \text{if batch } k \text{ contains at least one order and is delivered by} \\ & \text{regular vehicle } c, k = 1, \ldots, n, c = 1, \ldots, V \\ 0, & \text{otherwise} \end{cases}$

$$\min C_{\max} \tag{4.4.2}$$

$$\min \sum_{k=1}^{n} w_k^1 h_1 + \sum_{k=1}^{n} w_k^2 h_2 \tag{4.4.3}$$

$$\text{s.t.} \quad \sum_{k=1}^{n} x_{ik} = 1, \quad i = 1, \ldots, n \tag{4.4.4}$$

$$\sum_{m=1}^{2} w_k^m \leq 1, \quad k = 1, \ldots, n \tag{4.4.5}$$

$$\sum_{i=1}^{n} x_{ik} \leq \sum_{m=1}^{2} c_m w_k^m, \quad k = 1, \ldots, n \tag{4.4.6}$$

$$t_{k+1}^1 - t_k^1 \geq \sum_{i=1}^{n} p_i^1 x_{ik}, \quad k = 1, \ldots, n-1 \tag{4.4.7}$$

$$t_{k+1}^2 - t_k^2 \geq \sum_{i=1}^{n} p_i^2 x_{ik}, \quad k = 1, \ldots, n-1 \tag{4.4.8}$$

$$t_k^2 - t_k^1 \geq \sum_{i=1}^{n} p_i^1 x_{ik} + \tau w_k^2, \quad k = 1, \ldots, n \tag{4.4.9}$$

$$\sum_{c=1}^{V} y_{kc} = w_k^1, \quad k = 1, \ldots, n \tag{4.4.10}$$

$$\sum_{k=1}^{n} y_{kc} \leq 1, \quad c = 1, \ldots, V \tag{4.4.11}$$

$$t_k^1 + \sum_{i=1}^{n} p_i^1 x_{ik} \leq \sum_{s=1}^{L} \sum_{c=1}^{V} t_s u_{cs} y_{kc} + (1 - w_k^1) \sum_{i=1}^{n} p_i^1, \quad k = 1, \ldots, n \tag{4.4.12}$$

$$t_k^2 \geq \sum_{s=1}^{L} \sum_{c=1}^{V} t_s u_{cs} y_{kc} + \tau w_k^1, \quad k = 1, \ldots, n \tag{4.4.13}$$

$$C_{\max} \geq t_n^2 + \sum_{i=1}^{n} p_i^2 x_{in} \tag{4.4.14}$$

$$t_k^1 \geq 0, \quad k = 1, \ldots, n \tag{4.4.15}$$

$$t_k^2 \geq 0, \quad k = 1, \ldots, n \tag{4.4.16}$$

$$x_{ik} \in \{0, 1\}, \quad i, k = 1, \ldots, n \tag{4.4.17}$$

$$w_k^m \in \{0, 1\}, \quad k = 1, \ldots, n, \quad m = 1, 2 \tag{4.4.18}$$

$$y_{kc} \in \{0, 1\}, \quad k = 1, \ldots, n, \quad c = 1, \ldots, V \tag{4.4.19}$$

In the MILP, (4.4.2) and (4.4.3) are the objective functions. Constraints (4.4.4) ensure that each order is assigned to one batch. Constraints (4.4.5) ensure that one batch either is delivered by one type of vehicle or does not have any order. Constraints (4.4.6) ensure that a vehicle does not carry more orders than its capacity. Constraints (4.4.7) and (4.4.8) guarantee the precedence constraints between batches and the assumption of permutation schedules. Constraints (4.4.9) represent the constraints of delivery time when one batch is delivered by an express vehicle. Constraints (4.4.10) ensure that one regular vehicle is assigned to a batch if it is not empty, i.e $w_k^1 = 1$. Constraints (4.4.11) ensure that at most one batch is assigned to a regular vehicle. Constraints (4.4.12) mean that any batch delivered by an regular vehicle should be available at the fixed departure date for this regular vehicle. The part $(1 - w_k^1) \sum_{i=1}^{n} p_i^1$ guarantees that if the batch is not delivered by one regular vehicle, these constraints are always true. Constraints (4.4.13) represent the constraints of delivery time when one batch is delivered by a regular vehicle. Constraints (4.4.14) define the makespan $C_{\max}$. Constraints (4.4.15) – (4.4.19) give the domain of definition of each variable.

Formulation (4.4.2) – (4.4.19) is used in our experiments to derive production-distribution schedules.

4.4.2 Complexity

In what follows we investigate the complexity of the problem of finding efficient production-distribution schedules. In particular, we consider the following integrated

scheduling problem (in decision form):

Given a set of n orders, having processing times p_j^1, p_j^2, a set $L = \{t_1, \ldots, t_L\}$ of departure dates of regular vehicles, a number v_s of vehicles available at time t_s, capacity values c_1 and c_2 for regular and express vehicles respectively, transportation costs h_1 and h_2 for regular and express vehicles respectively, and two positive integers P, H, is there a production-distribution schedule (σ, θ) such that $C_{\max}(\sigma, \theta) \leq P$ and $TC(\sigma, \theta) \leq H$?

Esswein et al. (2005) prove that the integrated scheduling problem, in the case where there is only express transportation mode with unlimited number of vehicles and unbounded capacity is NP-complete. For bounded capacity, Aloulou et al. (2014) prove that this problem is NP-complete when $c_2 \geq 3$ even if $p_j^1 = p_j^2 = p_j$ for $j = 1, \ldots, n$. However the problem with $c_2 = 2$ is open. Therefore, in the following, we consider the problem where there is only regular transportation mode with limited number of vehicles.

Theorem 4.1 The integrated scheduling problem (in decision form) is NP-complete, even if $p_j^1 = p_j^2 = p_j$ for $j = 1, \ldots, n$, $\tau = 0$ and there is only regular transportation mode.

Proof. We use the following NP-complete problem: 3-partition (Garey and Johnson 1979). Given $3n$ integers $a_1, \ldots, a_{3n}$, so that $\sum a_j = nW$, and such that $W/4 < a_j < W/2$ for all j, is it possible to partition them into n triples each summing up to W?

Given an instance of 3-partition, define an instance of integrated problem as follows. There are $4n$ orders, namely n long orders and $3n$ short orders. The short orders correspond to the integers of 3-partition, so $p_j^1 = p_j^2 = p_j = a_j$ for them. The long orders have the same processing time $p_j^1 = p_j^2 = p_j = Q >> W$. There is only regular transportation mode and transportation time τ equals to zero. There are n periodic fixed departure dates $Q + W, 2(Q + W), \ldots, n(Q + W)$. We distinguish different cases.

- case where $c_1 \geq 4$. For each fixed departure date, there is one regular vehicle. We want to know whether there exists a schedule (σ, θ) consisting of n batches, such that $C_{\max}(\sigma, \theta) \leq (n + 1)(W + Q)$ and $TC(\sigma, \theta) \leq nh_1$.

- case where $c_1 = 2$ or 3. For each fixed departure date, there are two regular vehicles. We want to know whether there exists a schedule (σ, θ) consisting of $2n$ batches, such that $C_{\max}(\sigma, \theta) \leq (n + 1)(W + Q)$ and $TC(\sigma, \theta) \leq 2nh_1$.

- case where $c_1 = 1$. For each fixed departure date, there are four regular vehicles. We want to know whether there exists a schedule (σ, θ) consisting of $4n$ batches, such that $C_{\max}(\sigma, \theta) \leq (n+1)(W+Q)$ and $TC(\sigma, \theta) \leq 4nh_1$.

Suppose that a solution Σ to 3-partition exists. Then we can build a schedule (σ, θ) for integrated problem in the three different cases as follows.

- case where $c_1 \geq 4$. We put in each batch one long order and one of the triples of Σ. At time 0 and each fixed departure date $Q+W, 2(Q+W), \ldots, (n-1)(Q+W)$, we schedule one batch on M_1 without idle time. The order of batches and the order of orders in each batch on M_1 are irrelevant. Doing so, the total processing time of each batch is exactly $Q+W$, and each batch departs at its completion time which is exactly one fixed departure date. Then we schedule each batch on M_2 immediately when it arrives and the orders of each batch are scheduled in the same sequence as on M_1. We have $C_{\max} = (n+1)(W+Q)$ et $TC = nh_1$.

- case where $c_1 = 2$ or 3. We apply the same rule as the case where $c_1 \geq 4$ and replace one batch by two batches. We have $C_{\max} = (n+1)(W+Q)$ et $TC = 2nh_1$.

- case where $c_1 = 1$. We apply the same rule as the case where $c_1 \geq 4$ and replace one batch by four batches. We have $C_{\max} = (n+1)(W+Q)$ et $TC = 4nh_1$.

So the obtained schedule (σ, θ) is a solution to integrated problem.

Now we suppose that a solution (σ, θ) to integrated problem exists. The following facts concerning (σ, θ) hold.

Fact 1. There does not exist idle time between orders on both machines. If there exists idle time on M_1, there would have at least one order exceed the last fixed departure date $n(Q+W)$, and we do not have any feasible solution in this case. We must have idle time at the beginning of M_2, which has length $Q+W$. Since the total processing time of orders on M_2 is $n(Q+W)$, so there does not exist idle time between orders on M_2.

Fact 2. At each fixed departure date, orders with total processing time which equals to $Q+W$ are delivered. If not, there would have idle time on M_2. So at each fixed departure date, the batch(es) is(are) formed by one long order and three short orders, i.e. the total processing time of the short orders equals to W.

In conclusion, the short orders delivered at each fixed departure date define a triple which constitutes a solution to 3-partition. □

4.4.3 Special cases

We consider a special case where all orders have a same processing time on M_1, i.e. $p_j^1 = p^1$, for $j = 1, \ldots, n$. The following property holds for this special case.

Lemma 4.1 If $p_j^1 = p^1$, for all $j = 1, \ldots, n$, for any efficient solution for the integrated scheduling problem, there exists one in which the orders are sequenced by non-increasing processing time on machine M_2.

Proof. Suppose there exists an efficient solution (σ, θ), such that order j is processed after order i on M_2, $p_j^2 > p_i^2$ and i and j are consecutive in σ. We can swap i and j on both machines and on their delivery batches, we denote this new solution as (σ', θ'). By doing so, we do not change the transportation cost, i.e. $TC(\sigma', \theta') = TC(\sigma, \theta)$. Since $C_i^2(\sigma') = C_j^2(\sigma)$, also the makespan is unchanged, i.e., $C_{\max}(\sigma', \theta') = C_{\max}(\sigma, \theta)$. Repeating this argument we eventually obtain a schedule having the same $C_{\max}$ and TC as schedule (σ, θ), and in which orders on M_2 are LPT (Longest Processing Time)-ordered. □

Lemma 4.1 states that in the special case of $p_i^1 = p^1$ for all $i = 1, \ldots, n$, we can assume that all efficient schedules respect the Johnson's schedule. In view of this result, we can index orders by non-increasing processing time on M_2, so that $p_1^2 \geq \ldots \geq p_n^2$ and the sequences on both machines follow the indexes of orders. The following dynamic programming algorithm allows to finding a schedule that minimizes $C_{\max}$ among all the schedules having a certain number of express and regular batches (and hence, a certain value of TC). In such algorithm we use the *state* $(i, j, s, v, m, n_1, n_2)$, corresponding to the situation in which:

- the first i orders have been delivered,
- the currently last batch consists of orders $\{j, j+1, \ldots, i\}$, and

  - if $m = 1$ (regular transportation), the currently last batch departs at t_s, and at this time, v regular vehicles are used (including the vehicle carrying the currently last batch),
  - if $m = 2$ (express transportation), the currently last batch departs at C_i^1.

- n_1 and n_2 regular and, respectively, express batches have been used so far.

We propose Algorithm 1 to solve this integrated problem.

Algorithm 1

Step 1: Index orders by non-increasing processing time on M_2.

Step 2: Apply the dynamic programming algorithm DP4.1, and get a set of feasible schedules associated to all values of n_1 and n_2.

Step 3: Remove the dominated schedules. The remaining schedules are efficient.

Algorithm DP4.1

Value Function

$f(i, j, s, v, m, n_1, n_2)$ = the minimum $C_{\max}$ of all schedules in state $(i, j, s, v, m, n_1, n_2)$.

Initialisation

$f(i, j, s, v, m, n_1, n_2) = \infty, i = 0, \ldots, n, j = 0, \ldots, n, s = 0, \ldots, L, v = 0, \ldots, v_s$,$m = 1, 2, n_1 = 0, \ldots, n, n_2 = 0, \ldots, n$.

Boundary Condition

$f(0, 0, 0, 0, 1, 0, 0) = 0,\ f(0, 0, 0, 0, 2, 0, 0) = 0$.

Optimal Solution Value

For each $n_1 = 0, \ldots, n, n_2 = 0, \ldots, n$, we search $\min_{(j,s,v,m)\in A} f(n, j, s, v, m, n_1, n_2)$ which is not infinity, and $A = \{(j, s, v, m) | 1 \leq j \leq n, 0 \leq s \leq L, 0 \leq v \leq v_s, m = 1, 2\}$.

Recurrence Relation

For state $(i, j, s, v, m, n_1, n_2)$, such that $i \in \{1, \ldots, n\}$, $j \in \{1, \ldots, i\}$, $s \in \{0, \ldots, L\}$, $v \in \{0, \ldots, v_s\}$, $m \in \{1, 2\}$, $n_1 =\in \{v, \ldots, n\}$, $n_2 =\in \{0, \ldots, n\}$, if $m = 1$, then $C_i^1 \leq t_s$, $s > 0$ and $v > 0$, if $m = 2$, then $n_2 > 0$.

$$f(i,j,s,v,m,n_1,n_2) =$$

$$\min \begin{cases} f(i-1,j,s,v,m,n_1,n_2)+p_i^2, & \text{if } i>j, m=1, i-j+1\le c_1 \\ \max\{f(i-1,j,s,v,m,n_1,n_2)-\sum_{u=j}^{i-1}p_u^2, & \\ C_i^1+\tau\}+\sum_{u=j}^{i}p_u^2, & \text{if } i>j, m=2, i-j+1\le c_2 \\ \max\{\min_{(j',m',s',v')\in A_1} f(i-1,j',s',v',m', & \\ n_1-1,n_2), t_s+\tau\}+p_i^2, & \text{if } i=j, m=1 \\ \max\{\min_{(j',m')\in A_2} f(i-1,j',s,v,m',n_1, & \\ n_2-1), C_i^1+\tau\}+p_i^2, & \text{if } i=j, m=2 \end{cases}$$

where

$A_1 = \{(j',m',s',v') | 1 \le j' \le i-1, m'=1,2$, if $v>1$, then $s'=s$ and $v'=v-1$, otherwise $0 \le s' < s$ and $0 \le v' \le v_{s'}\}$,

$A_2 = \{(j',m') | 1 \le j' \le i-1, m'=1,2\}$.

In the recurrence relation, the first term corresponds to the case where order i and order $i-1$ are in the same regular batch. The second term corresponds to the case where order i and order $i-1$ are in the same express batch. In the third term, order i is in a new regular batch. The function max is to find the earliest starting time on M_2 for order i. In the fourth term, order i is in a new express batch. The function max is to find the earliest starting time on M_2 for order i.

The step 1 of Algorithm 1 requires $O(n \log n)$ time. In algorithm DP4.1, there are $O(n^4(L+V))$ states. In fact, for each $s \in \{0,...,L\}$, v can assume v_s+1 distinct values. Since $\sum_s v_s = V$, one has $O(L+V)$ possible pairs (s,v). In the recurrence relation, the computation of the first and second term requires $O(1)$ time and is done for $O(n^4(L+V))$ states. The computation of the third and the fourth term requires $O(n(L+V))$ and $O(n)$ time respectively and is done for $O(n^3(L+V))$ states. Hence step 2 of Algorithm 1 requires $O(n^4(L+V)^2)$ time. The dynamic programming algorithm finds $O(n^2)$ feasible schedules. Step 3 of Algorithm 1 requires $O(n^4)$ time. Therefore, the complexity of Algorithm 1 is $O(n^4(L+V)^2)$.

A symmetric discussion holds in the special case in which all orders have the same processing time on M_2, i.e. $p_j^2 = p^2$, for $j = 1,\ldots,n$. With a similar proof to that

of Lemma 4.1, one can show that we lose no generality in considering efficient solution in which orders are sequenced by non-decreasing processing times on M_1. Using such sequencing to index the orders, so that $p_1^1 \leq \ldots \leq p_n^1$, Algorithm 1 can be applied to solve the problem also in this case. In conclusion, we have the following result.

Theorem 4.2 When either $p_j^1 = p^1$ for all $j = 1, \ldots, n$, or $p_j^2 = p^2$ for all $j = 1, \ldots, n$, the integrated scheduling problem can be solved in polynomial time $O(n^4(L+V)^2)$.

4.4.4 Mechanism of coordination

The bi-criteria integrated scheduling problem generates a set of efficient production-distribution schedules which can be used as input for negotiation between the manufacturer and the 3PL provider. We propose a win-win mechanism of coordination for the two actors to find a compromised solution with which the objective functions (i.e. $C_{\max}$, TC) do not exceed the values obtained in the non-coordinated relationship.

Mechanism 2

1. The two actors estimate a production-distribution schedule in the non-coordinated relationship according to the bargaining power of actor(i.e. scenarios (1) and (2)) and define it as a benchmark production-distribution schedule, i.e. if the proposed schedule generates a objective function grater than the benchmark, this schedule is rejected. Let (σ_1, θ_1) denote this benchmark schedule.

2. The two actors share the necessary information and try to find two sufficient solutions (σ_4, θ_4) and (σ_5, θ_5):

   (a) In the first step, the actors optimize with respect to $C_{\max}$ and TC in lexicographic order:

      i. the manufacturer first seeks a schedule that minimizes $C_{\max}$, provided that the total cost to 3PL provider does not exceed the value in the benchmark schedule, i.e., we seek a schedule (σ', θ') such that

$$C_{\max}(\sigma', \theta') = \min\{C_{\max}(\sigma, \theta) | TC(\sigma, \theta) \leq TC(\sigma_1, \theta_1)\};$$

ii. thereafter, the 3PL provider seeks a distribution schedule (σ_4, θ_4) such that

$$TC(\sigma_4, \theta_4) = \min\{TC(\sigma, \theta) | C_{\max}(\sigma, \theta) \leq C_{\max}(\sigma', \theta')\}.$$

(b) In the second step, the lexicographic order is reversed, i.e.:

i. first the 3PL seeks a schedule that minimizes TC, provided that the makespan does not exceed the value in the benchmark schedule, i.e.,

$$TC(\sigma'', \theta'') = \min\{TC(\sigma, \theta) | C_{\max}(\sigma, \theta) \leq C_{\max}(\sigma_1, \theta_1)\};$$

ii. thereafter, the manufacturer computes a schedule (σ_5, θ_5) such that

$$C_{\max}(\sigma_5, \theta_5) = \min\{C_{\max}(\sigma, \theta) | TC(\sigma, \theta) \leq TC(\sigma'', \theta'')\}.$$

3. If (σ_4, θ_4) and (σ_5, θ_5) are the same solutions, the two actor accept this schedule. Otherwise, the actor with strong bargaining power proposes his preferred one between (σ_4, θ_4) and (σ_5, θ_5). The actor with weak bargaining power calculates the price of dominance, i.e., how much is he penalized by the dominating position of the other actor, even in a coordinated scenario. If the price of dominance is significant, the actor with weak bargaining power can propose the other one between (σ_4, θ_4) and (σ_5, θ_5) by recompensing the other actor.

Let $Price_of_Dominance_Manu$ and $Price_of_Dominance_3PL$ denote the price of dominance for the 3PL provider and the price of dominance for the manufacturer respectively.

$$Price_of_Dominance_Manu = \frac{C_{\max}(\sigma_5, \theta_5) - C_{\max}(\sigma_4, \theta_4)}{C_{\max}(\sigma_5, \theta_5)} \tag{4.4.20}$$

$$Price_of_Dominance_3PL = \frac{TC(\sigma_4, \theta_4) - TC(\sigma_5, \theta_5)}{TC(\sigma_4, \theta_4)} \tag{4.4.21}$$

4.5 Computational Results

In the considerations that follow, we evaluate the benefit of coordination in scenarios (3) manufacturer dominates, 3PL provider negotiates, and (4) manufacturer and 3PL

provider coordinate. Our benchmark schedule is (σ_1, θ_1), i.e., the schedule obtained in scenario (1) manufacturer dominates, 3PL provider adjusts. In what follows, let (σ_3, θ_3) denote the schedule obtained in Scenario (3), and recall that (σ_4, θ_4) and (σ_5, θ_5) denote the two schedules obtained in the mechanism 2 for the scenario (4).

We consider the relative benefit for the 3PL provider in Scenario (3):

$$\Gamma_{3pl}^{31} = \frac{TC(\sigma_1, \theta_1) - TC(\sigma_3, \theta_3)}{TC(\sigma_1, \theta_1)} \tag{4.5.22}$$

Such index is also a measure of how binding are responsiveness constraints, when keeping the production sequence fixed (to Johnson's sequence).

