# Models and algorithms for a class of production routing problems.

Yantong Li



université PARIS-SACLAY

NNT : 2018SACLE036

Thèse de doctorat

# Modèles et algorithmes pour une classe de problèmes combinés de production et de tournées de véhicules

Thèse de doctorat de l'Université Paris-Saclay
préparée à l'Université d'Evry-Val-d'Essonne

Ecole doctorale n°580 sciences et technologies de l'information et de la communication (STIC)
Spécialité de doctorat : Automatique

Thèse présentée et soutenue à Evry, le 26 Octobre, par

**YANTONG LI**

Composition du Jury :

| | |
|---|---|
| Patrick SIARRY<br>Professeur, Université Paris-Est Créteil | Président |
| Saïd HANAFI<br>Professeur, Université de Valenciennes | Rapporteur |
| Mhand HIFI<br>Professeur, Université de Picardie Jules Verne | Rapporteur |
| Xianyi ZENG<br>Professeur, Ecole Nationale Supérieure des Arts et Industries Textiles | Rapporteur |
| Jean-François CÔTÉ<br>Professeur adjoint Université Laval, Canada | Examinateur |
| Sourour ELLOUMI<br>Professeur, l'ENSTA ParisTech | Examinateur |
| Anass NAGIH<br>Professeur, Université de Lorraine | Examinateur |
| Feng CHU<br>Professeur, Université d'Evry Val-d'Essonne | Directeur de thèse |

# Acknowledgments

I would first like to sincerely thank my supervisor Prof. Feng CHU for her genuine advice, technical expertise, and constant encouragement. This thesis would definitely have not been achievable without her. It has been a great honor and pleasure to work with her.

I am truly grateful to the jury members of my Thesis defense: professor Saïd Hanafi (Université de Valenciennes), professor Mhand Hifi, (Université de Picardie Jules Verne), professor Xianyi Zeng, (Ecole Nationale Supérieure des Arts et Industries Textiles), associate professor Jean-François Côté, (Université Laval, Canada), professor Sourour Elloumi, (l'ENSTA ParisTech), professor Anass Nagih, (Université de Lorraine), and professor Patrick Siarry, (Université Paris-Est Crteil).

Special gratitude is extended to Professors Chengbin Chu, Mengchu Zhou, and Leandro C. Coelho for their valuable suggestions and comments on improving a part of this thesis. I would like to thank Professors Franck Delaplace, Hanna Klaudel, Jean-Marc Delosme, Madame Murielle Bourgeois, Monsieur Thierry Millant in Lab. IBISC for their help during my study in France.

I would like to thank all my friends, so many that I could not cite their names here. Thank you for being with me all the time. I greatly value our friendship.

Finally, biggest thanks go to my beloved family for their unwavering support. My parents, parents-in-law, brother, and sister-in-law, who give me all their unselfish love, and always stand behind me. I greatly appreciate and acknowledge my wife Ying Huang who constantly believes in me and accompanies me. She always gives me unconditional love, meticulous caring, and continuous support. Special thank goes to my baby daughter Sirou Li, who opens a new chapter of my life. She brings me happiness and hope.

Evry, 2018

# Publications

## Journal papers

Y. Li, F. Chu, C. Feng, C. Chu, and M. Zhou. Integrated Production Inventory Routing Planning for Intelligent Food Logistics Systems. *IEEE Transactions on Intelligent Transportation Systems*, pages 1-12, (99)2018 [72].
DOI: https://doi.org/10.1109/TITS.2018.2835145.

Y. Li, F. Chu, C. Chu, and Z. Zhu. An Efficient Three-level Heuristic for the Large-scaled Multi-product Production Routing Problem with Outsourcing. *European Journal of Operational Research*, 272(3):914-927, 2019. [71].
DOI: https://doi.org/10.1016/j.ejor.2018.07.018.

## Conference papers

Y. Li, F. Chu, C. Feng, Z. Yang, and R. Wolfler Calvo. A Production Inventory Routing Planning for Perishable Food with Quality Consideration. *IFAC-PapersOnLine*, pages 407-412, 49(3), 2016. In *14th IFAC Symposium on Control in Transportation Systems* , Istanbul, Turkey, May 18-20, 2016 [73].