The computation of (σ_4, θ_4) and (σ_5, θ_5) allows to investigate the benefits of coordination in scenario (4). In particular, the ratios

$$\Gamma_{m}^{41} = \frac{C_{\max}(\sigma_1, \theta_1) - C_{\max}(\sigma_4, \theta_4)}{C_{\max}(\sigma_1, \theta_1)} \tag{4.5.23}$$

$$\Gamma_{3pl}^{41} = \frac{TC(\sigma_1, \theta_1) - TC(\sigma_4, \theta_4)}{TC(\sigma_1, \theta_1)} \tag{4.5.24}$$

and

$$\Gamma_{m}^{51} = \frac{C_{\max}(\sigma_1, \theta_1) - C_{\max}(\sigma_5, \theta_5)}{C_{\max}(\sigma_1, \theta_1)} \tag{4.5.25}$$

$$\Gamma_{3pl}^{51} = \frac{TC(\sigma_1, \theta_1) - TC(\sigma_5, \theta_5)}{TC(\sigma_1, \theta_1)} \tag{4.5.26}$$

express the benefits of coordination for the two parties with (σ_4, θ_4) and (σ_5, θ_5) respectively. We want to compare the various schedules by means of numerical experiments.

We generate the instances of our experiments as follows.

- Processing times p_i^1 and p_i^2 are randomly generated integers from $[1, 100]$.
- We consider five values for the number of orders n, i.e. $n \in \{5, 10, 20, 50, 100\}$.
- Vehicle capacities have been fixed as follows. For regular vehicles, we consider three capacity values, $c_1 \in \{2, 3, 5\}$. For express vehicles, we fix the capacity c_2 equal to $\lfloor \alpha c_1 \rfloor$, with $\alpha \in \{0.5, 1\}$.

- For what concerns the cost of each trip, we let $h_1 = 1$ in all experiments and use parameter β to express the relationship between h_1 and h_2, so that $h_2 = \beta \frac{c_2}{c_1} h_1$, $\beta \in \{1, 2\}$.

- The transportation time τ is a randomly generated integer from $[1, 100]$.

- The number of regular vehicles is set to $V = \lceil \epsilon \frac{n}{c_1} \rceil$, where ϵ is randomly generated in the interval $[0.6, 1.4]$. Letting $\overline{v}$ denote the *average* number of regular vehicles for each fixed departure date, we determine the number of fixed departure dates L as $L = \lceil \frac{V}{\overline{v}} \rceil$. We assign $\overline{v} - 1$ regular vehicles to fixed departure dates t_s where $s = 1, \ldots, \overline{v}L - V$, and $\overline{v}$ vehicles to each remaining fixed departure date. In the experiments we consider three values for $\overline{v}$, i.e., $\overline{v} \in \{1, 2, 3\}$. The L departure dates are periodically fixed, with period $t = \lceil \frac{\sum_{i=1}^{n} p_i^1}{L} \rceil$.

- In the responsiveness constraints, we determine the value of T like $T = \lceil \gamma(t + \tau) \rceil$, where γ is randomly generated in three intervals, $[0.6, 0.8]$, $[0.95, 1.05]$, $[1.2, 1.4]$.

Summarizing, we have 5 values for n, 3 values for c_1, 2 values for α, 2 values for β, 3 values for $\overline{v}$ and 3 values for γ. This yields a total of 540 settings. For each setting, we generate 10 instances. Hence, we generate 5400 instances in total. The dynamic programming algorithms are implemented in C++, and the MILP is implemented in C++ and Cplex V12.2. The experiments are carried out on a DELL 2.50GHz personal computer with 8GB RAM.

Tables 4.1–4.6 report the average values of various figures. Table 4.1 gives the average computational time. For scenario (4), we set 200 seconds as the time limit for solving a single MILP. The computation time required by the polynomial algorithms described in scenarios (1) and (3) is compatible with all realistic problem sizes. Computation times required by the solution of (4.4.2)–(4.4.19) grow more rapidly. Table 4.2 gives the percentage of instances that CPLEX could not solve within the time limit, and the relative objective gap values for $C_{\max}$ and TC. These gaps are upper bounds of the following measures $(C_{\max}(\sigma_4, \theta_4) - C_{\max}(\sigma_4^*, \theta_4^*))/C_{\max}(\sigma_4^*, \theta_4^*)$, $(TC(\sigma_4, \theta_4) - TC(\sigma_4^*, \theta_4^*))/TC(\sigma_4^*, \theta_4^*)$, $(C_{\max}(\sigma_5, \theta_5) - C_{\max}(\sigma_5^*, \theta_5^*))/C_{\max}(\sigma_5^*, \theta_5^*)$, $(TC(\sigma_5, \theta_5) - TC(\sigma_5^*, \theta_5^*))/TC(\sigma_5^*, \theta_5^*)$, where (σ_4^*, θ_4^*) and (σ_5^*, θ_5^*) are the optimal schedules in mechanism 2. Table 4.3 compares the relative benefit of coordination

for the two parties (given in (4.5.23) - (4.5.26)) when either of the two lexicographic optimization problems is solved. Table 4.4 shows the values of what we indicated as price of dominance. Table 4.5 illustrates the benefits of coordination under various experiment aggregations. Finally, Table 4.6 shows the benefit stemming from the relaxation of the responsiveness constraint, when keeping Johnson's sequence.

Table 4.1: Average computational times of execution of algorithms.

n	(σ_1, θ_1)	(σ_3, θ_3)	(σ_4, θ_4)	(σ_5, θ_5)
5	0.02	0.02	0.24	0.25
10	0.30	0.33	1.59	1.00
20	1.34	1.56	74.00	50.83
50	7.79	7.07	145.53	202.64
100	34.75	49.65	-	-

Table 4.2: Failure rates and relative gaps of MILP.

n	(σ_4, θ_4)			(σ_5, θ_5)		
	failure rate	**Gap of** $C_{\max}$	**Gap of** TC	**failure rate**	**Gap of** $C_{\max}$	**Gap of** TC
5	0.00%	$\leq$ 0.00%	$\leq$ 0.00%	0.00%	$\leq$ 0.00%	$\leq$ 0.00%
10	0.00%	$\leq$ 0.00%	$\leq$ 0.00%	0.00%	$\leq$ 0.00%	$\leq$ 0.00%
20	1.39%	$\leq$ 0.50%	$\leq$ 0.48%	0.37%	$\leq$ 0.27%	$\leq$ 0.38%
50	54.72%	$\leq$ 0.53%	$\leq$ 2.23%	27.87%	$\leq$ 1.39%	$\leq$ 2.79%

Table 4.3: Benefits of coordination in scenario (4)

		Manufacturer's benefit		**3PL's benefit**	
		Γ_m^{41}	Γ_m^{51}	Γ_{3pl}^{41}	Γ_{3pl}^{51}
Overall averages		3.19%	2.16%	1.82%	5.97%
n	5	3.16%	3.04%	0.26%	0.72%
	10	3.99%	3.02%	1.10%	4.20%
	20	3.37%	1.57%	2.73%	8.38%
	50	1.11%	0.53%	5.04%	12.37%

Table 4.4: Price of dominance for 3PL provider.

n	$Price_of_Dominance_3PL$
5	0.46%
10	3.24%
20	5.86%
50	4.56%

A few comments are in order.

- From Table 4.3, one can observe that coordination is beneficial to both actors. However, as the number of orders grows, the relative benefit of 3PL provider increases, and the benefit of manufacturer decreases. The manufacturer's benefits are close. The difference of the 3PL provider's benefits is significant in two efficient solutions (σ_4, θ_4) and (σ_5, θ_5). This difference is represented by the price of dominance defined in 4.4.21 and is shown in Table 4.4. We note that even when the two parties coordinate, the dominant role of the manufacturer results in a "price" (on the 3PL provider) which exceeds 4.5% in largest instances. Hence, the 3PL provider may recompense the manufacturer in order to choose (σ_5, θ_5) as the final solution.

Table 4.5: Benefits of coordination in scenario (4) under various experiment aggregations.

		Manufacturer's benefit Γ_m^{41}	**3PL's benefit** Γ_{3pl}^{51}
Overall averages		3.19%	5.97%
	$\alpha = 1, \beta = 1$	2.11%	2.03%
	$\alpha = 1, \beta = 2$	4.41%	12.82%
	$\alpha = 0.5, \beta = 1$	2.98%	0.02%
	$\alpha = 0.5, \beta = 2$	3.35%	8.19%
γ	[0.6,0.8]	2.64%	6.78%
	[0.95,1.05]	3.59%	6.52%
	[1.2,1.4]	3.36%	4.61%
$\overline{v}$	1	2.45%	10.22%
	2	3.46%	4.69%
	3	3.68%	2.90%

- From Table 4.5, we notice that β (expressing the ratio between the costs of the two transportation modes) has a major impact on such benefit. In particular, when all vehicles have the same capacity ($\alpha = 1$) and the transportation cost of an express vehicle is twice that of a regular vehicle ($\beta = 2$), the 3PL provider's benefit is high (12.82%). Also the manufacturer has higher benefits in this situation, but it is on the whole less sensitive to the values of α and β.

- Still from Table 4.5, the 3PL provider's benefit from coordination is larger when the responsiveness constraint is more binding (6.78% for smaller values of T, i.e. $\gamma \in [0.6, 0.8]$). In this respect, the manufacturer's benefit is less sensitive to the value of T.

- Still from Table 4.5, aggregating the data according to parameter $\overline{v}$, one observes that the dispersion of fixed departure dates has a significant effect on the benefits. For a given number V of regular vehicles, recall that when $\overline{v} = 1$ we have $L = V$ departure dates, i.e., all vehicles have distinct departure dates, while for $\overline{v} = 3$, $L = \lceil V/3 \rceil$. As the departure dates of regular vehicles are more distributed over time, the 3PL provider's benefit increases significantly while the manufacturer's benefit slightly decreases. This can be explained considering that coordination allows a better adjustment of the production schedule to the regular vehicle timetable.

Table 4.6: Benefit of relaxing responsiveness constraints in scenario (3).

n	**3PL's benefit** Γ^{31}_{3pl}
5	0.60%
10	3.05%
20	6.48%
50	9.93%
100	10.82%
$\gamma \in [0.6, 0.8]$	6.70%
$\gamma \in [0.95, 1.05]$	6.88%
$\gamma \in [1.2, 1.4]$	4.95%

- From Table 4.6, it is interesting to observe how binding are responsiveness constraints for the 3PL provider. For the same makespan, replacing responsiveness

constraints with the makespan constraint (i.e., moving from Scenario (1) to Scenario (3)) may allow the 3PL provider to save an average 10%. As one could expect, such saving is higher for smaller values of γ, which correspond to smaller (hence, more binding) values of T.

- Comparing the benefits with schedule (σ_3, θ_3) (see Table 4.3) and the benefits with (σ_5, θ_5) (see Table 4.6), we observe that if the two parties coordinate without any imposed constraints, the coordination can bring better benefits to both parties.

4.6 Conclusions

In this chapter, we considered a production and interstage distribution scheduling problem in a supply chain with a manufacturer and a 3PL provider. The manufacturer's objective is to minimize the makespan and the 3PL provider's objective is to minimize the transportation cost. We analyzed scenario (3) manufacturer dominates, 3PL provider negotiates. We provided a mechanism of coordination, and proposed polynomial algorithms to solve the scheduling problems. We then turned to scenario (4) manufacturer and 3PL provider coordinate, i.e., the two actors may concurrently decide the production-distribution schedule. We proved the NP-hardness for the integrated scheduling problem and provided polynomial-time algorithms for some special cases. We also established a win-win mechanism of coordination. We performed experiments to investigate the benefits accruing from coordination between the two parties in scenarios (3) and (4). The results showed that the relaxation of the transportation responsiveness can bring a significant benefit for the 3PL provider and the coordination without any imposed constraints can bring better benefits to both actors.

Several important research issues remain open for future investigation. A first important research direction is to develop a heuristic for the integrated scheduling problem. Another issue is to investigate the extended model with a three-stage supply chain including interstage and outbound distribution, where a manufacturer has to process a set of orders at the upstream stage and the downstream stage. And a 3PL provider is in charge of transportation of semi-finished products from the upstream stage to the downstream stage, and then finished products to clients.

Part II

Production and Outbound Distribution Scheduling

Chapter 5

Production and Outbound Distribution Scheduling Problems with Release Dates and Deadlines

5.1 Introduction

In this chapter, we study a production and outbound distribution scheduling problem with the consideration of order release dates and delivery deadlines. A manufacturer has to process a set of orders on a single production line and a 3PL provider deliver them in batches to a customer. The objective of the manufacturer is to determine a feasible production schedule in which the orders are completed before or at their deadline. The 3PL provider aims at deciding a delivery schedule minimizing the transportation cost. Without coordination, the manufacturer may impose a feasible production schedule which requires that most of the orders have to be delivered immediately after their completion. This may entail higher costs for the 3PL provider. The coordinated problem is motivated to decide an integrated schedule minimizing the transportation cost subject to the delivery deadlines.

We investigate the individual scheduling problems faced by the manufacturer and the 3PL provider in a consecutive order: the manufacturer decides the feasible production schedule subject to the deadlines, then the 3PL provider adjusts a delivery schedule minimizing the transportation cost subject to the deadlines and the given production

schedule. Then, we consider three coordinated scheduling problems with different ways how an order can be produced and delivered: non-splittable production and delivery (NSP-NSD) problem, splittable production and non-splittable delivery (SP-NSD) problem and splittable production and delivery (SP-SD) problem. For these scheduling problems, we provide a polynomial-time algorithm for some restricted versions of SP-NSD and SP-SD problems and a branch-and-bound algorithm for NSP-NSD problem which is NP-hard. We evaluate the performance of the branch-and-bound algorithm using numerical experiments.

The most related research was provided by Chen and Pundoor (2009). They investigated an integrated production and outbound distribution scheduling (IPODS) problem in a supply chain where a manufacturer needs to process a set of orders at a single production line, pack the completed orders to form delivery batches, and deliver them to a customer. They investigated the problems in similar scenarios. Different from their model, we consider the orders with equal size and release dates. The literature of the IPODS problem with release dates can be found in section 2.3 of chapter 2.

This chapter is organized as follows. In section 5.2, we formally describe the problems and introduce notations and terminology. Section 5.3 is devoted to the individual scheduling problems, and section 5.4 to the coordinated scheduling problems. In section 5.5, we evaluate the performance of the branch-and-bound algorithm used to solve NSP-NSD problem. Section 5.6 contains some conclusions and perspectives.

5.2 Problems and Notations

The manufacturer has to process a set of orders $N = \{1, \ldots, n\}$ on a single machine. Each order $j \in N$ has a release date r_j (the date when raw material is available to process order j), a processing time p_j and a delivery deadline $\overline{d}_j$. After processing on the machine, the orders can be grouped into batches of maximum size $c > 0$, corresponding to a full truck load, and then sent to the customer locations. The orders are unit sized, i.e. a truck can carry at most c orders at a time. The delivery operation is outsourced by the manufacturer to a 3PL provider that is supposed to be able to deliver any batch at any time. The batch is available to be delivered when all orders of this batch are

completed. The transportation time of a batch and the corresponding subcontracting cost are supposed to be independent on the batch constitution. Hence, we can assume without loss of generality that the transportation time is 0 and the transportation cost of a batch is a constant h. Hence, the delivery deadline is also the production deadline.

Let (σ, θ) denote the integrated schedule, where σ and θ are respectively the production schedule and the delivery schedule. In this integrated schedule, $C_j(\sigma)$ is the completion time of order j on the machine and $D_j(\theta)$ is the delivery time of order j to the customer location. Sometimes, we use C_j and D_j instead of $C_j(\sigma)$ and $D_j(\theta)$ to simplify the notations.

We consider two scenarios: (1) the manufacturer and the 3PL provider decide their individual schedule without coordination in a consecutive order (i.e. first the manufacturer, then the 3PL provider); (2) the manufacturer and the 3PL provider coordinate to decide concurrently an integrated schedule. These scheduling problems are formally defined as follows.

1. **Individual scheduling problems.**

   (a) **Manufacturer's problem.** The manufacturer's objective is to determine a feasible production schedule in which the orders are completed before or at their deadline. We investigate the problem in two cases:

   - *Non-splittable production (NSP) problem:* An order is non-preemptable (or non-splittable) in production. Using the three-field notation $\alpha|\beta|\gamma$ for machine scheduling problems (Graham et al. 1979), this problem can be denoted by $1|r_j, \overline{d}_j|-$.
   - *Splittable production (SP) problem*: An order can be split in production. This problem can be denoted by $1|r_j, pmtn, \overline{d}_j|-$.

   (b) **3PL provider's problem.** With the production schedule imposed by the manufacturer, the 3PL provider aims at deciding a delivery schedule minimizing the transportation cost TC. A delivery schedule is a partition of the orders into batches, along with the departure time for each batch. We investigate the problem in two cases:

- *Non-splittable delivery (NSD) problem:* A finished order must be delivered in one batch.
- *Splittable delivery (SD) problem*: A finished order can be split and delivered in several batches.

2. **Coordinated scheduling problems.** Clearly, an integrated model minimizing the transportation cost TC subject to the deadline constraints is the best way for the coordination. Since the implementation of coordination is simple, we concentrate on the resolution of the integrated scheduling problem.

   We consider the integrated problem in three cases with different ways how an order can be produced and delivered.

   - *Non-splittable production and delivery (NSP-NSD) problem:* An order is non-preemptable (or non-splittable) in production and a finished order must be delivered in one batch. Using the five-field notation proposed by Chen (2010), this problem can be denoted by $1|r_j, \overline{d}_j|V(\infty, c), direct|1|TC$, where $V(\infty, c)$ and *direct* mean that we consider the direct batch delivery by an unlimited number of trucks with the capacity of c.
   - *Splittable production and non-splittable delivery (SP-NSD) problem*: An order can be split in production, and a finished order must be delivered in one batch. This problem can be denoted by $1|r_j, pmtn, \overline{d}_j|\ V(\infty, c), direct|1|TC$.
   - *Splittable production and delivery (SP-SD) problem:* An order can be split in both production and delivery. This problem can be denoted by $1|r_j, pmtn, \overline{d}_j|V(\infty, c), direct, split|1|TC$.

   We do not consider the non-splittable production and splittable delivery (NSP-SD) problem, because according to Lemma 5.2 in section 5.3.2, for any feasible NSP production schedule, there exists an optimal delivery schedule which is a NSD schedule.

Example 1: To illustrate the integrated problems, we consider the following example with seven orders where the vehicle capacity c is equal to 2. Table 5.1 gives the orders' parameters.

Table 5.1: Example for the integrated problems

Order j	1	2	3	4	5	6	7
p_j	4	2	2	2	2	3	1
r_j	0	2	2	2	13	12	17
$\overline{d}_j$	12	5	12	12	16	18	19

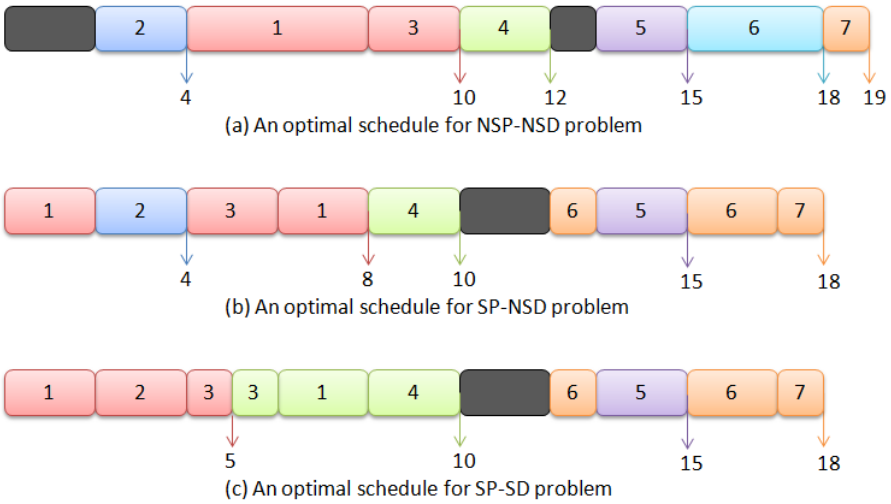


Figure 5.1: Optimal schedules for the integrated problems

Figure 5.1 shows the optimal schedules for the integrated problems. In a production schedule, $[j]$ means that order j is produced without preemption. In a delivery schedule, $[j]$ means that order j is delivered without splitting. When $[j]$ is preceded by a constant α, $0 < \alpha < 1$, this means that a part α of order j is produced or delivered.

NSP-NSD problem: In an optimal schedule as shown in Figure 5.1(a), the production sequence is $([2], [1], [3], [4], [5], [6], [7])$. There exists an idle time before order 2, because if another order is processed before 2, then order 2 would be late. A similar reason for the second idle time holds. There are six delivery batches: $\{[2]\}, \{[1], [3]\}, \{[4]\}, \{[5]\}, \{[6]\}$ and $\{[7]\}$, which depart respectively at time 4, 10, 12, 15, 18 and 19.

SP-NSD problem: In an optimal schedule as shown in Figure 5.1(b), the production sequence is $(\frac{1}{2}[1], [2], [3], \frac{1}{2}[1], [4], \frac{1}{3}[6], [5], \frac{2}{3}[6], [7])$, where orders 1 and 6 are split into two parts. The optimal schedule has five delivery batches: $\{[2]\}, \{[1], [3]\}, \{[4]\}, \{[5]\}$ and $\{[6], [7]\}$, which depart respectively at time 4, 8, 10, 15 and 18. Since order 2 cannot be

delivered with any other order, the transportation cost cannot be improved for the first 4 orders with the non-splittable delivery. However, we can split order 6 in production in order to deliver orders 6 and 7 in one batch.

SP-SD problem: In an optimal schedule as shown in Figure 5.1(c), the production sequence is the same as SP-NSD problem. The optimal schedule has four delivery batches: $\{\frac{1}{2}[1],[2],\frac{1}{2}[3]\}$, $\{\frac{1}{2}[3],\ \frac{1}{2}[1],[4]\}$, $\{[5]\}$ and $\{[6],[7]\}$, which depart respectively at time 5, 10, 15 and 18. For example, the first full filled delivery batch consists of half of order 1, whole order 2 and half of order 3. With the splittable delivery, the first four orders can be delivered in two full batches.

Remark that in the above problems, the orders delivered together are not necessarily sequenced consecutively, which makes the considered problems different from classical batching models.

Example 2: To illustrate the benefit of coordination, we consider the following example with five orders where the vehicle capacity c is equal to 3. Table 5.2 gives the orders' parameters.

Table 5.2: Example for evaluation of the benefit of coordination

Order j	1	2	3	4	5
p_j	8	2	8	6	2
r_j	2	10	6	1	12
$\overline{d}_j$	16	18	32	28	22

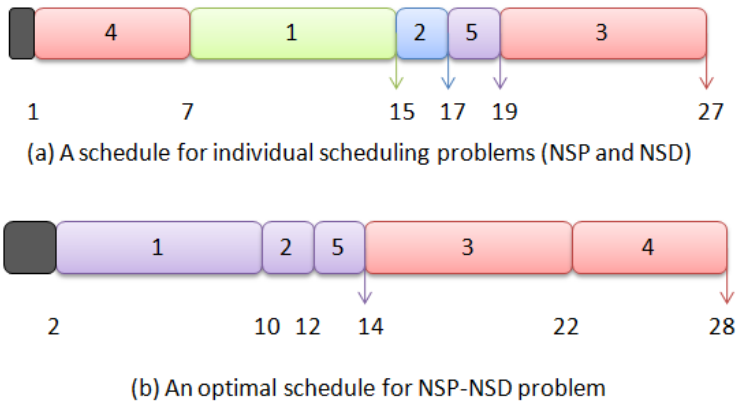


Figure 5.2: Schedules for the individual problems and the integrated problem

Figure 5.2(a) shows a feasible schedule for individual problems: NSP problem and NSD problem. Figure 5.2(b) shows an optimal schedule for NSP-NSD problem. We compare two schedules to evaluate the benefit of coordination.

Individual problems (NSP and NSD): Applying the Non-preemptive EDD rule (see section 5.3.1), we find a feasible production schedule. As shown in Figure 5.2(a), the production sequence is $([4], [1], [2], [5], [3])$. All orders are completed before their deadline. With this production schedule, the best distribution schedule consists of four delivery batches: $\{[1]\}, \{[2]\}, \{[5]\}$ and $\{[3], [4]\}$, which depart respectively at time 15, 17, 19 and 27.

NSP-NSD problem: In an optimal schedule as shown in Figure 5.2(b), the production sequence is $([1], [2], [5], [3], [4])$. The optimal schedule has two delivery batches: $\{[1], [2], [5]\}$ and $\{[3], [4]\}$, which depart respectively at time 14 and 28. Comparing with the schedule for individual problems, we observe that with the coordination, the transportation cost reduces 50%.

5.3 Individual Scheduling Problems

In the non-coordinated scenario, the manufacturer and the 3PL provider decide their individual schedule consecutively. We review known exact algorithms to solve the manufacturer' problems (i.e. NSP and SP problems) and develop exact algorithms to solve the 3PL provider's problems (i.e. NSD and SD problems).

5.3.1 Manufacturer's Problem

The manufacturer's objective is to determine a feasible production schedule in which the orders are completed before or at their deadline. We introduce first the definitions of production triplet (see definition 5.1) and production block (see definition 5.2). Then we investigate NSP and SP problems.

Definition 5.1 In a production schedule σ, a production triplet is an order or a part of order which is processed without preemption. Let $V_j(\sigma) = (J_j, a_j, b_j)$ denote production triplet j, where the order $J_j \in N$ is scheduled in the time interval $[a_j, b_j]$, a_j and b_j represent respectively the starting time and ending time of the triplet. Hence the production

schedule σ can be represented by a sequence of production triplets denoted by $V(\sigma)$.

Definition 5.2 In a production schedule σ, a production block is defined as a subset of orders which are processed consecutively. Set the minimum starting processing time of orders of the block as the starting time of the block and the maximum completion time of orders of the block as the ending time of the block. The sequence of orders is not taken into account in the definition of a block. Let $K_i(\sigma)$ denote the production block i in σ.

NSP problem In this problem, an order is non-preemptable (or non-splittable) in production. This decision problem $1|r_j, \overline{d}_j|-$ is NP-complete (Garey and Johnson 1979). Carlier proposed an efficient binary branch-and-bound algorithm to solve a head-tail problem where an order j is available for processing by the machine at release date r_j (called also head), and has to spend an amount of time p_j on the machine and an amount of time q_j (called tail) in the system after its processing, and the objective is to minimize $\max_{j\in N}(C_j + q_j)$. This problem is equivalent to the problem $1|r_j|L_{\max}$, where $L_{\max} = \max_{j\in N} L_j = \max_{j\in N}(C_j - d_j)$, L_j is the lateness and d_j is the due date (i.e. it can be violated). In fact, we can define $q_j = \max_{i\in N} d_i - d_j$, then minimizing $\max_{j\in N}(C_j + q_j)$ is equivalent to minimizing $L_{\max}$. The problem $1|r_j, \overline{d}_j|-$ is the decision version of the optimization problem $1|r_j|L_{\max}$, i.e. does there exist a production schedule σ such that $L_{\max}(\sigma) \leq 0$? NSP problem can be solved by applying Carlier's branch-and-bound algorithm and stopping when a feasible solution with $L_{\max} \leq 0$ is found.

We review the Carlier's branch-and-bound algorithm for the problem $1|r_j|L_{\max}$. The algorithm computes a lower bound and an upper bound for each node based on preemptive and non-preemptive EDD rule (Jackson 1955), respectively.

- *Preemptive EDD rule:* At each decision point t in time, consisting of each release date and each order completion time, schedule an available order j (i.e. $r_j \leq t$) with the earliest due date. If no orders are available at a decision point, schedule an idle time until the next release date.

- *Non-preemptive EDD rule:* At each decision point t in time, consisting of each starting time of production block and each order completion time, schedule an

available order j (i.e. $r_j \leq t$) with the earliest due date without preemption. If no orders are available at a decision point, schedule an idle time until the next release date.

At every node u, the algorithm constructs the non-preemptive EDD schedule and renumbers the orders according to the sequence in obtained schedule. Let l be the order with the smallest index such that $L_l = L_{\max}$. Let $h \leq l$ be the order with the largest index such that $h = 1$ or $C_{h-1} < s_h$ where s_h is the starting time of order h. Let $[h, l]$ denote the set of orders from h to l. If $d_l = \max_{k\in[h,l]} d_k$, then the obtained schedule is optimal. Otherwise, the algorithm defines a critical order $e \in [h, l]$ with the largest index such that $d_e > d_l$ and a set of critical set $J \in [e + 1, l]$. The algorithm considers two subsets of schedules corresponding to two nodes u_1 and u_2. Let $r_j(u)$ and $d_j(u)$ be the release date and the due date of order j at node u, respectively.

- In node u_1, the algorithm requires the critical order to be processed before the orders of the critical set by setting

$$d_e(u_1) = \max_{j\in J} d_j(u) - \sum_{j\in J} p_j \tag{5.3.1}$$

$$d_k(u_1) = d_k(u), \quad k \in N\backslash\{e\} \tag{5.3.2}$$

$$r_k(u_1) = r_k(u), \qquad k \in N \tag{5.3.3}$$

- In node u_2, the algorithm requires the critical order to be processed after the orders of the critical set by setting

$$r_e(u_2) = \min_{j\in J} r_j(u) + \sum_{j\in J} p_j \tag{5.3.4}$$

$$r_k(u_2) = r_k(u), \quad k \in N\backslash\{e\} \tag{5.3.5}$$

$$d_k(u_2) = d_k(u), \qquad k \in N \tag{5.3.6}$$

SP problem In this problem, the preemption is allowed in production. This problem $1|r_j, pmtn, \overline{d}_j|-$ is a decision problem corresponding to the optimization problem $1|r_j, pmtn|L_{\max}$ which is solved with the preemptive EDD rule in $O(n \log n)$ time (Horn 1974). Hence SP problem can be solved with the preemptive EDD rule in $O(n \log n)$ time. Since the preemption occurs only at release dates in the schedule generated with the preemptive EDD rule, there are at most $n - 1$ preemptions. Hence there are $O(n)$ production triplets in this production schedule.

5.3.2 3PL Provider's Problem

With the production schedule σ imposed by the manufacturer, the 3PL provider aims at deciding a delivery schedule minimizing the transportation cost. We assume that the orders are indexed in the increasing completion time, i.e. $C_1(\sigma) < \ldots < C_n(\sigma)$. This sorting operation requires $O(n \log n)$ time. Here, σ can be a NSP schedule or a SP schedule. We recall that there are $O(n)$ production triplets in σ (see section 5.3.1). We first provide a general property for NSD and SD problems. Then we investigate NSD and SD problems separately.

Lemma 5.1 There exists an optimal solution for NSD and SD problems, such that each batch is delivered at its completion time, i.e. when all orders (or parts of orders) of the batch are completed.

Proof. Suppose that there is an optimal delivery solution for NSD and SD problem, such that it does not respect the property. We can anticipate the delivery time of each batch to its completion time without changing the number of delivery batches. □

NSD problem In this case, a finished order must be delivered in one batch. We propose a polynomial-time algorithm (see algorithm GA5.1) for NSD problem.

Algorithm GA5.1

Step 1: Let $N' \subseteq N$ denote the set of undelivered orders. Set the current delivery time $T = \max_{j \in N'} C_j(\sigma)$.

Step 2: Find the set of undelivered orders with deadline greater than or equal to T. Let $S \subseteq N'$ denote this set.

Step 3: If $|S| < c$, deliver all orders of S in one batch which departs at time T. Otherwise, deliver the last c completed orders of S in one delivery batch which departs at time T. Then, update N'. If all orders are delivered, then STOP. Otherwise, go to step 1.

Theorem 5.1 Algorithm GA5.1 finds an optimal delivery schedule for NSD problem in $O(n^2)$ time.

Proof. We first prove the complexity. The steps 1 and 2 require $O(n)$ time both at each iteration. Since the orders of N are sorted in the increasing completion time, the orders of S obtained at the step 2 are also sorted in the increasing completion time. Hence the step 3 requires $O(1)$ time at each iteration. Since there are at most n iterations, the complexity is $O(n^2)$.

Then we prove that the algorithm GA5.1 provides an optimal solution. Suppose that there is an optimal delivery schedule θ^* respecting Lemma 5.1 for NSD problem. Let θ be the delivery schedule generated by algorithm GA5.1. Suppose that the k last delivery batches are the same in the two schedules and the $(k+1)$th last delivery batch B_{k+1} is different in the two schedules. According to Lemma 5.1 and the step 1 of algorithm GA5.1, $B_{k+1}(\theta^*)$ and $B_{k+1}(\theta)$ are delivered at the same time $T = \max_{j \in N'} C_j(\sigma)$ where N' is the set of delivered orders before the k last delivery batches. Let S be the set of delivered orders before the k last delivery batches with the deadline greater than or equal to T. We distinguish two cases:

- if $|S| < c$, it is clear that the orders of $B_{k+1}(\theta^*)$ are in $B_{k+1}(\theta)$. We can put all orders j, such that $j \in B_{k+1}(\theta)$ and $j \notin B_{k+1}(\theta^*)$, in $B_{k+1}(\theta^*)$ without increasing the number of delivery batches. Now $B_{k+1}(\theta^*)$ becomes the same as $B_{k+1}(\theta)$.

- if $|S| \geq c$, we have $|B_{k+1}(\theta)| = c$ and $B_{k+1}(\theta^*) \subset S$. If $|B_{k+1}(\theta^*)| < c$, we fill $B_{k+1}(\theta^*)$ with some orders of S which are not in $B_{k+1}(\theta^*)$ and update the delivery time of modified batches. Now we do not increase the number of batches and have $|B_{k+1}(\theta^*)| = c$. If there exists an order j such that $j \notin B_{k+1}(\theta)$ and $j \in B_{k+1}(\theta^*)$, then there exists another order i such that $i \in B_{k+1}(\theta)$, $i \notin B_{k+1}(\theta^*)$ and $C_j < C_i$ since order i is one of the last c completed order. We can interchange orders i and j in θ^* without changing the number of batches and update the delivery time of modified batches. We repeat this operation until $B_{k+1}(\theta^*)$ becomes the same as $B_{k+1}(\theta)$.

Hence, we can transform any optimal schedule θ^* to θ without increasing the transportation cost. □

SD problem In this case, a finished order can be split and delivered in several batches. We propose a polynomial-time algorithm (see algorithm GA5.2) for SD problem.

Algorithm GA5.2

Step 1: Let $V' \subseteq V(\sigma)$ denote the set of production triplets (see definition 5.1) corresponding to the undelivered parts of orders. Set current delivery time $T = \max_{V_j \in V'} b_j$.

Step 2: Find the set of production triples corresponding to the orders with a deadline greater than or equal to T from V'. Let $S \subseteq V'$ denote this set.

Step 3: If $\sum_{V_j \in S}(b_j - a_j)/p_{J_j} < c$, deliver the parts of orders corresponding to S in one batch which departs at time T. Otherwise, create one batch which departs at time T as follows: Iteratively, if the remaining capacity of the batch, denoted by c', is enough, add the part of order corresponding to the last completed production triplet $V_j \in S$ in the delivery batch, otherwise split V_j into two production triplets $V_j' = (J_j, a_j, b_j - c'p_{J_j})$ and $V_j'' = (J_j, b_j - c'p_{J_j}, b_j)$. Put the part of order J_j corresponding to V'' in the batch to form a full batch. Then update V'. If all orders are delivered, then STOP. Otherwise, go to step 1.

Theorem 5.2 Algorithm GA5.2 finds an optimal delivery schedule for SD problem in $O(n^2)$ time.

Proof. The proof is similar as for Theorem 5.1. □

Lemma 5.2 For any given feasible NSP production schedule, there exists an optimal delivery schedule in which the orders are not split.

Proof. For a given NSP schedule, algorithm GA5.2 finds an optimal delivery schedule which is a NSD schedule. In fact, in the case NSP, each production triplet V_j corresponds to a non split order J_j, i.e., $b_j - a_j = p_{J_j}$. In the step 3 of algorithm GA5.2, when we create a full batch in the case $\sum_{V_j \in S}(b_j - a_j)/p_{J_j} > c$, we do not split any production triplet, i.e. the orders are put in the delivery batch without splitting. □

As discussed in section 5.2, we do not consider the non-splittable production and splittable delivery (NSP-SD) problem, because according to Lemma 5.2, for any feasible NSP production schedule, there exists an optimal delivery schedule which is a NSD schedule.

5.4 Coordinated Scheduling Problems

Recall that the manufacturer's problem is a decision problem and the 3PL provider's problem is an optimization problem, hence clearly, an integrated model minimizing the transportation cost TC subject to the deadline constraints is the best way for the coordination. Concerning the implementation of coordination, since the manufacturer does not benefit from the coordination, the 3PL provider needs to recompense the manufacturer. In order to guarantee the benefit for both parties, the amount of compensation P has to respect the formulation

$$TC_2 + P \leq TC_1 \tag{5.4.7}$$

where TC_1 and TC_2 are the transportation cost in the individual schedule and the integrated schedule respectively.

In what follows, we concentrate on the resolution of the integrated scheduling problems. We first consider SP-NSD and SP-SD problems, then NSP-NSD problem.

5.4.1 SP-NSD Problem and SP-SD Problem

In this section, we first give some properties for SP-NSD and SP-SD problems. Then we provide a polynomial-time algorithm that solves these problems in two special cases. This algorithm will be used to compute lower bounds in the branch-and-bound algorithm that solves NSP-NSD problem.

Lemma 5.3 There exists an optimal integrated schedule for SP-NSD and SP-SD problems, if it exists, such that the following properties hold:

(1) Each order is processed in one production block only.

(2) Each production block starts at the minimum release date of orders within this block.

(3) Each batch is delivered at its completion time when all orders (or parts of orders) of the batch are completed.

Proof. (1) Suppose there exists an optimal integrated schedule (σ^*, θ^*) which does not respect property 1, such that order j is the first order which is split and scheduled in several production blocks. Let K_i be the first block containing order j (see figure 5.3(a)). We reschedule as early as possible the rest of order j in the idle times after K_i (see figure 5.3(b)). Consequently, the order j is processed only in K_i. The delivery schedule θ^* is also feasible for the new production schedule. So this new integrated schedule is also optimal. We can repeat this argument in a finite number of times until the property 1 is satisfied.

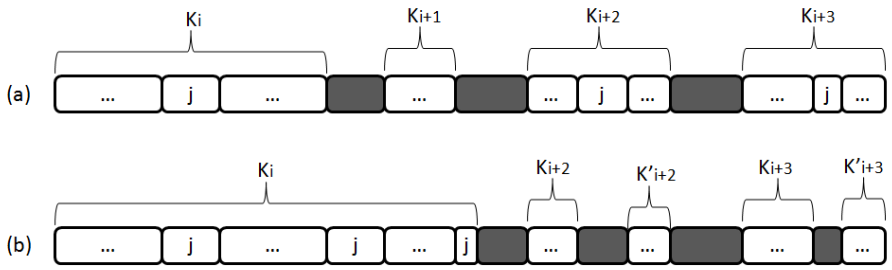


Figure 5.3: Illustration of property 1 of Lemma 5.3

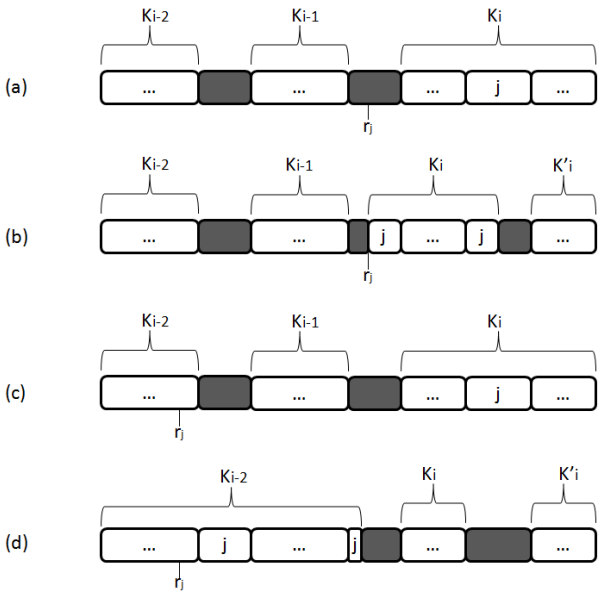


Figure 5.4: Illustration of property 2 of Lemma 5.3

(2) Suppose there exists an optimal integrated schedule (σ^*, θ^*) which respects property 1 but does not respect property 2, such that production block K_i is the first block which does not respect property 2. Suppose order j has the earliest release date among the orders of block K_i. We reschedule order j as early as possible without changing other orders. We distinguish two cases: in the first case, the completion time of the production block K_{i-1} is less than r_j (see figure 5.4(a)), in the new production schedule all blocks before K_i' respect property 2 (see figure 5.4(b)); in the second case, the completion time of the production block K_{i-1} is greater than or equal to r_j (see figure 5.4(c)), in the new production schedule all blocks before K_i respect property 2 (see figure 5.4(d)). In the new production schedules((b) and (d)), we reduce the total size of blocks which do not respect property 2. The delivery schedule θ^* is also feasible for these new production schedules. So this new integrated schedule is also optimal. We can repeat this argument in polynomial time until property 2 is satisfied.

(3) The proof is the same as Lemma 5.1. □

Lemma 5.4 There exists an optimal integrated schedule for SP-NSD and SP-SD problems, if it exists, such that the structure of production blocks, consisting of the orders composition, the starting time and the ending time of each block, is the same as that constructed by the preemptive EDD rule.

Proof. Suppose there exists an optimal integrated schedule (σ^*, θ^*) which respects the properties of Lemma 5.3, but does not respect the property of Lemma 5.4. Let $(K_1^*, \ldots, K_l^*)$ be the set of production blocks of σ^*. Let σ denote the production schedule constructed by the preemptive EDD rule. Let $(K_1, \ldots, K_u)$ be the set of production blocks of σ. Suppose K_i^* and K_i are the first block which are different in two schedules.

According to the preemptive EDD rule, in σ there is a idle time only if there is no available order. Hence there is no idle time among the split parts of each order. In addition, at each end of idle time, the rule schedules always one of remaining orders with the earliest release date. Consequently, σ respects the properties of Lemma 5.3.

According to property 2 of Lemma 5.3, K_i^* and K_i must start at the same time. Noting that in σ there is a idle time only if there is no available order, we know that all orders of K_i^* must be in K_i, i.e. $K_i^* \subseteq K_i$.