Y. Li, F. Chu, C. Chu, W. Zhou, and Z. Zhu. Integrated production inventory routing planning with time windows for perishable food. In *19th IEEE International conference on Intelligent Transportation Systems*, Rio de Janeiro, Brazil, November 1-4, 2016 , pages 2651-2656, 2016 [70].

Y. Li, F. Chu, and K. Chen. Coordinated Production Inventory Routing Planning for Perishable Food. *IFAC-PapersOnLine*, pages 4246-4251, 50(1), 2017. In *20th World Congress of the International Federation of Automatic Control*, Toulouse, France, July 9-14, 2017 [69].

Y. Li, and F. Chu. Bi-objective Optimization of an Integrated Production Inventory Routing Planning for Perishable Food. In *18$^{ème}$ Conférence ROADEF Société Française de Recherche Opérationnelle et Aide à la Décision*, Metz, France, February 22-24, 2017 [68].

# Contents

# Acronyms

**$\epsilon$-CTIH** $\epsilon$-constraint-based two-phase iterative heuristic.

**AKS** adaptive kernel search.

**B&B** branch-and-bound.
**B&C** branch-and-cut.
**B&P** branch-and-price.
**BFPRP** bi-objective food production routing problem.

**DP** dynamic programming.

**FO** fix-and-optimize.
**FPRP** food production routing problem.
**FPRPTW** food production routing problem with time windows.
**FSC** food supply chain.

**HM** hybrid matheuristic.

**IRP** inventory routing problem.

**KS** kernel search.

**LSP** lot-sizing problem.

**MFPRP** multi-plant food production routing problem.
**MILP** mixed integer linear program.
**MPRP** multi-product production routing problem.
**MPRPOS** multi-product production routing problem with outsourcing.

**PDP** production direct-distribution problem.
**PRP** production routing problem.
**PSVRP** production scheduling and vehicle routing problem.

**RO** route-based optimizition.
**RPDP** restricted PDP.

**SC** supply chain.

**TI** two-phase iterative method.
**TLH** three-level heuristic.
**TSP** traveling salesman problem.

**VRP** vehicle routing problem.
**VRPTW** vehicle routing problem with time windows.

# Chapter 1

# Introduction

This thesis investigates a class of production routing problems (PRPs), in which several fundamental activities, i.e., production planning, inventory control and vehicle routing, are jointly optimized to improve the performance of the supply chain (SC). The focus is to propose new models and algorithms for the studied problems. In this chapter, the research motivation is introduced, and then the content and contributions are presented.

## 1.1 Background

Facing the increasing fierce competition in the global market, companies have reached the consensus that the integrated planning of two or more activities of the SC is one of the key factors in improving their competitiveness. The speedy development of advanced information and communication technology has enabled the accomplishment of integrated planning. Two widely studied partially integrated planning problems are the production direct-distribution problem (PDP) and the inventory routing problem (IRP). The PDP aims to make a production and direct-distribution plan to minimize the production, inventory (at both plants and customers), and direct transportation costs [47]. However, the PDP does not consider routing decisions. In a single-plant and multi-customer distribution network, the IRP aims to minimize the total inventory and routing costs by determining the timing and size of each delivery to its customers and the corresponding routes [30]. However, the production quantities for each period are given in advance in the IRP. Recently, companies have set up more integrated planning systems in which the production, inventory, and routing activities are optimized. The optimization problem for implementing such an integrated system is known as the PRP, which has been drawing increasing attention since it was introduced by Chandra [31] in 1993. The PRP can be defined on a SC network where a single production plant produces and distributes a single type of product to a set of customers to meet their dynamic demand over the planning time horizon.

The classic PRP consists of simultaneously optimizing the production, inventory and routing decisions to minimize the total cost. A solution to the PRP determines for each period: 1) how much to produce at the plant; 2) how much to deliver to each customer; 3) how much inventory to hold in the plant and each customer; and 4) how to arrange the vehicle routes for each planned delivery. The PRP is a generalization of the PDP and IRP. It reduces to a PDP if a direct-distribution assumption is made, and becomes an IRP if the production quantity in each period is fixed. Various studies and practices have shown that implementing the PRP can enhance synchronization, reduce product cost and improve service level. Chandra and Fisher [32] point out that an integrated production, inventory and routing planning can reduce the total operating cost by 3% - 20%. The Kellogg Company saved 4.5 million dollars in 1995 with an integrated planning system called Kellogg planning system [29]. Meanwhile, customers can benefit from high service level with low stockout risk through the integration of production, storage, and routing [19], [94]. Although the classic PRP has been widely investigated since the last few decades [1]–[4], [18]–[20], [25], [27], [88], [92], it has not been sufficiently studied. For instance, the multi-product production routing problem (MPRP) has not received enough attention, although companies often deal with multiple products in their production and distribution planning. In addition, most of the existing studies on the PRP focus on the general SC without capturing the important characteristics of some special industry. In particular, the food production routing problem (FPRP) for the food supply chain (FSC), in which the food quality and perishability have to be carefully managed, has not been studied yet.