Suppose order j is the first order such that $j \notin K_i^*$ and $j \in K_i$. Since the orders before j of K_i are also in K_i^*, we know that K_i^* can finish only at or after r_j. According to property 2 of Lemma 5.3, the block including the order j must start before or at r_j. Consequently, the order j must be in K_i^*, which is in conflict with the assumption of order j. That means that all orders of K_i must be in K_i^*, i.e. $K_i \subseteq K_i^*$.

Hence, we have $K_i = K_i^*$ and the ending times of K_i and K_i^* are the same. So K_i^* and K_i are not the first block which are different in two schedules. Hence the property of Lemma 5.4 is satisfied. □

Then, we introduce the *Shortest Remaining Processing Time (SRPT)* rule to construct a production schedule in SP-NSD and SP-SD problems.
SRPT rule: at each decision point t in time, consisting of each release date and each order completion time, schedule an available order j (i.e. $r_j \leq t$) with the shortest remaining processing time. If no orders are available at a decision point, schedule an idle time until the next release date.

Next, we provide a polynomial-time algorithm (see algorithm GA5.3) for SP-NSD and SP-SD problems in the following two special cases:

case 1: The vehicle capacity is unlimited, i.e. $c = \infty$.

case 2: The set of orders N can be divided into two subset of orders N_1 and N_2. $\forall j \in N_1$, $\nexists j' \in N_1$ such that $r_j \leq r_{j'} < r_j + p_j$. $\forall j \in N_1$ and $i \in N_2$, $r_j + p_j \leq r_i$. In any production block of the schedule constructed by preemptive EDD rule, the orders of N_2 have the same release date.

Algorithm GA5.3

Step 1: Generate a production schedule σ with the preemptive EDD rule. If $C_j(\sigma) \leq \overline{d}_j, \forall j \in N$, go to Step 2, otherwise there is no solution and STOP.

Step 2: Let $N' \subseteq N$ denote the set of undelivered orders. Set the current delivery time $T = \max_{j \in N'} C_j(\sigma)$.

Step 3: Find the set of undelivered orders with deadline greater than or equal to T. Let S denote this set.

Step 4: If $|S| < c$, deliver the orders of S in one batch which departs at time T. Otherwise, reschedule the orders of S in σ with the SRPT rule and do not change the schedule of other orders, then deliver the last c completed orders of S in one batch which departs at time T. Then update N'. If all orders are delivered, then STOP. Otherwise, go to step 2.

Theorem 5.3 Algorithm GA5.3 finds an optimal integrated schedule for SP-NSD and SP-SD problems in the special case 1 in $O(n^2)$ time, and the special case 2 in $O(n^2 \log n)$ time.

Proof. We first prove the complexity of algorithm GA5.3. At step 1, the generation of σ takes $O(n \log n)$ time. We take $O(n)$ time to check feasibility of the solution. At each iteration, step 2 and step 3 take $O(n)$ time respectively. The step 4 takes $O(1)$ time for the case $|S| \leq c$ and takes $O(n \log n)$ time to reschedule the orders of S with the SRPT rule for the case $|S| > c$. There are $O(n)$ iterations. We note that for the problem in the special case 1, at the step 4 we have always $|S| \leq c$, the algorithm GA5.3 finds an optimal integrated schedule for SP-NSD and SP-SD problems in the special case 1 in $O(n^2)$ time, and the special case 2 in $O(n^2 \log n)$ time.

Next, we prove that the algorithm provides an optimal solution. We use a recursion theorem to prove it. Let (σ, θ) denote the integrated schedule provided by algorithm GA5.3. Let B_i denote the i^{th} last batch of θ. Let $|B_i|$ denote the size of B_i. Let $T(B_i)$ denote the departure date of B_i. According to Lemma 5.4, we find a structure of production blocks of an optimal integrated schedule with the preemptive EDD rule. We can fix the value for minimum $T(B_1)$ which equals to the makespan. Suppose that $T(B_i)$ is minimum. We prove the problems in two special cases 1 and 2 separately.

Case 1: Since $c = \infty$, the algorithm generates a non full batch $|B_i| < c$. Since B_i delivers all undelivered available orders at time $T(B_i)$, $T(B_{i+1})$ is a production completion time of one order of which the deadline is less than $T(B_i)$. According to the preemptive EDD rule, we cannot anticipate the maximum production completion time of all orders of which the deadlines are less than $T(B_i)$. Hence $T(B_{i+1})$ is minimum,

Case 2: In this case, if the algorithm generates a non full batch $|B_i| < c$, with the same

argument of the case 1, we can prove that $T(B_{i+1})$ is minimum. If the algorithm generates a full batch $|B_i| = c$, we can also prove the minimization of $T(B_{i+1})$ as follows:

- if $T(B_{i+1})$ is a production completion time of one order of which the deadline is less than $T(B_i)$, according to the preemptive EDD rule, we cannot anticipate the maximum production completion time of all orders of which the deadlines are less than $T(B_i)$.
- if $T(B_{i+1})$ is a production completion time of one order of N_1 of which the deadline is greater than or equal to $T(B_i)$, according to the preemptive EDD rule the completion times of orders of N_1 cannot be anticipated and the SRPT rule does not change the completion times of N_1, hence we cannot anticipate $T(B_{i+1})$. If $T(B_{i+1})$ is a production completion time of one order of N_2 of which the deadline is greater than or equal to $T(B_i)$, according to the SRPT rule and the characteristic of the special case 2, this order is executed before all orders of B_i in σ and $B_i \subseteq S \cap N_2$. Since the orders of B_i have the longest processing time among the orders of $S \cap N_2$, we cannot anticipate $T(B_{i+1})$.

For the problems in the two special cases, we prove that $T(B_{i+1})$ is minimum. Hence the algorithm GA5.3 generates the minimum number of batches to deliver all orders. □

Remark that the computational complexities of SP-NSD and SP-SD problems in general case are still open.

5.4.2 NSP-NSD Problem

It can be observed easily that problem $1|r_j, \overline{d}_j|-$ reduces to NSP-NSD problem, i.e. it is a special case of NSP-NSD problem with $c = 1$. Consequently, NSP-NSD problem is NP-hard in the strong sense. In this section, we first present two heuristics to determine upper bounds of TC. Then we describe a branch-and-bound algorithm to solve NSP-NSD problem. Finally, we provide two MILP (mixed integer linear programming) models which are used to evaluate the performance of the branch-and-bound algorithm.

(i) Heuristics

In our branch-and-bound algorithm, we will use two heuristics that try to construct a feasible integrated schedule for NSP-NSD problem.

The first heuristic, denoted by H5.1, uses the non-preemptive EDD rule, which forces to create a production schedule without preemption. If the obtained production schedule is feasible, then we apply algorithm GA5.1.

Heuristic H5.1

Step 1: Create a production schedule σ with the non-preemptive EDD rule. If $C_j(\sigma) \leq \overline{d}_j, \forall j \in N$, go to step 2. Otherwise, the algorithm cannot provide a feasible solution and STOP.

Step 2: Apply algorithm GA5.1 to compute a delivery schedule.

The second heuristic, denoted by H5.2, uses a SP-NSD integrated schedule computed by algorithm GA5.3 to construct, if possible, a feasible integrated schedule for NSP-NSD problem.

Heuristic H5.2

Step 1: Create a priority list of orders, such that in the given schedule (σ, θ), if $D_i(\theta) < D_j(\theta)$, order i must be before order j in the list, and if $D_i(\theta) = D_j(\theta)$ and $C_i(\sigma) < C_j(\sigma)$, order i must be before order j in the list.

Step 2: Schedule each order as early as possible without preemption. When there are several orders which can be scheduled, we choose the order with the highest priority. Let σ' be the constructed production schedule. If $C_j(\sigma') \leq \overline{d}_j, \forall j \in N$, go to step 3. Otherwise, the algorithm cannot provide a feasible solution and STOP.

Step 3: Apply algorithm GA5.1 to compute a delivery schedule.

(ii) Branch-and-bound algorithm

We propose a branch-and-bound algorithm (see algorithm B5.1) for NSP-NSD problem based on the branch-and-bound algorithm of Carlier recalled in section 5.3.1.

Algorithm 4: Algorithm B5.1

```
1  Generate the root associated with LB(L_max, root) and UB(L_max, root) as the
   algorithm of Carlier, and put this node in list L;
2  while L ≠ ∅ do
3      Choose one node u in L with minimum LB(L_max, u) ;
4      if UB(L_max, u) > 0 and LB(L_max, u) ≤ 0 then
5          Compute LB(TC, u) and UB(TC, u) as algorithm B5.2;
6          if LB(TC, u) < UB*(TC) then
7              if UB(TC, u) < n + 1 then
8                  Apply algorithm B5.2 with p_j, r_j(u), d̄_j(u), the original deadlines
                   d̄_j(root) for j ∈ N, and the precedence relations between orders
                   imposed at the path from the root to node u;
9              else
10                 Branch as Carlier's algorithm and add new nodes with the bounds
                   of L_max in L;
11     else
12         if LB(L_max, u) ≤ UB(L_max, u) ≤ 0 then
13             Apply algorithm B5.2 with p_j, r_j(u), d̄_j(u), the original deadlines
               d̄_j(root) for j ∈ N, and the precedence relations between orders
               imposed at the path from the root to node u;
14     Remove u from L.
```

In the search tree, a node u is characterized by: release dates $r_j(u)$ and deadlines $\overline{d}_j(u)$ of orders $j \in N$, a lower bound of $L_{\max}$ denoted by $LB(L_{\max}, u)$, an upper bound of $L_{\max}$ denoted by $UB(L_{\max}, u)$, a lower bound of TC denoted by $LB(TC, u)$, an upper bound of TC denoted by $UB(TC, u)$, the current best upper bound of TC denoted by $UB^*(TC)$, and precedent constraints between the orders. If the node u is the root

of search tree, $r_j(root)$ and $\overline{d}_j(root)$ represent the original release dates and deadlines, respectively.

Algorithm B5.1 first applies Carlier's algorithm. When a feasible solution, i.e. $UB(TC, u) < n + 1$ (line 7 of algorithm B5.1) or $UB(L_{\max}, u) \leq 0$ (line 12 of algorithm B5.1), is found at node u, we apply another branch-and-bound algorithm denoted by algorithm B5.2 from node u to try to find a local optimal solution minimizing TC. When algorithm B5.2 stops, algorithm B5.1 continues the branching of Carlier's algorithm for the remaining active nodes (line 10 of algorithm B5.1).

In algorithm B5.2, the lower bound $LB(TC, u)$ is computed by solving two relaxed problems which respect the two special cases of SP-NSD problem. The upper bound $UB(TC, u)$ is obtained by applying the two heuristics H5.1 and H5.2. Branching of algorithm B5.2 is done by assigning to each position of production schedule an order respecting a set of rules. Moreover, when algorithm B5.2 applies algorithm GA5.3, heuristics H5.1 and H5.2, the part of production adopts the modified deadlines $\overline{d}_j(u)$ which are necessary to determine a feasible production schedule according to Carlier's algorithm, and the part of delivery adopts the original deadlines $\overline{d}_j(root)$.

Algorithm B5.2

Lower bound: At node u, we solve two relaxed problems which respect the two special cases of SP-NSD problem:

Problem 1: Set $c = n$.

Problem 2: Divide the set of orders N in two subsets of orders N_1 and N_2. $\forall j \in N_1$, $\nexists j' \in N_1$ such that $r_j(u) \leq r_{j'}(u) < r_j(u) + p_j$. $\forall j \in N_1$ and $i \in N_2$, $r_j(u) + p_j \leq r_i(u)$. Schedule the orders with preemptive EDD rule, then modify the release dates of the orders of N_2 utile that in each production block the orders of N_2 have the same release date.

We solve these relaxed problems by applying algorithm GA5.3: execute step 1 of algorithm GA5.3 with $\overline{d}_j(u)$ for $j \in N$, and execute the remaining steps of the algorithm with the original deadlines, i.e. $\overline{d}_j(root)$ for $j \in N$. Let (σ_1, θ_1) and (σ_2, θ_2) denote the obtained SP-NSD integrated schedules for the above problems

respectively. Set $LB(TC, u) = \max\{TC(\sigma_1, \theta_1), TC(\sigma_2, \theta_2)\}$.

Upper bound: Firstly, generate a NSP-NSD integrated schedule by applying heuristic H5.2 with the above obtained schedule (σ_2, θ_2) and the original deadlines, i.e. $\overline{d}_j(root)$ for $j \in N$. Secondly, generate a second NSP-NSD integrated schedule by applying heuristic H5.1: execute the step 1 of heuristic H5.1 with $\overline{d}_j(u)$ for $j \in N$, and execute the step 2 of heuristic with the original deadlines, i.e. $\overline{d}_j(root)$ for $j \in N$. Finally, if one or both constructed integrated schedules are feasible, set $UB(TC, u)$ as the smallest TC among the two schedules. Otherwise, set $UB(TC, u) = n + 1$. Update $UB^*(TC)$ if necessary.

Branching: if $LB(TC, u) < UB^*(TC, u)$ for a node u, firstly choose one order to be scheduled in the current production position. Order j is a valid candidate if it respects the following rules. Let N' denote the set of unscheduled orders without order j.

active scheduling rule: $r_j(u) < \min_{k \in N'}(r_k(u) + p_k)$

deadline rule: $r_j(u) + p_j \leq \min_{k \in N'}(\overline{d}_k(u) - p_k)$

precedence relations rule: $\sum_{k \in N'} x_{kj} = 0$, where $x_{kj} = 1$ if order k precedes order j, otherwise $x_{kj} = 0$.

Then, require the valid candidate j to be scheduled at the current production position and let u' be the corresponding new node. Set $r_k(u') = \max(r_k(u), r_j(u) + p_j), \forall k \in N'$.

Table 5.3: Example for branch-and-bound algorithm B5.1

Order j	1	2	3	4	5	6	7
p_j	13	18	19	20	7	8	2
r_j	35	38	14	21	1	48	14
$\overline{d}_j$	69	79	99	80	65	88	51

Example To illustrate algorithm B5.1, we consider the following example with six orders where the vehicle capacity c is equal to 2 and the transportation cost h per batch is equal to 1. Table 5.3 gives the orders' parameters.

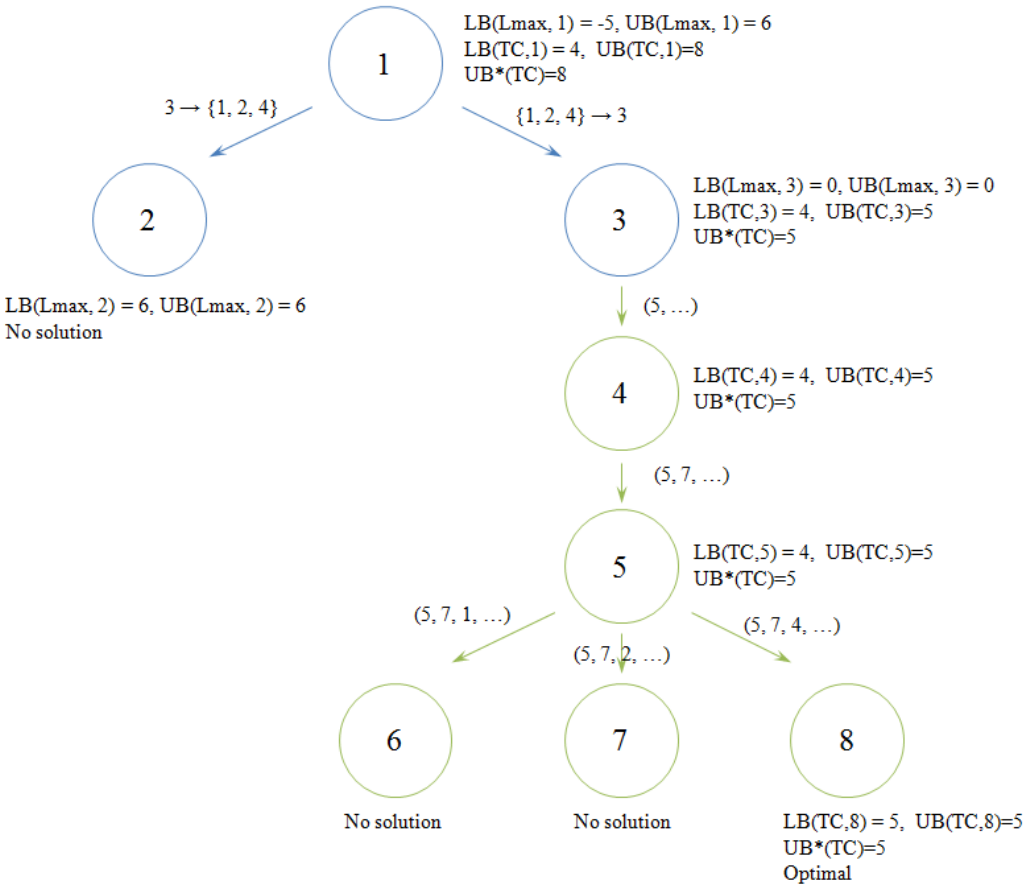


Figure 5.5: Illustration of branch-and-bound algorithm B5.1

Figure 5.5 illustrates the search tree of the branch-and-bound algorithm B5.1. At the root, i.e. node 1, since $UB(L_{\max}, u) > 0$ and $LB(L_{\max}, u) \leq 0$, we check $UB(TC, u)$ and $LB(TC, u)$. Since $UB(TC, 1) = 8$, i.e. the algorithm does not find a feasible NSP schedule, the tree branches as Carlier's algorithm. Here, we have the critical order $e = 3$ and the critical set $J = \{1, 2, 4\}$.

In node 2, Carlier's algorithm requires the critical order to be processed before the orders of critical set by setting the deadline of critical order 3, i.e. $\overline{d}_3(2) = 29$. Since $LB(L_{\max}, 2) = 6$ and $UB(L_{\max}, 2) = 6$, the algorithm ensures that there is no feasible NSP-NSD schedule for node 2.

In node 3, Carlier's algorithm requires the critical order to be processed after the orders of the critical set by setting the release date of the critical order 3, i.e. $r_3(3) = 72$. Since $LB(L_{\max}, 3) = 0$ and $UB(L_{\max}, 3) = 0$, the algorithm ensures that there is at least one feasible NSP-NSD schedule. Then it applies algorithm B5.2. The precedence

relations include that the order 3 has to be processed after the orders 1, 2 and 4. Since initially $LB(TC,3) = 4$ and $UB(TC,3) = 5$, the tree branches as algorithm B5.2. $UB^*(TC)$ is updated to 5.

For the first position of production schedule, algorithm B5.2 finds that order 5 is the only order that respects the rules of candidate. By scheduling order 5 in the first position, node 4 is generated. Since $r_k(3) < r_5(3) + p_5(3), \forall k \in N\backslash\{5\}$, the algorithm does not change the release dates, i.e. $r_k(4) = r_k(4)$, $\forall k \in N$. Since $LB(TC,4) = 4$ and $UB(TC,4) = 5$, the tree continues to branch. We still have $UB^*(TC) = 5$.

The algorithm finds the only candidate 7 for the second position of production schedule. By scheduling order 7 in the second position, node 5 is generated. Since $r_k(4) < r_7(4) + p_7, \forall k \in N\backslash\{5,7\}$, the algorithm does not change the release dates, i.e. $r_k(5) = r_k(5)$, $\forall k \in N$. Since $LB(TC,5) = 4$ and $UB(TC,5) = 5$, the tree continues to branch. We still have $UB^*(TC) = 5$.

For the third position of production schedule, algorithm B5.2 finds a set of candidates $\{1,2,4\}$.

By scheduling order 1 in the third position, node 6 is generated. The algorithm sets $r_2(6) = \max\{r_2(5), r_1(5) + p_1\} = 48$ and $r_4(6) = \max\{r_4(5), r_1(5) + p_1\} = 48$. With this modified setting, there is no feasible solution for SP-NSD problem in the two special cases. Hence there is no feasible solution for NSP-NSD problem.

By scheduling order 2 in the third position, node 7 is generated. The algorithm sets $r_1(7) = \max\{r_1(5), r_2(5) + p_2\} = 56$ and $r_4(7) = \max\{r_4(6), r_2(6) + p_2\} = 56$. With this modified setting, there is no feasible solution for SP-NSD problem in the two special cases. Hence there is no feasible solution for NSP-NSD problem.

By scheduling order 4 in the third position, node 8 is generated. The algorithm sets $r_1(8) = \max\{r_1(5), r_4(5) + p_4\} = 41$ and $r_2(8) = \max\{r_2(5), r_4(5) + p_4\} = 41$. With this modified setting, algorithm B5.1 computes $LB(TC,8) = 5$ and $UB(TC,8) = 5$, a local optimal solution is found. Since there is no active node, the algorithm stops and an global optimal solution for NSP-NSD problem is found (see figure 5.6).

Figure 5.6 shows an optimal solution for NSP-NSD problem. The production sequence is $(5,7,4,1,2,6,3)$. There are five delivery batches: $\{7\}, \{5,1\}, \{2\}, \{4,6\}$, and $\{3\}$, which depart respectively at time 16, 54, 72, 80, and 99.

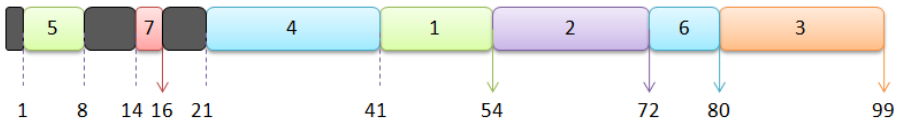


Figure 5.6: An optimal solution for NSP-NSD problem

(iii) Two mixed integer linear programming models

We first present two properties of optimal integrated schedule for NSP-NSD problem.

Lemma 5.5 There exists an optimal integrated schedule for NSP-NSD problem, if it exists, such that each batch is delivered

(1) either at one production completion time of order,

(2) or at one delivery deadline of order.

Proof. Suppose that there is an optimal integrated schedule for NSP-NSD problem, such that it does not respect the property. We can change the delivery time of each batch to respect property (1) or (2) without changing the number of delivery batches. □

Then, we propose the first MILP model (see MILP5.1) which extends the well-known positional scheduling model as follows. In this model, according to property (1) of Lemma 5.5, we suppose that each batch departs at one production completion time of order. Let $M_1 = \min(\max_{i\in N} r_i + \sum_{i\in N} p_i, \max_{i\in N} \overline{d}_i)$.

Decision variables:

- $x_{iq} = \begin{cases} 1, & \text{if order } i \text{ is processed in position } q, i, q \in \{1,\ldots,n\} \\ 0, & \text{otherwise} \end{cases}$
- $C_{[q]}$ = production completion time of the order processed in position q, $q \in \{1,\ldots,n\}$
- $y_{iq} = \begin{cases} 1, & \text{if order } i \text{ is delivered at time } C_{[q]}, i, q \in \{1,\ldots,n\} \\ 0, & \text{otherwise} \end{cases}$
- w_q = number of batches departing at time $C_{[q]}$, $q \in \{1,\ldots,n\}$

MILP5.1:

$$\min \quad h\sum_{q=1}^{n} w_q \tag{5.4.8}$$

$$\text{s.t.} \quad \sum_{i=1}^{n} x_{iq} = 1 \qquad q \in \{1,\dots,n\} \tag{5.4.9}$$

$$\sum_{q=1}^{n} x_{iq} = 1 \qquad i \in \{1,\dots,n\} \tag{5.4.10}$$

$$\sum_{i=1}^{n} y_{iq} \leq c w_q \qquad q \in \{1,\dots,n\} \tag{5.4.11}$$

$$\sum_{q=1}^{n} y_{iq} = 1 \qquad i \in \{1,\dots,n\} \tag{5.4.12}$$

$$\sum_{q'=0}^{q} y_{iq'} \leq \sum_{q'=0}^{q} x_{iq'} \qquad i,q \in \{1,\dots,n\} \tag{5.4.13}$$

$$C_{[q]} \leq \overline{d}_i y_{iq} + M_1(1 - y_{iq}) \qquad i,q \in \{1,\dots,n\} \tag{5.4.14}$$

$$C_{[q]} \geq \sum_{i=1}^{n} (r_i + p_i) x_{iq} \qquad q \in \{1,\dots,n\} \tag{5.4.15}$$

$$C_{[q]} \geq C_{[q-1]} + \sum_{i=1}^{n} p_i x_{iq} \qquad q \in \{2,\dots,n\} \tag{5.4.16}$$

$$x_{iq} \in \{0,1\} \qquad i,q \in \{1,\dots,n\} \tag{5.4.17}$$

$$y_{iq} \in \{0,1\} \qquad i,q \in \{1,\dots,n\} \tag{5.4.18}$$

$$w_q \in \mathbb{N} \qquad q \in \{1,\dots,n\} \tag{5.4.19}$$

In MILP5.1, the objective function is to minimize the transportation cost. Constraints (5.4.9) and (5.4.10) ensure that one order is processed in only one position and one position is affected to only one order. Constraints (5.4.11) are the constraints of the batch capacity. Constraints (5.4.12) guarantee that one order is delivered at only one possible departure date. Constraints (5.4.13) ensure that the order can be delivered only if it is completed. Constraints (5.4.14) are the constraints of deadlines and $M_1(1 - y_{iq})$ guarantees that the constraints are always true if order i is not delivered at time $C_{[q]}$. Constraints (5.4.15) impose that if order i is processed in position q, the completion time of order i should be after or at time $r_i + p_i$. Constraints (5.4.16) impose that if order i is processed in position q, the completion time of order i should be after or at time $C_{[q-1]} + p_i$. Constraints (5.4.17)-(5.4.19) give the domain of definition of each variable.

Finally, we propose a second MILP model (see MILP5.2) which extends the well-known disjunctive scheduling model as follows. In this model, according to property (2) of Lemma 5.5, we can suppose that each batch departs at one order deadline. Let $s_1, \ldots, s_u$ denote the possible departure dates. Let $M_2 = \max_{i \in N} \overline{d}_i$.

Decision variables:

- $x_{ij} = \begin{cases} 1, & \text{if order } i \text{ is processed before order } j, i, j \in \{1, \ldots, n\} \\ 0, & \text{otherwise} \end{cases}$
- $t_j =$ production starting time of order j , $j \in \{1, \ldots, n\}$
- $y_{iq} = \begin{cases} 1, & \text{if order } i \text{ is delivered at time } s_q, i \in \{1, \ldots, n\}, q \in \{1, \ldots, u\} \\ 0, & \text{otherwise} \end{cases}$
- $w_q =$ number of batches departing at time s_q, $q \in \{1, \ldots, u\}$

MILP5.2:

$$\min h \sum_{q=1}^{u} w_q \tag{5.4.20}$$

$$\text{s.t.} \quad x_{ij} + x_{ji} = 1, \quad i, j \in \{1, \ldots, n\}, i \neq j \tag{5.4.21}$$

$$t_j - t_i \geq p_i - (1 - x_{ij})M_2, \quad i, j \in \{1, \ldots, n\} \tag{5.4.22}$$

$$t_j \geq r_j, \quad j \in \{1, \ldots, n\} \tag{5.4.23}$$

$$t_j + p_j \leq \sum_{q=1}^{u} (y_{jq} s_q), \quad j \in \{1, \ldots, n\} \tag{5.4.24}$$

$$\sum_{i=1}^{n} y_{iq} \leq c w_q, \quad q \in \{1, \ldots, u\} \tag{5.4.25}$$

$$\sum_{q=1}^{u} y_{iq} = 1, \quad i \in \{1, \ldots, n\} \tag{5.4.26}$$

$$y_{iq} = 0, \quad i \in \{1, \ldots, n\}, q \in \{1, \ldots, u\}, \overline{d}_i < s_q \tag{5.4.27}$$

$$x_{ij} \in \{0, 1\}, \quad i, j \in \{1, \ldots, n\} \tag{5.4.28}$$

$$y_{iq} \in \{0, 1\}, \quad i \in \{1, \ldots, n\}, q \in \{1, \ldots, u\} \tag{5.4.29}$$

$$w_q \in \mathbb{N}, \quad q \in \{1, \ldots, u\} \tag{5.4.30}$$

In MILP5.2, the objective function is to minimize the transportation cost. Constraints (5.4.21) show that either order i is processed before order j or order j before order i for any two different orders i and j. Constraints (5.4.22) ensure that order j starts its processing after or at the completion of order i if order i is processed before order j. Constraints (5.4.23) guarantee that each order starts its processing after or at its release date. Constraints (5.4.24) ensure that each order is delivered after or at its production completion time. Constraints (5.4.25) are the constraints of the batch capacity. Constraints (5.4.26) ensure that each order is delivered in one batch only. Constraints (5.4.27) are the delivery deadlines constraints. Constraints (5.4.28)-(5.4.30) give the domain of definition of each variable.

5.5 Computational Results

In this section, we evaluate the performance of branch-and-bound algorithm B5.1 by comparing it with MILP5.1 and MILP5.2. The branch-and-bound algorithm is implemented in C++ and the MILP models are implemented in Cplex V12.5.1. The experiments are carried out on a DELL 2.50GHz personal computer with 8GB RAM.

We reuse the method of Briand et al. (2010) to generate instances. We consider $n \in \{10, 20, 30, 50, 70, 100, 150, 200, 300, 500\}$. The integers p_j, r_j and $\overline{d}_j$ are generated respectively from the uniform distributions [1,50], $[0, \alpha \sum_{j=1}^{n} p_j]$ and $[(1-\beta)a \sum_{j=1}^{n} p_j, a \sum_{j=1}^{n} p_j]$, where $\alpha, \beta \in \{0.2, 0.4, 0.6, 0.8, 1\}$ and $a \in \{100\%, 110\%\}$. If $\overline{d}_j < r_j + p_j$, $\overline{d}_j$ has been updated by $r_j + p_j$. The transportation cost of one batch h is equal to 1. We choose a set of hard instances as follows: we apply the branch-and-bound algorithm of Carlier to find the minimum $L_{\max}$ for each instance, if the problem for this instance cannot be solved at the root of the search tree, we consider this instance as a hard instance. If the found $L_{\max}$ of this hard instance is positive, we add this value to each $\overline{d}_j$ of this instance to ensure that we have at least one feasible solution. For $n \leq 70$, we consider the batch capacity $c \in \{2, 3, \lceil \frac{n}{8} \rceil, \lceil \frac{n}{4} \rceil\}$, and $c \in \{\lceil \frac{n}{50} \rceil, \lceil \frac{n}{30} \rceil, \lceil \frac{n}{20} \rceil, \lceil \frac{n}{10} \rceil\}$ for $n \geq 100$. 80 hard instances for each value of n are generated. Totally 800 hard instances are generated.

Tables 5.4 - 5.7 illustrate the performance of branch-and-bound algorithm B5.1. Im-

Table 5.4: Performance of branch-and-bound algorithm B5.1.

n	Fea	Opt	Node	Time
10	100%	100%	2	0.07
20	100%	100%	16	0.85
30	100%	96.25%	165	14.82
50	100%	95%	173	19.16
70	100%	91.25%	183	36.13
100	100%	77.5%	324	78.46
150	100%	66.25%	334	118.18
200	100%	51.25%	298	150.02
300	100%	32.5%	240	209.01
500	100%	32.5%	118	212.98

Table 5.5: Performance of two MILP models.

	MILP5.1				MILP5.2			
n	Fea	Opt	Node	Time	Fea	Opt	Node	Time
10	100%	100%	155	0.42	100%	100%	1140	0.35
20	100%	100%	27791	26.59	98.75%	76.25%	262621	82.88
30	97.5%	62.5%	63950	159.56	63.75%	33.75%	349789	210.06
50	18.75%	10%	9603	290.07	20%	6.25%	139462	287.76

posing 5 minutes as the limit of execution time, we use the following measures to compare the branch-and-bound algorithm with the two MILP models, i.e., MILP5.1 and MILP5.2.

Fea: the percentage of instances for which a feasible solution is determined within the given time.

Opt: the percentage of instances which are solved to optimality within the given time.

Node: the average number of explored nodes.

Time: the average CPU time in seconds.

Gap1: the relative gap measured by $(UB^*(TC) - LB^*(TC))/LB^*(TC)$, where $UB^*(TC)$ and $LB^*(TC)$ are the best upper bound and lower bound. We consider the instances for which we obtained at least one feasible solution (optimal

Table 5.6: Gaps of solutions of branch-and-bound algorithm B5.1.

	Gap1	Gap2		
n	Average	Min	Max	Average
10	0%	0%	0%	0%
20	0%	0%	0%	0%
30	0.4%	6.67%	12.5%	10.56%
50	0.7%	5.88%	16.67%	13.97%
70	0.76%	6.25%	12.5%	8.7%
100	2.5%	4%	28.57%	11.1%
150	3.92%	2%	31.58%	11.62%
200	5.64%	2%	30%	11.57%
300	7.98%	2%	30.23%	11.83%
500	8.8%	2.22%	32%	13.03%

Table 5.7: Gaps of solutions of two MILP models.

	MILP5.1				MILP5.2			
	Gap1	Gap2			Gap1	Gap2		
n	Average	Min	Max	Average	Average	Min	Max	Average
10	0%	0%	0%	0%	0%	0%	0%	0%
20	0%	0%	0%	0%	4.24%	8.33%	34.29%	18.61%
30	9.67%	6.25%	54.15%	26.93%	10.55%	6.25%	81.25%	22.42%
50	12.22%	3.85%	64.74%	26.19%	18.81%	7.41%	78.63%	27.36%

solution included).

Gap2: the relative gap for the instances for which we obtained at least one feasible solution (optimal solution excluded).

The results show that the branch-and-bound algorithm B5.1 outperforms the MILP models. From Table 5.4 and Table 5.5, we observe that the average execution time and the number of nodes with the MILP models are always larger than the branch-and-bound algorithm, and the MILP models cannot find a feasible solution with $n \geq 70$ and 5 minutes as time limit. The branch-and-bound algorithm solves all instances with $n \leq 20$ optimally within a very short execution time less than one second, and more than 90% of instances with $n \leq 70$ within an average execution time less than 40 seconds.

The branch-and-bound algorithm finds at least a feasible solution and solve 32.5% of instances optimally with n up to 500 and 5 minutes as time limit.

Consulting the gaps in Table 5.6 and Table 5.7, we observe that the branch-and-bound algorithm has a much better performance. In average, Gap1 and Gap2 of branch-and-bound algorithm are less than 0.8% and 14% when $n \leq 70$. However, the maximum Gap2 shows some hard cases for branch-and-bound algorithm when $n \geq 100$. For two MILP models,in average, Gap1 and Gap2 exceed 10% and 20% respectively when $n \leq 50$.

5.6 Conclusions

In this chapter, we studied a production and outbound distribution scheduling problem in a supply chain with a manufacturer, a 3PL provider and a customer. We considered a single machine production and a direct batch delivery. Moreover, we considered an important feature in production and distribution: splittable or non-splittable. The manufacturer's objective is to determine a feasible production schedule in which the orders are completed before or at their deadline. The 3PL provider aims at deciding a delivery schedule minimizing the transportation cost.

We first investigated the individual scheduling problems. We reviewed the production scheduling problems (i.e. SP and NSP problems) and provided two polynomial-time algorithms to solve the distribution scheduling problems (i.e. SD and NSD problems). Then we investigated the coordinated scheduling problems (i.e. SP-NSD, SP-SD and NSP-NSD problems). We provided a polynomial algorithm to solve two special cases of SP-NSD and SP-SD problems. We also provided a branch-and-bound algorithm for NSP-NSD problem and evaluated its performance using numerical experiments. The results showed that the proposed algorithm has a better performance than the MILP models and can solve more than 90% of instances with $n \leq 70$ optimally within an average execution time less than 40 seconds.

Several important research issues remain open for future investigations. A first research direction is to study the complexities of SP-NSD and SP-SD problems. Another issue is to provide a better lower bound for the branch-and-bound algorithm. Finally, one might consider extending the model to the production system with parallel machines.

Chapter 6

Production and Outbound Distribution Scheduling Problems with Setup Times and Delivery Time Windows

6.1 Introduction

In this chapter, we investigate a production and outbound distribution scheduling problem proposed by an enterprise working in the packaging industry. The manufacturer has to process a set of orders on unrelated parallel machines with the consideration of sequence-dependent setup times and costs. Then the 3PL provider delivers orders in batches to the customers with heterogeneous vehicles subject to delivery time windows. The manufacturer's objective is to minimize the total setup cost and the 3PL provider's objective is to minimize the transportation cost. In order to minimize the total setup cost, the manufacturer prefers a special production sequence which may entail higher cost for the 3PL provider. For example it may happen that in a production sequence the completion times of orders are close to their delivery time windows, and consequently there are few flexibilities to group the orders in batches which increases the transportation cost. The coordination is motivated to decide an integrated schedule minimizing the total cost of supply chain while guaranteeing the delivery time windows.

Our objective in this chapter is to model this coordinated scheduling problem and propose a first decomposition approach to solve it using a commercial solver, to evaluate the feasibility of the approach and the potential benefit of coordination, and to propose some directions of improvement for future research.

We first investigate the individual scheduling problems. The manufacturer's problem is a production scheduling problem with setup costs and deadlines. The production scheduling problem with setup times (costs) has attracted an extensive research. Allahverdi et al. (2008) provided a survey of production scheduling problems with setup times or costs. We can find the related literature in section 2.1 of chapter 2. The 3PL provider's problem is a heterogeneous vehicle routing problem with time windows and release dates of orders (HVRPTWRD), which is an extension of the vehicle routing problem with time windows (VRPTW). The literature of VRPTW can be found in section 2.2 of chapter 2.

We also investigate the coordinated scheduling problem. An integrated model is adopted and a mechanism of coordination is developed. The corresponding scheduling problem is an integrated production and outbound distribution scheduling (IPODS) problem with setup times, routing delivery and time windows. The related literature can be found in section 2.3 of chapter 2. Ullrich (2013) investigated an IPODS problem where a set of orders of general size is processed on identical parallel machines subject to the machine release times, and delivered to customers within the time windows by a fleet of heterogeneous vehicles on minimizing the sum of tardiness. They provided a genetic algorithm for the integrated problem and evaluated its performance by comparing with two classical decomposition approaches. Different from his problem, our problem considers sequence-dependent setup cost in production, a new type of transportation cost in distribution and a different objective function.

Chapter 6 is organized as follows. In section 6.2, we formally describe the problems and introduce notations and terminology. Section 6.3 is devoted to the individual scheduling problems, and section 6.4 to the coordinated scheduling problems. In section 6.5, we evaluate the feasibility of the approach and the potential benefit of coordination using numerical experiments with small instances. Section 6.6 contains some conclusions and propositions for future research.

6.2 Problems and Notations

A manufacturer has to process a set of orders $N = \{1, \ldots, n\}$ from customers on a set of unrelated parallel machines $M = \{1, \ldots, m\}$. Order $j \in N$ requires the processing of q_j identical items on machines. Each order can be processed on any machine. We consider splitting property in production, in that each order can be split into parts and processed independently on several machines at the same time. There is no preemption of orders on each machine. Let p_j^e denote the processing time of unit item of order $j \in N$ on machine $e \in M$. Let C_j denote the completion time of order $j \in N$. Let ϱ_j denote the delivery destination of order $j \in N$. Moreover machine $e \in M$ has a release time γ^e.

On one machine, a sequence-dependent setup time and a setup cost occur when production changes from one order to another order. Let s_{0j} denote the setup time of order $j \in N$ which is processed as the first order on one machine. Let $s_{j_1 j_2}$ denote the setup time when production changes from order j_1 to order j_2 and $s_{j_1 j_2} = 0$ if $j_1 = j_2$, $j_1, j_2 \in N$. The setup times respect the triangle rule, i.e. $s_{j_1 j_2} + s_{j_2 j_3} \geq s_{j_1 j_2}$, $j_1, j_2, j_3 \in N$. The setup cost is proportional to the setup time. Let ρ be the cost for unit setup-time. Hence $\rho s_{j_1 j_2}$ is the setup cost when production changes from order $j_1 \in N$ to order $j_2 \in N$.

After completion of order $j \in N$, order j is delivered by a 3PL provider to its destination ϱ_j at its delivery time window $[a_j, b_j]$. If one delivery vehicle arrives before the delivery time window, it should wait until time a_j to unload. Hence the delivery time is also the beginning time of unloading. We consider batch delivery, i.e. one order can be delivered with other orders in one shipment. There is a set of vehicles denoted by K, consisting of several types of vehicles. For each type of vehicles there are a sufficient number of vehicles. Any order can be delivered by any type of vehicle. Vehicle $k \in K$ has a capacity Q^k, which is measured by the number of pallets. Let ϕ_j be the number of pallets to deliver order $j \in N$. One pallet cannot contain more than one order.

Let τ_{0j} denote the transportation time from the plant to the destination of order j. τ_{0j} includes the loading time. Let $\tau_{j_1 j_2}$ denote the transportation time from the destination of order $j_1 \in N$ to the destination of order $j_2 \in N$. Let T denote the constant unloading time of an order at its destination. $\tau_{j_1 j_2} = 0$ if $\varrho_{j_1} = \varrho_{j_2}$. The transportation times respect the triangle rule, i.e. $\tau_{j_1 j_2} + \tau_{j_2 j_3} \geq \tau_{j_1 j_2}$, $j_1, j_2, j_3 \in N$.

There are two types of transportation: direct delivery from the plant to one destination; routing delivery from the plant to several destinations in one shipment. There is a limit of length of any shipment, denoted by L.

In a direct delivery, the transportation cost from the plant to the destination of order $j \in N$ with vehicle $k \in K$, is denoted by h_{0j}^k. And the transportation time is τ_{0j}.

In a routing delivery, the transportation cost is equal to the most expensive direct batch cost of one order among the orders of this shipment plus the total drop costs. A drop cost occurs when we deliver more than one destination in one shipment. Let φ^k denote the drop cost per destination with vehicle $k \in K$. The transportation time is equal to the traveling time of the shipment plus the unloading times. Clearly, the direct delivery is a special case of the routing delivery.

Let σ denote a *production schedule* that specifies how to assign each order on machines and when each order is processed on its assigned machine(s). Let θ denote a *delivery schedule* that specifies how many batches are used, which orders are in each batch, when each batch departs, and what is the traveling route for each batch. Let (σ, θ) denote an *integrated schedule* that specifies a production schedule and a delivery schedule.

The manufacturer's objective is to minimize the total setup cost, denoted by SC. The 3PL provider's objective is to minimize the transportation cost, denoted by TC, which is the sum of transportation costs of all batches.

We consider two scenarios: (1) the manufacturer and the 3PL provider decide their individual schedule without coordination in a consecutive order (i.e. first the manufacturer, then the 3PL provider); (2) the manufacturer and the 3PL provider coordinate to decide concurrently an integrated schedule. The scheduling problems are formally defined as follows.

1. **Individual scheduling problems.**

   (a) **Manufacturer's problem.** The problem is to determine a production schedule minimizing SC subject to deadlines. We follow the three-field classification $\alpha|\beta|\gamma$ introduced by Graham et al. (1979). This is a production scheduling problem minimizing the total setup cost with unrelated parallel machines, machine release times, splitting of order, deadlines and sequence-dependent setup times, denoted by $R, \gamma^e|split, \overline{d}_j, s_{ij}|SC$.

(b) **3PL provider's problem.** The problem is to determine a delivery schedule minimizing TC subject to release dates of orders and delivery time windows. The completion time of each order imposes a release date of the order for delivery. The problem is a heterogeneous vehicle routing problem with time windows and release dates (HVRPTWRD).

2. **Coordinated scheduling problem.** The integrated model is adopted to determine an integrated schedule minimizing $SC + TC$ subject to machine availability constraints and delivery time windows. A savings-sharing scheme is necessary to motivate the coordination. Using the five-field notation proposed by Chen (2010), the integrated scheduling problem can be denoted by $R, \gamma^e, split|[a_j, b_j]|V(\infty, Q^k), routing|u|SC + TC$, where R means the unrelated parallel machines, *split* means the order splitting in production, $V(\infty, Q^k)$ and *routing* mean the routing delivery with sufficient heterogeneous vehicles, and $u \leq n$ represents the number of customers.

Example: To illustrate the individual and integrated problems, we consider the following example.

- Number of orders $n = 2$, number of machines $m = 2$.
- Quantity of items of order $j \in N$: $q_j = 25$.
- Processing time of unit item of order $j \in N$ on machines: $p_j^1 = 1$ and $p_j^2 = 2$.
- Setup times: $s_{01} = 5$, $s_{02} = 4$, $s_{12} = s_{21} = 3$.
- Cost of unit setup time: $\rho = 100$.
- Machines release times are zero.
- There are 2 identical vehicles with capacity of 20.
- Number of pallets to deliver order $j \in N$: $\phi_j = 10$.
- Unloading time $T = 1$, and limit of length of shipment $L = 30$.
- Transportation times: $\tau_{01} = 10$, $\tau_{02} = 15$, $\tau_{12} = \tau_{21} = 12$.

- Delivery time windows: $[a_1, b_1] = [50, 60]$ and $[a_2, b_2] = [60, 70]$.
- Transportation costs with vehicle $k \in K$: $h_{01}^k = 750$ and $h_{02}^k = 1000$.
- Drop costs of vehicle $k \in K$: $\psi^k = 100$.

1. **Individual scheduling problems.** Figure 6.1 illustrates an optimal production schedule for manufacturer's individual scheduling problem. With $C_1 = 30$ and $C_2 = 54$, two orders cannot be delivered in one shipment because of the deadline of order 1. Hence we have $SC = 100 * (5 + 4) = 900$, $TC = 750 + 1000 = 1750$, total cost is equal to $SC + TC = 2650$.

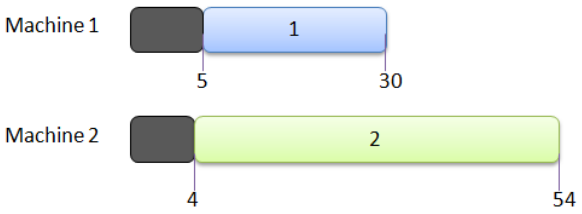


Figure 6.1: Optimal production schedule without coordination

2. **Coordinated scheduling problem.** Figure 6.2 illustrates a production schedule for coordinated scheduling problem. In this schedule, order 2 is split: 10 items are processed on machine 1 and 15 items are processed on machine 2. The setup cost increases to $SC = 100 * (5 + 4 + 3) = 1200$. With $C_1 = 30$ and $C_2 = 43$, the two orders can be delivered in one shipment: the shipment departs at time 43, drops order 1 at time 53 and reaches the destination of order 2 at time 66. Hence we have $TC = 1000 + 100 = 1100$, total cost is equal to $SC + TC = 2300$. The benefit of coordination of total cost is 13.2%.

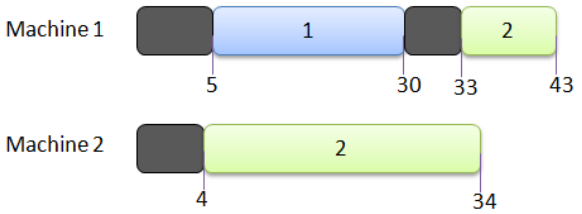


Figure 6.2: Optimal production schedule with coordination

6.3 Individual Scheduling Problems

In the non-coordinated scenario, the manufacturer and the 3PL provider decide their individual schedule consecutively. We provide a mixed integer linear programming (MILP) model for each individual scheduling problem.

6.3.1 Manufacturer's Problem

We recall that the single machine scheduling problem $1|s_{ij}|C_{\max}$ is NP-hard (Bruno and Downey 1978). Since this single machine scheduling problem is a special case of the manufacturer's problem, the manufacturer's problem is also NP-hard. Here, we consider that the deadline $\overline{d}_j$ of order $j \in N$ is equal to $b_j - \tau_{0j}$.

We provide a MILP model, which is similar to the model of Zhu and Heady (2000). We introduce two fictive orders 0 and $n+1$. We define decision variables as follows.

$$X_{ij}^e = \begin{cases} 1, & \text{if order } i \text{ precedes order } j \text{ on machine } e, i = 0, \ldots, n, \\ & j = 1, \ldots, n+1, i \neq j, e \in M \\ 0, & \text{otherwise} \end{cases}$$

Y_j^e = number of items of order j processed on machine e, $j \in N$, $e \in M$.

C_j = completion time of order j, $j \in N$.

MILP6.1:

$$\min \rho \sum_{e=1}^{m} \sum_{i=0}^{n} \sum_{j \in N, j \neq i} s_{ij} X_{ij}^e \tag{6.3.1}$$

$$\text{s.t.} \quad \sum_{i=0,\ldots,n, i \neq j} X_{ij}^e \leq 1, \qquad j = 1, \ldots, n+1, e \in M \tag{6.3.2}$$

$$\sum_{i=0,\ldots,n, i \neq j} X_{ij}^e - \sum_{g=1,\ldots,n+1, g \neq j} X_{jg}^e = 0, \qquad j \in N, e \in M \tag{6.3.3}$$

$$\sum_{i=0,\ldots,n, i \neq j} X_{ij}^e q_j \geq Y_j^e, \qquad j \in N, e \in M \tag{6.3.4}$$

$$\sum_{i=0,\ldots,n, i \neq j} X_{ij}^e \leq Y_j^e, \qquad j \in N, e \in M \tag{6.3.5}$$

$$\sum_{e=1}^{m} Y_j^e = q_j, \qquad j \in N \tag{6.3.6}$$

$$C_j \leq \overline{d}_j, \qquad j \in N \tag{6.3.7}$$

$$C_j - C_i \geq p_j^e Y_j^e + X_{ij}^e s_{ij} + (X_{ij}^e - 1)(p_j^e q_j + \max\{b_i, b_j\}), \quad i, j \in N, i \neq j, e \in M \tag{6.3.8}$$

$$C_j \geq X_{0j}^e(\gamma^e + s_{0j}) + p_j^e Y_j^e, \quad j \in N, e \in M \tag{6.3.9}$$

$$X_{ij}^e \in \{0,1\}, \quad i = 0, \ldots, n, j = 1, \ldots, n+1, i \neq j, e \in M \tag{6.3.10}$$

$$Y_j^e \in \mathbb{N}, \quad j \in N, e \in M \tag{6.3.11}$$

The objective function (6.3.1) minimizes the total setup cost. Constraints (6.3.2) ensure that one order is processed on each machine once at most. Constraints (6.3.3) impose that for each order $j \in N$, the number of its direct predecessors is equal to the number of its direct successors on each machine. Constraints (6.3.4)-(6.3.5) impose the relation between variables X_{ij}^e and Y_j^e: if $Y_j^e > 0$, then $\sum_{i=0,\ldots,n,i\neq j} X_{ij}^e > 0$, otherwise $\sum_{i=0,\ldots,n,i\neq j} X_{ij}^e = 0$. Constraints (6.3.6) ensure that all orders are processed. Constraints (6.3.7) enforce the order deadline restriction. In constraints (6.3.8), if order i precedes order j on machine e, i.e., $X_{ij}^e = 1$, we ensure that the completion time of order j is far enough after that of order i to include the processing time of processed parts of order j and setup time for order j on machine e. Otherwise, i.e., $X_{ij}^e = 0$, we have $C_j - C_i \geq -\max\{b_i, b_j\} \geq p_j^e Y_j^e - p_j^e q_j - \max\{b_i, b_j\}$, hence constraints (6.3.8) are always valid. Constraints (6.3.9) enforce the machine release time restriction. Constraints (6.3.10)-(6.3.11) give the domain of definition of each variable.

6.3.2 3PL Provider's Problem

In the 3PL provider's problem, if the delivery destinations are given for one shipment, the transportation cost of this shipment is fixed. Because of this difference from the classical VRPTW, we need to prove the complexity of the 3PL provider's problem. We consider the following special case of the 3PL provider's problem:

- each customer has one order only,
- the delivery time window $[a_j, b_j] = [0, \infty]$ for $j \in N$,
- the production completion time $C_j = 0$ for $j \in N$,

- the limit of length of a trip $L = \infty$,
- the vehicles are identical,
- the transportation costs $h^k_{0j_1} = h^k_{0j_2} = h^k$ for $j_1, j_2 \in N$ and $k \in K$.

Let B denote the number of delivery batches. In this case, we have the following equation for the overall transportation cost TC of a delivery schedule with B batches.

$$TC = Bh^k + (n - B)\psi^k \tag{6.3.12}$$

Hence the objective of minimizing TC is equivalent to the objective of minimizing the number of delivery batches B. This special case is the bin packing problem which is NP-hard in the strong sense (Garey and Johnson 1979). In the bin packing problem, objects of different volumes must be packed into a finite number of bins of equal size in a way that minimizes the number of bins used. So the 3PL provider's problem is NP-hard in the strong sense.

Then, we provide a multicommodity network flow MILP model similar to that of Desrochers et al. (1988) for the 3PL provider's problem.

This problem can be defined on a direct graph $G = (V, A)$, where $V = \{0, 1, \ldots, n+1\}$. The vertex $j \in \{1, \ldots, n\}$ represents the destination ϱ_j of order j. The vertex $0, n + 1$ represent the plant and one fictive ending point, denoted by ϱ_0 and ϱ_{n+1} respectively. The arcs represent the paths between two places. Feasible vehicle routes correspond to paths starting at vertex 0 and ending at vertex $n + 1$. We set service time $\psi_i = T$ for vertex $i \in N$ and $\psi_0 = \psi_{n+1} = 0$ for vertexes 0 and $n + 1$. τ_{ij} is the travel time from vertex i to vertex j. We set $\tau_{i,n+1} = 0$, for $i \in \{0, \ldots, n\}$. Each vertex i is associated to a time window $[a_i, b_i]$. And we set $a_0 = \min_{i \in N} C_i$, $b_0 = \max_{i \in N}\{b_i - \tau_{oi}\}$, $a_{n+1} = \min_{i \in N}\{a_i + \psi_i\}$ and $b_{n+1} = \max_{i \in N}\{b_i + \psi_i\}$. Here, completion time C_j of order $j \in N$ is given by the manufacturer. Arc (i, j), for $i, j \in N$ and $i \neq j$, exists only if

- order i can be delivered before order j on respecting their delivery time window, i.e. $a_i + \psi_i + \tau_{ij} \leq b_j$,
- the shipment including orders i and j respects the limit of length of a shipment, i.e. $\tau_{0i} + \psi_i + \max\{\tau_{ij}, a_j - b_i - \psi_i\} \leq L$,

- the completion time of order j is no later than the latest possible departure date of the shipment including orders i and j, i.e. $b_i - \tau_{0i} \geq C_j$.

There exists an arc from vertex 0 to each other vertex, and from each other state to vertex $n+1$. Let $\delta^+(i) = \{j : (i,j) \in A\}$ and $\delta^-(j) = \{i : (i,j) \in A\}$. We define the decision variables as follows.

$$x_{ij}^k = \begin{cases} 1, & \text{if arc } (i,j) \text{ is visited by vehicle } k, (i,j) \in A, k \in K \\ 0, & \text{otherwise} \end{cases}$$

w_i^k = starting time of unloading of vehicle k at vertex i, $i \in V$,$k \in K$.

H^k = transportation cost of the trip accomplished by vehicle k, $k \in K$. Remark that since the number of vehicles of each type is sufficient, we can suppose that each vehicle is assigned at most one trip.

MILP6.2:

$$\min \sum_{k \in K} H^k \tag{6.3.13}$$

$$\text{s.t.} \quad \sum_{k \in K} \sum_{j \in \delta^+(i)} x_{ij}^k = 1, \qquad i \in N \tag{6.3.14}$$

$$\sum_{j \in \delta^+(0)} x_{0j}^k = 1, \qquad k \in K \tag{6.3.15}$$

$$\sum_{i \in \delta^-(j)} x_{ij}^k - \sum_{i \in \delta^+(j)} x_{ji}^k = 0, \qquad k \in K, j \in N \tag{6.3.16}$$

$$w_j^k \geq w_i^k + \psi_i + \tau_{ij} - (1 - x_{ij}^k) \max\{b_i, b_j\}, \qquad k \in K, (i,j) \in A \tag{6.3.17}$$

$$a_i \leq w_i^k \leq b_i, \qquad k \in K, i \in V \tag{6.3.18}$$

$$\sum_{i \in N} \sum_{j \in \delta^+(i)} x_{ij}^k \phi_i \leq Q^k, \qquad k \in K \tag{6.3.19}$$

$$w_0^k \geq C_j \sum_{i \in \delta^-(j)} x_{ij}^k, \qquad j \in N, k \in K \tag{6.3.20}$$

$$w_j^k - w_0^k \leq L + (1 - \sum_{i \in \delta^-(j)} x_{ij}^k) b_j, \qquad j \in N, k \in K \tag{6.3.21}$$

$$H^k \geq h_{0j}^k \sum_{i \in \delta^-(j)} x_{ij}^k + \varphi^k (\sum_{(u,v) \in A, \varrho_u \neq \varrho_v} x_{uv}^k - 2) \qquad j \in N, k \in K \tag{6.3.22}$$

$$x_{ij}^k \in \{0,1\}, \qquad k \in K, (i,j) \in A \tag{6.3.23}$$

$$H^k \geq 0, \qquad k \in K \tag{6.3.24}$$

The objective function (6.3.13) minimizes the transportation cost. Constraints (6.3.14) ensure that one order is delivered once. Constraints (6.3.15) ensure that one vehicle is used once. Constraints (6.3.16) state that the solution satisfy the flow conservation at each vertex. Constraints (6.3.17)-(6.3.18) ensures that each order is delivered at its destination in the delivery time windows. Constraints (6.3.19)-(6.3.20) enforce the vehicle capacity restriction and the order availability restriction. Constraints (6.3.21) enforce the delivery length restriction. Constraints (6.3.22) calculate the transportation cost. Constraints (6.3.23)- (6.3.24) give the domain of definition of each variable.

6.4 Coordinated Scheduling Problems

We adopt an integrated model minimizing $SC + TC$ subject to machine availability constraints and delivery time windows. The objective is to optimize the performance of the global supply chain. Clearly, the 3PL provider can benefit from this coordination while the cost of the manufacturer cannot be improved or can be even higher. We propose a savings-sharing mechanism to motivate the manufacturer to join the coordination. The 3PL provider recompense the manufacturer. In order to guarantee the benefit for both actors, the amount of compensation P has to respect the formulation

$$TC_2 + P \leq TC_1 \tag{6.4.25}$$

$$SC_2 - P \leq SC_1 \tag{6.4.26}$$

where TC_1 and TC_2 are the transportation costs in the individual schedule and the integrated schedule respectively. SC_1 and SC_2 are the total setup costs in the individual schedule and the integrated schedule respectively.

In what follows, we propose a nonlinear programming model and a heuristic to solve the integrated scheduling problem. Since the individual scheduling problems are NP-hard, the integrated scheduling problem is also NP-hard.

6.4.1 Nonlinear programming model

We combine MILP6.1 and MILP6.2 to construct a nonlinear programming model for the integrated scheduling problem. Since C_j and x_{ij}^k are both decision variables, this model is nonlinear.

NLP6.1:

$$\min \rho \sum_{e=1}^{m} \sum_{i=0}^{n} \sum_{j \in N, j \neq i} s_{ij} X_{ij}^e + \sum_{k \in K} H^k \tag{6.4.27}$$

$$\text{s.t.} \quad (6.3.2) - (6.3.11)$$

$$(6.3.14) - (6.3.24)$$

6.4.2 Two-phase iterative heuristic

Absi et al. (2014) proposed a two-phase iterative heuristic to solve an integrated problem considering the integration of production planning and vehicle routing decisions. They considered the production planning different from the production scheduling, and the vehicle routing problem without delivery time windows. We propose a similar two-phase iterative heuristic (see algorithm 5) to solve our integrated scheduling problem.

Algorithm 5: Two-phase iterative heuristic

```
1 Initialize F_v^j = 0 and η_j = 1, for j ∈ N and v ∈ I_j;
2 while ending criterion do
3     Solve the production scheduling problem;
4     Remove all idles times and update C_j for j ∈ N;
5     Solve the distribution scheduling problem with fixed C_j for j ∈ N;
6     Update the best solution so far;
7     Update F_v^j, for j ∈ N and v ∈ I_j;
8     Update η_j for j ∈ N;
```

In the first phase, we solve a production scheduling problem in which an approximate of the transportation cost is integrated. We first determine the discrete possible delivery times in each delivery time windows, like $\{a_j, a_j+1, \ldots, b_j\}$ for the delivery time window

$[a_j, b_j]$ and $j \in N$. Let I_j denote a set of indexes of possible delivery times of order $j \in N$ and t_v^j denote the v^{th} possible delivery time for $v \in I_j$. In order to evaluate the transportation cost, we introduce F_v^j to represent an estimation of the transportation cost of order j if order j is delivered at time t_v^j, $j \in N$ and $v \in I_j$. We introduce a decision variable λ_v^j which is equal to 1 if t_v^j is chosen, and 0 otherwise. And we use a parameter η_j for $j \in N$ to modify the completion time constraints (see constraints 6.4.29 of MILP6.4). In the obtained production schedule of the first phase, we remove all idle times on each machine and update C_j for $j \in N$. The anticipation of completion times offers a better input for the second phase.

In the second phase, we solve the distribution scheduling problem with fixed C_j. We use the solution of the second phase to update F_v^j and η_j for next iteration. The procedure stops when a fixed number of iterations is reached or the solution is not improved for a fixed number of iterations.

(i) The production phase

In this phase, we propose a MILP to solve the production scheduling problem in which an approximate of transportation cost is integrated. The decision variable λ_v^j is equal to 1 if t_v^j is chosen for $j \in N$ and $v \in I_j$, and 0 otherwise. The other decision variables are introduced in MILP6.1.

MILP6.3:

$$\min \rho \sum_{e=1}^{m} \sum_{i=0}^{n} \sum_{j \in N, j \neq i} s_{ij} X_{ij}^e + \sum_{j=1}^{n} \sum_{v \in I_j} F_v^j \lambda_v^j \tag{6.4.28}$$

$$\text{s.t.} \quad C_j \leq \eta_j(\lambda_v^j t_v^j + (1 - \lambda_v^j) b_j - \tau_{0j}), \quad j \in N, v \in I_j \tag{6.4.29}$$

$$\sum_{v \in I_j} \lambda_v^j = 1, \quad j \in N \tag{6.4.30}$$

$$\lambda_v^j \in \{0, 1\}, \quad j \in N, v \in I_j \tag{6.4.31}$$

$$(6.3.2) - (6.3.6)$$

$$(6.3.8) - (6.3.11)$$

The objective function (6.4.28) minimizes the sum of the total setup cost and the approximate transportation cost. Constraints (6.4.29) ensure that if $\lambda_v^j = 1$, the completion time $C_j \leq \eta_j(t_v^j - \tau_{0j})$, otherwise $C_j \leq \eta_j(b_j - \tau_{0j})$. And η_j is a parameter to

control the degree to force the reduction of completion time. Initially, η_j is equal to 1 and is reduced at the end of each iteration. Constraints (6.4.30) ensure that each order is delivered exactly once.

At the end of the first phase, in the obtained production schedule, we remove all idle times and update C_j for $j \in N$.

(ii) The distribution phase

In this phase, we use MILP6.2 to solve the HVRPTWRD with fixed C_j for $j \in N$. Then, we use the solution of second phase to update F_v^j and η_j (see algorithm 6).

Algorithm 6: Procedure of updating F_v^j and η_j

for $j \in N$ **do**
  for $v \in I_v$ **do**
    $F_v^j = \infty$;
    for $k \in K$ **do**
      if *order j is visited by vehicle k at time $t_{v'}^j$, where $v' \in I_j$, and the delivery time of order j in vehicle k can be replaced by t_v^j* **then**
        B^k = set of orders delivered by vehicle k;
        H_1 = cheapest cost to deliver the orders of $B^k \setminus \{j\}$;
        $F_v^j = \min\{F_v^j, H^k - H_1\}$;
      if *order j is not visited by vehicle k and can be inserted in the trip of vehicle k and visited at time t_v^j* **then**
        B^k = set of orders delivered by vehicle k;
        H_2 = cheapest cost to deliver the orders of $B^k \cup \{j\}$;
        $F_v^j = \min\{F_v^j, H_2 - H^k\}$;
for $j \in N$ **do**
  if $0.8\eta_j(b_j - \tau_{0j}) \geq \min_{e \in M}(\gamma^e + p_j^e q_j) + s_{0j}$ *and order j is delivered by a vehicle of which the number of delivered pallets is less than or equal to* $\max_{k \in K} Q^k - \min_{i \in N} \phi_i$ **then**
    $\eta_j = 0.8\eta_j$;

In algorithm 6, we update F_v^j with the consideration of two cases:

1. If order j is visited by vehicle k at time $t_{v'}^j$ where $v' \in I_j$, replacing delivery time $t_{v'}^j$ by t_v^j in the trip is allowed if (lines 5-8):

   - The transportation times from its direct predecessor to order j, and from order j to its direct successor are respected.
   - The new trip does not violate the limit of length of a trip.

   If the conditions are satisfied, we update F_v^j by $\min\{F_v^j, H_2 - H^k\}$. Remark that if $v = v'$, the above conditions are satisfied.

2. If order j is not visited by vehicle k, the insertion of order j in vehicle k at time t_v^j is allowed if (lines 9-12):

   - The biggest capacity of vehicle, i.e. $\max_{g \in K} Q^g$, allows.
   - There exists two successively visited vertexes which allow the insertion of order j with delivery time t_v^j, i.e., the transportations times are respected.
   - The new trip does not violate the limit of length of a trip.

   If the conditions are satisfied, we update F_v^j by $\min\{F_v^j, H_2 - H^k\}$.

   After the consideration of all vehicles, F_v^j represents the cheapest transportation cost to deliver order j at time t_v^j for $j \in N$ and $v \in I_j$.

   Moreover, we explain how to find the cheapest cost to deliver a set of orders B (lines 7 and 11). For given B and vehicle $k \in K$, the corresponding transportation cost is fixed. Hence we choose the cheapest vehicle to deliver this set of orders.

Concerning the parameter η_j for $j \in N$, if η_j is small enough or order j is delivered by a vehicle in which the size of delivered orders is close to the biggest vehicle capacity, we do not change η_j, otherwise we reduce η_j by 20%.

In the distribution schedule obtained in the second phase, F_v^j approximates the transportation cost if order j is delivered at time t_v^j, $j \in N$ and $v \in I_j$. In next iteration, for each order $j \in N$, the algorithm may choose another delivery time with smaller transportation cost. This modification of delivery time of each order can influence the

production completion time of each order in the first phase (see constraints 6.4.29 of MILP6.3). The objective of the reduction of η_j is to force the algorithm to reduce the production completion time of order j and increase the opportunity to find a better transportation cost in the second phase.

6.5 Computational Results

In this section, we evaluate the feasibility of the approach and the potential benefit of coordination. We propose an approach of generation of instances and analyze the results with small instances.

The benefit of coordination is measured by comparing the integrated schedule generated by the heuristic with the individual schedules generated by MILP6.1 and MILP6.2. The algorithms are implemented in C++ and Cplex V12.5.1. The experiments are carried out on a DELL 2.50GHz personal computer with 8GB RAM.

We consider $n \in \{5, 10, 15, 20\}$ and $m \in \{2, 3, 5\}$. The integers q_j and ϕ_j, for $j \in N$, are generated from the uniform distributions [50,200] and $[q_j/10, q_j/5]$ respectively. The processing times of unit item p_j^e, for $j \in N$ and $e \in M$, are generated from the uniform distribution [0.01, 0.1]. The machines release times γ^e, for $e \in M$, are generated from the uniform distribution [0,4]. The integers s_{ij}, for $i = 0, \ldots, n$, $j \in N$, are generated from the uniform distribution $[0.1 \min\{\max_{e\in M} p_i^e q_i, \max_{e\in M} p_j^e q_j\}, 0.5 \min\{\max_{e\in M} p_i^e q_i, \max_{e\in M} p_j^e q_j\}]$. We set $s_{ij} = 0$ for $i = j$. In order to guarantee the triangle property, after generation, if $s_{ij} \leq S/2$, where S is the maximum generated setup time, we regenerate another $s_{ij} \in]S/2, S]$. The cost per unit setup time ρ is equal to 100.

We suppose that one customer has only one order. The customers are divided into two groups, $N_1 = \{1, \ldots, n/2\}$ and $N_2 = \{n/2 + 1, \ldots, n\}$. The integers τ_{0j}, for $j \in N_1$ are generated from the uniform distribution $[15, 19]$ and for $j \in N_2$ from $[25, 29]$. If i, j are in the same group, the integers τ_{ij} are generated from the uniform distribution $[4, 6]$, otherwise from $[15, 19]$. We set $\tau_{ij} = \tau_{ji}$ and set $\tau_{ij} = 0$ if $\varrho_i = \varrho_j$. The limit length of a trip $L \in \{45, 60\}$. The unloading time $T = 1$. The integer lower bounds of time windows a_j, for $j \in N$, are generated from the uniform distribution $[B/2, B]$, where

$B = \max_{e\in M} \gamma^e + \max_{e\in M} \sum_{j\in N} p_j^e q_j/m + L + 0.75nS$ represents an estimated delivery time if all orders begin their processing at the latest machine release time, each order is split in m parts and processed on m machines, and the transportation time of the trip reaches the limit L. The integer upper bounds of time windows b_j, for $j \in N$, are generated from the uniform distribution $[a_j + \epsilon - 5, a_j + \epsilon]$, where $\epsilon_2 = \{10, 15\}$. There are two types of vehicles and $2n$ vehicles totally, i.e., $K = \{1, \ldots, 2n\}$. For $k \leq n$ and $j \in N$, $Q^k = 30$, $\varphi^k = 50$ and $h_{0j}^k = 50\tau_{oj}$. For $k > n$ and $j \in N$, $Q^k = 60$, $\varphi^k = 80$ and $h_{0j}^k = 80\tau_{oj}$. 10 instances are generated for each combination of parameters n, m, L and ϵ. Totally 480 instances are generated.

We impose 3 minutes as a limit of execution time of a MILP. We generate individual schedules in three steps: first apply MILP6.1 to create a production schedule, then remove the idle times in the obtained production schedule, and finally apply MILP6.2 to create a distribution schedule. We apply the two-phase iterative heuristic to generate an integrated schedule. Concerning the ending criterion, we set that the total number of iterations cannot exceed 6 and the number of iterations without improvement cannot exceed 3.

Table 6.1: Average computational times of execution of heuristic

n	5	10	15	20
Time	3.34	238.28	952.30	929.29

Table 6.2: Benefit of coordination

	Improved			Average Benefit			Max Benefit		
n	$m=2$	$m=3$	$m=5$	$m=2$	$m=3$	$m=5$	$m=2$	$m=3$	$m=5$
5	25.00%	20.00%	30.00%	2.83%	2.40%	3.15%	21.43%	29.70%	24.28%
10	32.50%	20.00%	32.50%	1.16%	0.68%	1.00%	14.69%	6.47%	6.13%
15	57.50%	57.50%	67.50%	1.52%	1.41%	1.47%	6.61%	6.51%	5.29%
20	42.50%	77.50%	55.00%	1.09%	2.55%	1.63%	5.72%	7.90%	7.98%

Table 6.1 and Table 6.2 illustrate the benefit of coordination. The measures are described as follows.

Time: the average CPU time in seconds to execute the heuristic.

Improved: the percentage of instances which has a positive benefit.

Benefit: the benefit of coordination measured by

$$\frac{SC_1 + TC_1 - SC_2 - TC_2}{SC_1 + TC_1} \tag{6.5.32}$$

where SC_1 and TC_1 are the values of objective functions of the individual schedules, and SC_2 and TC_2 are the values of objective functions of the integrated schedule.

From Table 6.1, one can observe that the average execution time of the heuristic grows rapidly. When $n = 5$, all MILPs can be solved optimally in the given time. When $n \geq 15$, we observe a difficulty for MILP6.2 which cannot find an optimal solution in the given time.

From Table 6.2, we observe that the average benefit of coordination is not significant. We think that this small benefit is influenced by: the possible lack of generality of instances and the low efficiency of MILP6.2 in the second phase. In what follows, we explain the above conclusions by analyzing the indicators "Improved"and "Max Benefit". When $n = 5$ and 10, we find that there exists the instance with a significant benefit which can reaches 29.7% when $n = 5$ and 14.69% when $n = 10$. At the same time, we find more than 67.5% of instances which cannot be improved. Since the MILPs can find an optimal solution for the instances with $n = 5$ and 10 in the given time, we have the reason to doubt the generality of instances. When $n = 15$ and 20, more than 42.5% of instances can be improved, however the low efficiency of MILP6.2 impedes the improvement of the transportation cost.

The significant maximum benefit of coordination verifies the feasibility of the heuristic and the potential benefit of coordination. Moreover, the results point out the possible lack of generality of instances and the low efficiency of MILP6.2 in the second phase. In order to improve the efficiency of MILP6.2 in the second phase, we tested another time-expanded network flow-based model and found that the new MILP is less efficient than MILP6.2. For future research, we need to provide a better approach of generation of instances, and develop an efficient exact algorithm or an efficient heuristic for the distribution phase of the two-phase iterative heuristic.

6.6 Conclusions

In this chapter, we investigated a production and outbound distribution scheduling problem in a supply chain with a manufacturer, a 3PL provider and several customers. This problem is proposed by an enterprise working in the packaging industry. The manufacturer's objective is to minimize the total setup cost and the 3PL provider's objective is to minimize the transportation cost. We first proposed MILP models for individual scheduling problems. Then we provided a nonlinear programming model and a two-phase iterative heuristic for integrated scheduling problem, and developed a savings-sharing mechanism to motivate the coordination.

We also proposed an approach of generation of instances and evaluated the benefit of coordination using numerical experiments with small instances. The significant maximum benefit of coordination verified the feasibility of the heuristic and the potential benefit of coordination. The average benefit of coordination is not significant, which may be influenced by the possible lack of generality of instances and the low efficiency of MILP6.2 in the second phase.

We pointed out the need to improve the approach of generation of instances and the efficiency of the algorithm for the distribution phase of the two-phase iterative heuristic. In order to evaluate the performance of the two-phase iterative heuristic, one might develop a meta-heuristic and compare their solutions using numerical experiments.

Chapter 7

Conclusions and Perspectives

In this dissertation, we investigated three supply chain scheduling problems in the make-to-order business model.

In part I, we addressed a production and interstage distribution scheduling problem. In the considered supply chain, a manufacturer outsources the transportation of semi-finished products between the two production locations to a 3PL provider. We considered the regular and express transportation modes. In chapter 3, we discussed the individual scheduling problems with the consideration of different bargaining powers of the two decision makers and different transportation settings. Different from the classical production scheduling problems and distribution scheduling problems, the individual scheduling problems considered some imposed constraints of the decision maker with a dominant bargaining power and the reaction of the other decision maker. In chapter 4, we discussed the mechanisms of coordination when one of the imposed constraints was relaxed and when all imposed constraints were removed. For the individual scheduling problems and coordinated scheduling problems, we provided polynomial-time algorithms or proved their NP-completeness. We performed experiments to investigate the benefits accruing from coordination between the two parties in two coordinated scenarios. The results showed that the relaxation of the transportation responsiveness can bring a significant benefit for the 3PL provider and the coordination without any imposed constraints can bring better benefits to both actors.

In part II, we investigated the coordination in outbound distribution environment.

Firstly, in chapter 5, we considered a production and outbound distribution schedul-

ing problem with the consideration of order release dates in production and delivery deadlines. In this problem, we considered a single machine production and a direct batch delivery to one customer. Moreover, we considered an important feature in production and distribution: splittable orders or non-splittable orders. We reviewed the individual production scheduling problems and provided two polynomial-time algorithms to solve the distribution scheduling problems. We provided a branch-and-bound algorithm for the coordinated scheduling problem with non-splittable production and distribution (NSP-NSD problem). In order to evaluate the performance of the branch-and-bound algorithm, we provided two MILP models which extended the well-known positional and disjunctive scheduling models. The computational results show that the branch-and-bound algorithm outperforms the two MILP models.

Secondly, in chapter 6, we investigated a production and outbound distribution scheduling problem in the real industry environment. In this system, the production and distribution schedules are decided without coordination. We first provided two MILP models for the individual scheduling problems. These problems are both NP-hard because of the complex system consisting of sequence-dependent setup times, unrelated parallel machines and routing batch delivery to several customers with delivery time windows and heterogeneous vehicles. Then, we proposed an integrated model and provided a nonlinear programming model and a two-phase iterative heuristic to solve the integrated scheduling problem. Using numerical experiments with small instances, we verified the feasibility of the heuristic and the potential benefit of coordination. And we also pointed out the need to improve the approach of generation of instances and the efficiency of the algorithm for the distribution phase of the two-phase iterative heuristic, and to develop a meta-heuristic.

We propose some problems for future investigation.

1. *Coordinated production and interstage distribution scheduling problem with fixed delivery departure dates.*

   Chen (2010) pointed out the need of research on the scheduling problems with fixed delivery departure dates in his survey on integrated production and distribution scheduling problems. In chapter 4, we proved that this problem with two criteria, i.e. the makespan and the transportation cost, is NP-hard. There is a need to

develop exact algorithms and heuristics for this problem.

2. *Coordinated production, interstage distribution and outbound distribution scheduling problem.*

   In the literature, few paper considers the supply chain with both interstage distribution and outbound distribution. Lee and Chen (2001) considered a supply chain with both interstage and outbound distribution in an integrated production and distribution scheduling problem. However, they considered the two distribution problems separately. We can extend our first problem to a supply chain with a manufacturer, a 3PL provider and a (several) customer(s). The manufacturer has to process a set of orders on one machine at the upstream stage and the downstream stage. The 3PL provider is in charge of the transportation of semi-finished products from the upstream stage to the downstream stage, and the transportation of finished products from the downstream stage to the customer(s).

3. *Integrated production and outbound distribution scheduling problem with release dates, deadlines and preemption.*

   This problem is motivated by our second problem. In this integrated problem, a set of preemptive orders has to be processed on one machine, then be delivered in batches to the customer. Each order has a release date and a delivery deadline. The objective is to minimize the transportation cost subject to delivery deadlines. In chapter 5, we proved that this problem is polynomially solvable in two special cases, and the complexity of this problem in general case is open. There is also a need to investigate this problem in more complex models, e.g., the parallel machines production model, the vehicle routing distribution model, etc.

4. *Coordinated production and distribution scheduling problem with 3PL provider.*

   As observed in chapter 1, the road freight transport is mostly outsourced to independent partners like Third Party Logistics (3PL) providers, improving the efficiency of transport. In the literature, few papers considered the scheduling problem with 3PL provider (Li et al. 2008, Zhong et al. 2010, Agnetis et al. 2014). In our considered problems, we investigated the coordinated scheduling problem with a manufacturer and a 3PL provider. There is a need to investigate coordinated

scheduling problem and mechanisms of coordination in more complex models, e.g., models with several manufacturers and/or several 3PL providers, etc.

Bibliography

[1] N. Absi, C. Archetti, S. Dauzère-Pérès, and D. Feillet. A two-phase iterative heuristic approach for the production routing problem. *Transportation Science*, Published online July 11, 2014.

[2] A. Agnetis, M. A. Aloulou, and L.-L. Fu. Coordination of production and interstage batch delivery with outsourced distribution. *European Journal of Operational Research*, 238(1):130 – 142, 2014a.

[3] A. Agnetis, M. A. Aloulou, and L.-L. Fu. Production and interplant batch delivery scheduling: Dominance and cooperation. *submitted to International Journal of Production Economics*, 2014b.

[4] A. Agnetis, M. A. Aloulou, L.-L. Fu, and M. Y. Kovalyov. Two faster algorithms for coordination of production and batch delivery: a note. *European Journal of Operational Research, accepted*, 2014c.

[5] A. Agnetis, N. G. Hall, and D. Pacciarelli. Supply chain scheduling: Sequence coordination. *Discrete Applied Mathematics*, 154(15):2044–2063, 2006.

[6] A. Allahverdi, C. Ng, T. Cheng, and M. Y. Kovalyov. A survey of scheduling problems with setup times or costs. *European Journal of Operational Research*, 187(3):985 – 1032, 2008.

[7] M. A. Aloulou, A. Bouzaiene, N. Dridi, and D. Vanderpooten. A bicriteria two-machine flow-shop serial-batching scheduling problem with bounded batch size. *Journal of Scheduling*, 17:17–29, 2014.

[8] G. Alvarenga, G. Mateus, and G. de Tomi. A genetic and set partitioning two-phase approach for the vehicle routing problem with time windows. *Computers & Operations Research*, 34(6):1561 – 1584, 2007.

[9] A. Anglani, A. Grieco, E. Guerriero, and R. Musmanno. Robust scheduling of parallel machines with sequence-dependent set-up costs. *European Journal of Operational Research*, 161(3):704 – 720, 2005.

[10] C. Archetti and M. Speranza. The split delivery vehicle routing problem: A survey. In B. Golden, S. Raghavan, and E. Wasil, editors, *The Vehicle Routing Problem: Latest Advances and New Challenges*, volume 43 of *Operations Research/Computer Science Interfaces*, pages 103–122. Springer US, 2008.

[11] C. Artigues and O. Roubellat. An efficient algorithm for operation insertion in a multi-resource job-shop schedule with sequence-dependent setup times. *Production Planning & Control*, 13(2):175–186, 2002.

[12] I. Averbakh, O. Berman, and I. Chernykh. A 6/5-approximation algorithm for the two-machine routing open-shop problem on a two-node network. *European Journal of Operational Research*, 166(1):3 – 24, 2005.

[13] I. Averbakh and Z. Xue. On-line supply chain scheduling problems with preemption. *European Journal of Operational Research*, 181(1):500 – 504, 2007.

[14] T. Aydinliyim and G. L. Vairaktarakis. Coordination of outsourced operations to minimize weighted flow time and capacity booking costs. *Manufacturing & Service Operations Management*, 12(2):236–255, 2010.

[15] E. Baker and J. Schaffer. Solution improvement heuristics for the vehicle routing and scheduling problem with time window constraints. *American Journal of Mathematical and Management Sciences*, 6(3):261–300, 1986.

[16] K. R. Baker, E. L. Lawler, J. K. Lenstra, and A. H. G. R. Kan. Preemptive scheduling of a single machine to minimize maximum cost subject to release dates and precedence constraints. *Operations Research*, 31(2):381–386, 1983.

[17] R. Baldacci, M. Battarra, and D. Vigo. Routing a heterogeneous fleet of vehicles. In B. Golden, S. Raghavan, and E. Wasil, editors, *The Vehicle Routing Problem: Latest Advances and New Challenges*, volume 43 of *Operations Research/Computer Science Interfaces*, pages 3–27. Springer US, 2008.

[18] J. F. Bard, G. Kontoravdis, and G. Yu. A branch-and-cut procedure for the vehicle routing problem with time windows. *Transportation Science*, 36(2):250–269, 2002.

[19] J. Behnamian, M. Zandieh, and S. Fatemi Ghomi. Parallel-machine scheduling problems with sequence-dependent setup times using an aco, sa and vns hybrid algorithm. *Expert Systems with Applications*, 36(6):9637 – 9644, 2009.

[20] R. Bent and P. Van Hentenryck. A two-stage hybrid local search for the vehicle routing problem with time windows. *Transportation Science*, 38(4):515–530, 2004.

[21] J. Berger, M. Barkaoui, and O. Bräysy. A route-directed hybrid genetic approach for the vehicle routing problem with time windows. *INFOR*, 41:179194, 2003.

[22] J.-C. Billaut, I. Gribkovskaia, and V. Strusevich. An improved approximation algorithm for the two-machine open shop scheduling problem with family setup times. *IIE Transactions*, 40(4):478 – 493, 2008.

[23] O. Bräysy. Fast local searches for the vehicle routing problem with time windows. *INFOR*, 40:319–330, 2002.

[24] O. Bräysy. A reactive variable neighborhood search for the vehicle-routing problem with time windows. *INFORMS Journal on Computing*, 15(4):347–368, 2003.

[25] O. Bräysy and M. Gendreau. Vehicle routing problem with time windows, part i: Route construction and local search algorithms. *Transportation Science*, 39(1):104–118, 2005.

[26] O. Bräysy and M. Gendreau. Vehicle routing problem with time windows, part ii: Metaheuristics. *Transportation Science*, 39(1):119–139, 2005.

[27] C. Briand, S. Ourari, and B. Bouzouia. An efficient ilp formulation for the single machine scheduling problem. *RAIRO - Operations Research*, 44:61–71, 2010.

[28] P. Brucker. *Scheduling Algorithms*. Springer Publishing Company, Incorporated, 5th edition, 2007.

[29] P. Brucker, S. Knust, T. Cheng, and N. Shakhlevich. Complexity results for flow-shop and open-shop scheduling problems with transportation delays. *Annals of Operations Research*, 129(1-4):81–106, 2004.

[30] P. Brucker, S. Knust, and G. Wang. Complexity results for flow-shop problems with a single server. *European Journal of Operational Research*, 165(2):398 – 407, 2005.

[31] P. Brucker, M. Kovalyov, Y. Shafransky, and F. Werner. Batch scheduling with deadlines on parallel machines. *Annals of Operations Research*, 83(0):23–40, 1998.

[32] J. Bruno and P. Downey. Complexity of task sequencing with deadlines, set-up times and changeover costs. *SIAM Journal on Computing*, 7(4):393–404, 1978.

[33] X. Cai and G. L. Vairaktarakis. Coordination of outsourced operations at a third-party facility subject to booking, overtime, and tardiness costs. *Operations Research*, 60(6):1436–1450, 2012.

[34] J. Carlier. The one-machine sequencing problem. *European Journal of Operational Research*, 11(1):42–47, 1982.

[35] H.-K. Chen, C.-F. Hsueh, and M.-S. Chang. Production scheduling and vehicle routing with time windows for perishable food products. *Computers & Operations Research*, 36(7):2311 – 2319, 2009.

[36] Z.-L. Chen. Integrated production and outbound distribution scheduling: Review and extensions. *Operations Research*, 58(1):130–148, 2010.

[37] Z.-L. Chen and N. G. Hall. Supply chain scheduling: Conflict and cooperation in assembly systems. *Operations Research*, 55(6):1072–1089, 2007.

[38] Z.-L. Chen and G. Pundoor. Integrated order scheduling and packing. *Production and Operations Management*, 18(6):672–692, 2009.

[39] T. E. Cheng, G. Wang, and C. Sriskandarajah. One-operatortwo-machine flowshop scheduling with setup and dismounting times. *Computers & Operations Research*, 26(7):715 – 730, 1999.

[40] W. Cheung and H. Zhou. Using genetic algorithms and heuristics for job shop scheduling with sequence-dependent setup times. *Annals of Operations Research*, 107(1-4):65–81, 2001.

[41] W.-C. Chiang and R. A. Russell. A reactive tabu search metaheuristic for the vehicle routing problem with time windows. *INFORMS Journal on Computing*, 9(4):417–430, 1997.

[42] W. Cook and J. L. Rich. A parallel cutting-plane algorithm for the vehicle routing problem with time windows, 1999.

[43] J.-F. Cordeau and G. Laporte. A tabu search algorithm for the site dependent vehicle routing problem with time windows. *INFOR*, 39:292298, 2001.

[44] J.-F. Cordeau, G. Laporte, J.-Y. Potvin, and M. W. Savelsbergh. Chapter 7 transportation on demand. In C. Barnhart and G. Laporte, editors, *Transportation*, volume 14 of *Handbooks in Operations Research and Management Science*, pages 429 – 466. Elsevier, 2007.

[45] J.-F. Cordeau, G. Laporte, M. W. Savelsbergh, and D. Vigo. Chapter 6 vehicle routing. In C. Barnhart and G. Laporte, editors, *Transportation*, volume 14 of *Handbooks in Operations Research and Management Science*, pages 367 – 428. Elsevier, 2007.

[46] R. Cordone and R. Wolfler Calvo. A heuristic for the vehicle routing problem with time windows. *Journal of Heuristics*, 7(2):107–129, 2001.

[47] G. B. Dantzig and J. H. Ramser. The truck dispatching problem. *Management Science*, 6(1):80–91, 1959.

[48] M. Dawande, H. N. Geismar, N. G. Hall, and C. Sriskandarajah. Supply chain scheduling: Distribution systems. *Production and Operations Management*, 15(2):243–261, 2006.

[49] M. Desrochers, J. Desrosiers, and M. Solomon. A new optimization algorithm for the vehicle routing problem with time windows. *Operations Research*, 40(2):342–354, 1992.

[50] M. Desrochers, J. Lenstra, M. Savelsbergh, and F. Soumis. Vehicle routing with time windows: Optimization and approximation. In B. Golden and A. Assad, editors, *Vehicle Routing: Methods and Studies*, Studies in Management Science and Systems, pages 65–84. North-Holland, 1988.

[51] C. Esswein, J.-C. Billaut, and V. A. Strusevich. Two-machine shop scheduling: Compromise between flexibility and makespan value. *European Journal of Operational Research*, 167(3):796–809, 2005.

[52] . European Commission. Road freight transport vademecum 2010 report, 2011.

[53] M. L. Fisher. Optimal solution of vehicle routing problems using minimum k-trees. *Operations Research*, 42(4):626–642, 1994.

[54] M. L. Fisher, K. O. Jrnsten, and O. B. G. Madsen. Vehicle routing with time windows: Two optimization algorithms. *Operations Research*, 45(3):488–492, 1997.

[55] L.-L. Fu, M. A. Aloulou, and C. Artigues. Solving one-to-one integrated production and outbound distribution scheduling problems with job release dates and deadlines. 10^{th} *International Conference on MOdeling, Optimization and SIMulation - MOSIM14*, 2014.

[56] L. M. Gambardella, E. Taillard, and G. Agazzi. Macs-vrptw: A multiple colony system for vehicle routing problems with time windows. In *New Ideas in Optimization*, pages 63–76. McGraw-Hill, 1999.

[57] A. Garcia-Najera and J. A. Bullinaria. An improved multi-objective evolutionary algorithm for the vehicle routing problem with time windows. *Computers & Operations Research*, 38(1):287 – 300, 2011.

[58] M. Garey and D. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W.H. Freeman, 1979.

[59] M. R. Garey, D. S. Johnson, and R. Sethi. The complexity of flowshop and jobshop scheduling. *Mathematics of Operations Research*, 1(2):117–129, 1976.

[60] H. Gehring and J. Homberger. Parallelization of a two-phase metaheuristic for routing problems with time windows. *Journal of Heuristics*, 8(3):251–276, 2002.

[61] M. Gendreau, G. Laporte, and E. M. Guimaraes. A divide and merge heuristic for the multiprocessor scheduling problem with sequence dependent setup times. *European Journal of Operational Research*, 133(1):183 – 189, 2001.

[62] M. Gendreau and C. Tarantilis. Solving large-scale vehicle routing problems with time windows: The state-of-the-art, 2010.

[63] A. Gharbi and M. Haouari. Minimizing makespan on parallel machines subject to release dates and delivery times. *Journal of Scheduling*, 5(4):329–355, 2002.

[64] H. Gong and L. Tang. Two-machine flowshop scheduling with intermediate transportation under job physical space consideration. *Computers & Operations Research*, 38(9):1267–1274, 2011.

[65] R. L. Graham, E. L. Lawler, J. K. Lenstra, and A. H. G. Rinnooy Kan. Optimization and approximation in deterministic sequencing and scheduling: a survey. In E. J. P.L. Hammer and B. Korte, editors, *Discrete Optimization II*, volume 5 of *Annals of Discrete Mathematics*, pages 287–326. Elsevier, 1979.

[66] L. A. Hall and D. B. Shmoys. Approximation algorithms for constrained scheduling problems. *Proc. 30th Annual Sympos. Foundations Comput. Sci.*, page 134140, 1989.

[67] L. A. Hall and D. B. Shmoys. Jackson's rule for single-machine scheduling: Making a good heuristic better. *Mathematics of Operations Research*, 17(1):22–35, 1992.

[68] N. Hall and C. Potts. The coordination of scheduling and batch deliveries. *Annals of Operations Research*, 135(1):41–64, 2005.

[69] N. G. Hall and Z. Liu. Capacity allocation and scheduling in supply chains. *Operations Research*, 58(6):1711–1725, 2010.

[70] N. G. Hall and C. N. Potts. Supply chain scheduling: Batching and delivery. *Operations Research*, 51(4):566–584, 2003.

[71] W. A. Horn. Some simple scheduling algorithms. *Naval Research Logistics Quarterly*, 21(1):177–185, 1974.

[72] J. Hurink and S. Knust. Makespan minimization for flow-shop problems with transportation times and a single robot. *Discrete Applied Mathematics*, 112(13):199 – 216, 2001.

[73] G. Ioannou, M. Kritikos, and G. Prastacos. A greedy look-ahead heuristic for the vehicle routing problem with time windows. *The Journal of the Operational Research Society*, 52(5):523–537, 2001.

[74] J. R. Jackson. Scheduling a production line to minimize maximum tardiness. Technical report, University of California, Los Angeles, 1955.

[75] S. M. Johnson. Optimal two- and three-stage production schedules with setup times included. *Naval Research Logistics Quarterly*, 1(1):61–68, 1954.

[76] B. Kallehauge, J. Larsen, and O. B. Madsen. Lagrangian duality applied to the vehicle routing problem with time windows. *Computers & Operations Research*, 33(5):1464 – 1487, 2006.

[77] P. Kaminsky. The effectiveness of the longest delivery time rule for the flow shop delivery time problem. *Naval Research Logistics*, 50:257–272, 2003.

[78] H. Kellerer, V. Kotov, F. Rendl, and G. J. Woeginger. The stock size problem. *Operations Research*, 46(3-Supplement-3):S1–S12, 1998.

[79] P. Kilby, P. Prosser, and P. Shaw. Guided local search for the vehicle routing problem with time windows. In S. Voss, S. Martello, I. Osman, and C. Roucairol, editors, *Meta-Heuristics*, pages 473–486. Springer US, 1999.

[80] C. Kille and M. Schwemmer. Top 100 in european transport and logistics services 2013/2014. *Fraunhofer SCS*, 2013.

[81] U. Koc, A. Toptal, and I. Sabuncuoglu. A class of joint production and transportation planning problems under different delivery policies. *Operations Research Letters*, 41(1):54 – 60, 2013.

[82] F. Koehler and S. Khuller. Optimal batch schedules for parallel machines. In F. Dehne, R. Solis-Oba, and J.-R. Sack, editors, *Algorithms and Data Structures*, volume 8037 of *Lecture Notes in Computer Science*, pages 475–486. Springer Berlin Heidelberg, 2013.

[83] N. Kohl, J. Desrosiers, O. B. G. Madsen, M. M. Solomon, and F. Soumis. 2-path cuts for the vehicle routing problem with time windows. *Transportation Science*, 33(1):101–116, 1999.

[84] N. Kohl and O. B. G. Madsen. An optimization algorithm for the vehicle routing problem with time windows based on lagrangian relaxation. *Operations Research*, 45(3):395–406, 1997.

[85] G. Kontoravdis and J. F. Bard. A grasp for the vehicle routing problem with time windows. *ORSA Journal on Computing*, 7(1):10–23, 1995.

[86] S. N. Kumar and R. Panneerselvam. A survey on the vehicle routing problem and its variants. *Intelligent Information Management*, 4(3):66–74, 2012.

[87] H. C. Lau, M. Sim, and K. M. Teo. Vehicle routing problem with time windows and a limited number of vehicles. *European Journal of Operational Research*, 148(3):559 – 569, 2003.

[88] E. L. Lawler. Optimal sequencing of a single machine subject to precedence constraints. *Management Science*, 19(5):544–546, 1973.

[89] C.-Y. Lee and Z.-L. Chen. Machine scheduling with transportation considerations. *Journal of Scheduling*, 4(1):3–24, 2001.

[90] C.-Y. Lee and V. A. Strusevich. Two-machine shop scheduling with an uncapacitated interstage transporter. *IIE Transactions*, 37(8):725–736, 2005.

[91] J. Lenstra, A. Rinnooy Kan, and P. Brucker. Complexity of machine scheduling problems. In P. Hammer, E. Johnson, B. Korte, and G. Nemhauser, editors, *Studies in Integer Programming*, volume 1 of *Annals of Discrete Mathematics*, pages 343 – 362. Elsevier, 1977.

[92] K. Li, V. K. Ganesan, and A. I. Sivakumar. Scheduling of single stage assembly with air transportation in a consumer electronic supply chain. *Computers and Industrial Engineering*, 51(2):264–278, 2006.

[93] K. Li, A. I. Sivakumar, and V. K. Ganesan. Analysis and algorithms for coordinated scheduling of parallel machine manufacturing and 3PL transportation. *International Journal of Production Economics*, 115(2):482–491, 2008.

[94] K. P. Li, V. K. Ganesan, and A. I. Sivakumar. Synchronized scheduling of assembly and multi-destination air-transportation in a consumer electronics supply chain. *International Journal of Production Research*, 43(13):2671–2685, 2005.

[95] Z. Liu and T. Cheng. Scheduling with job release dates, delivery times and preemption penalties. *Information Processing Letters*, 82(2):107 – 111, 2002.

[96] C. Low, C.-M. Chang, R.-K. Li, and C.-L. Huang. Coordination of production scheduling and delivery problems with heterogeneous fleet. *International Journal of Production Economics*, 153(0):139 – 148, 2014.

[97] C. Low, R.-K. Li, and C.-M. Chang. Integrated scheduling of production and delivery with time windows. *International Journal of Production Research*, 51(3):897 – 909, 2013.

[98] L. Lu, J. Yuan, and L. Zhang. Single machine scheduling with release dates and job delivery to minimize the makespan. *Theoretical Computer Science*, 393(13):102 – 108, 2008.

[99] P. Maggu and G. Das. On 2xn sequencing problem with transportation times of jobs. *Pure and Applied Mathematika Sciences*, 12:1–6, 1980.

[100] M. Mastrolilli. Efficient approximation schemes for scheduling problems with release dates and delivery times. *Journal of Scheduling*, 6(6):521–531, 2003.

[101] M. M. Mazdeh, A. N. Esfahani, S. E. Sakkaki, and A. E. Pilerood. Single-machine batch scheduling minimizing weighted flow times and delivery costs with job release times. *International Journal of Industrial Engineering Computations*, 3(3):347–364, 2012.

[102] M. M. Mazdeh, M. Sarhadi, and K. S. Hindi. A branch-and-bound algorithm for single-machine scheduling with batch delivery and job release times. *Computers & Operations Research*, 35(4):1099 – 1111, 2008.

[103] A. S. Mendes, F. M. Müller, P. M. Frana, and P. Moscato. Comparing meta-heuristic approaches for parallel machine scheduling problems. *Production Planning and Control*, 13(2):143–154, 2002.

[104] D. Mester and O. Bräysy. Active guided evolution strategies for large-scale vehicle routing problems with time windows. *Comput. Oper. Res.*, 32(6):1593 –1614, 2005.

[105] D. Miller, H.-C. Chen, J. Matson, and Q. Liu. A hybrid genetic algorithm for the single machine scheduling problem. *Journal of Heuristics*, 5(4):437–454, 1999.

[106] Y. Nagata and O. Bräysy. A powerful route minimization heuristic for the vehicle routing problem with time windows. *Operations Research Letters*, 37(5):333 – 338, 2009.

[107] J. A. Nelder and R. Mead. A simplex method for function minimization. *Computer Journal*, 7:308–313, 1965.

[108] C. Ng and L. Lu. On-line integrated production and outbound distribution scheduling to minimize the maximum delivery completion time. *Journal of Scheduling*, 15(3):391–398, 2012.

[109] A. Palmer, M. J. Saenz, T. V. Woensel, and E. Ballot. Characteristics of collaborative business models. *CO3 position paper*, 2012.

[110] K.-W. Pang. An adaptive parallel route construction heuristic for the vehicle routing problem with time windows constraints. *Expert Syst. Appl.*, 38(9):11939–11946, 2011.

[111] S. S. Panwalkar. Scheduling of a two-machine flowshop with travel time between machines. *The Journal of the Operational Research Society*, 42(7):609–613, 1991.

[112] S. Parragh, K. Doerner, and R. Hartl. A survey on pickup and delivery problems. *Journal für Betriebswirtschaft*, 58(1):21–51, 2008.

[113] S. Parragh, K. Doerner, and R. Hartl. A survey on pickup and delivery problems. *Journal für Betriebswirtschaft*, 58(2):81–117, 2008.

[114] V. Pillac, M. Gendreau, C. Guret, and A. L. Medaglia. A review of dynamic vehicle routing problems. *European Journal of Operational Research*, 225(1):1 – 11, 2013.

[115] M. Pinedo. *Scheduling Theory, Algorithms, and Systems, 2nd ed.* Prentice Hall, Upper Saddle River, NJ, 2002.

[116] C. N. Potts. Analysis of a heuristic for one machine sequencing with release dates and delivery times. *Operations Research*, 28(6):1436–1441, 1980.

[117] C. N. Potts and M. Y. Kovalyov. Scheduling with batching: A review. *European Journal of Operational Research*, 120(2):228 – 249, 2000.

[118] J.-Y. Potvin and J.-M. Rousseau. A parallel route building algorithm for the vehicle routing and scheduling problem with time windows. *European Journal of Operational Research*, 66(3):331 – 340, 1993.

[119] J.-Y. Potvin and J.-M. Rousseau. An exchange heuristic for routeing problems with time windows. *The Journal of the Operational Research Society*, 46(12):1433–1446, 1995.

[120] G. Pundoor and Z.-L. Chen. Scheduling a productiondistribution system to optimize the tradeoff between delivery tardiness and distribution cost. *Naval Research Logistics*, 52(6):571–589, 2005.

[121] R. A. Russell. An effective heuristic for the m-tour traveling salesman problem with some side conditions. *Operations Research*, 25(3):517–524, 1977.

[122] R. A. Russell. Hybrid heuristics for the vehicle routing problem with time windows. *Transportation Science*, 29(2):156–166, 1995.

[123] E. Selvarajah, G. Steiner, and R. Zhang. Single machine batch scheduling with release times and delivery costs. *Journal of Scheduling*, 16(1):69–79, 2013.

[124] M. M. Solomon. Algorithms for the vehicle routing and scheduling problems with time window constraints. *Operations Research*, 35(2):254–265, 1987.

[125] K. E. Stecke and X. Zhao. Production and transportation integration for a make-to-order manufacturing company with a commit-to-delivery business mode. *Manufacturing & Service Operations Management*, 9(2):206–224, 2007.

[126] H. I. Stern and G. Vitner. Scheduling parts in a combined production-transportation work cell. *The Journal of the Operational Research Society*, 41(7):625–632, 1990.

[127] D. N. Tahar, F. Yalaoui, C. Chu, and L. Amodeo. A linear programming approach for identical parallel machine scheduling with job splitting and sequence-dependent setup times. *International Journal of Production Economics*, 99(12):63 – 73, 2006.

[128] E. Taillard, P. Badeau, M. Gendreau, F. Guertin, and J.-Y. Potvin. A tabu search heuristic for the vehicle routing problem with soft time windows. *Transportation Science*, 31(2):170–186, 1997.

[129] L. Tang, J. Guan, and G. Hu. Steelmaking and refining coordinated scheduling problem with waiting time and transportation consideration. *Computers & Industrial Engineering*, 58(2):239 – 248, 2010.

[130] D. J. Thomas and P. M. Griffin. Coordinated supply chain management. *European Journal of Operational Research*, 94:1–15, 1996.

[131] P. Toth and D. Vigo, editors. *The Vehicle Routing Problem*. Society for Industrial and Applied Mathematics, 2002.

[132] C. A. Ullrich. Integrated machine scheduling and vehicle routing with time windows. *European Journal of Operational Research*, 227(1):152 – 165, 2013.

[133] M. G. Van Buer, D. L. Woodruff, and R. T. Olson. Solving the medium newspaper production/distribution problem. *European Journal of Operational Research*, 115(2):237 – 253, 1999.

[134] A. Vignier, B. Sonntag, and M.-C. Portmann. A hybrid method for a parallel-machine scheduling problem. *Emerging Technologies and Factory Automation, ETFA*, 1:671–678, 1999.

[135] H. Wang and C.-Y. Lee. Production and transport logistics scheduling with two transport mode choices. *Naval Research Logistics*, 52(8):796–809, 2005.

[136] X. Wang and T. Cheng. Heuristics for parallel-machine scheduling with job class setups and delivery to multiple customers. *International Journal of Production Economics*, 119(1):199 – 206, 2009.

[137] G. J. Woeginger. A polynomial-time approximation scheme for single-machine sequencing with delivery times and sequence-independent batch set-up times. *Journal of Scheduling*, 1(2):79–87, 1998.

[138] F. Yalaoui and C. Chu. An efficient heuristic approach for parallel machine scheduling with job splitting and sequence-dependent setup times. *IIE Transactions*, 35(2):183–190, 2003.

[139] S. Zdrzaka. Approximation algorithms for single-machine sequencing with delivery times and unit batch set-up times. *European Journal of Operational Research*, 51(2):199 – 209, 1991.

[140] S. Zdrzaka. Preemptive scheduling with release dates, delivery times and sequence independent setup times. *European Journal of Operational Research*, 76(1):60 – 71, 1994.

[141] S. Zdrzaka. Analysis of approximation algorithms for single-machine scheduling with delivery times and sequence independent batch setup times. *European Journal of Operational Research*, 80(2):371 – 380, 1995.

[142] W. Zhong, Z.-L. Chen, and M. Chen. Integrated production and distribution scheduling with committed delivery dates. *Operations Research Letters*, 38(2):133–138, 2010.

[143] Z. Zhu and R. B. Heady. Minimizing the sum of earliness/tardiness in multi-machine scheduling: a mixed integer programming approach. *Computers & Industrial Engineering*, 38(2):297 – 305, 2000.