The FSC is an important branch of the general SC that aims to provide customers with high-quality food products with low cost. It plays an important role in the food industry and is highly related to people's life quality. Despite that the FSC shares some common characteristics with the general SC, it has the following specific features.

1) Most food has a relatively short shelf life and starts deteriorating once being produced. Examples include fresh meat, fresh-cut vegetables, dairy products, and bakery products. The value or quality of perishable food products decay when being stored or delivered [33].

2) The process of food production, packaging, storage and transportation needs to respect strict conditions and regulations. Most of the perishable food should be stored and transported under a certain range of recommended temperatures, otherwise, the food quality and safety cannot be guaranteed. Improper packaging, storage and transportation will increase the amount of food wastes, food losses and the risk of food safety, while respecting these conditions and regulations will imperatively increase

cost. In addition, the delivery of food products may respect more strict delivery time windows.

3) Food price fluctuates severely. As the food quality decreases, the selling price could not keep the same. In reality, perishable food products are frequently sold with promotions (under certain discount policies). Thus the total revenue may be impacted due to adopted discount policies.

4) Consumers have high requirements for food quality, safety and traceability. The customers today are more critical on food quality than ever before. They would like to know more information about the food products they are buying, e.g., the quality, origin, and storage and transport conditions. Customer satisfaction significantly depends on the quality of food received by customers.

Considering the above-mentioned characteristics of the FSC, it is meaningful to study the FPRP capturing these characteristics. However, studies on the FPRP are nearly blank in the literature. This motivates us to fill in the research gap by studying the FPRP with food quality and perishability considerations. In the thesis, we investigate three FPRPs that capture some critical features of the FSC, such as packaging, customer satisfaction (dependent on food quality), and delivery time windows. The FPRP is generally more complex than the general PRP due to the additional complexity of integrating the food characteristics, thus the existing models and solution methods cannot be directly applied to solve it.

## 1.2 Content and contributions

Based on the context described in Section 1.1, this thesis focuses on developing new models and algorithms for a class of production routing problems. Firstly, to further investigate the classic PRP, a multi-product PRP with outsourcing (MPRPOS) is investigated. The MPRPOS extends the classic PRP by considering multiple products and outsourcing decisions. Then, based on the study of the classic PRP, considering food quality and perishability, a novel FPRP with multiple plants and packaging consideration (MFPRP) is studied. Thirdly, to investigate the tradeoff between cost and customer satisfaction, a bi-objective FPRP (BFPRP) is addressed. Finally, a FPRP with time windows (FPRPTW) is studied. The studied problems integrate important aspects that have not been addressed in the literature yet. They are presented in Table 1.1.

The main contributions of the thesis are summarized as follows.

1) A novel MPRPOS is studied. For the problem, a mixed integer linear program (MILP) model is formulated and a three-level heuristic (TLH) is developed. The heuristic is also able to solve the classic PRP. Numerical results on MPRPOS

Table 1.1: Summary of the studied problems

| Problems | Objective | Plant | Product | Food | Packaging | Outsourcing | TW | Solution method |
|---|---|---|---|---|---|---|---|---|
| MPRPOS | S | S | M | | | × | | TLH |
| MFPRP | S | M | S | × | × | | | HM |
| BFPRP | M | S | S | × | | | | $\epsilon$-CTIH |
| FPRPTW | S | S | S | × | | | × | CPLEX |

Note: S: single, M: multiple, TLH: three-level heuristic, HM: hybrid matheuristic
TW: time windows, $\epsilon$-CTIH: $\epsilon$-constraint-based two-phase iterative heuristic

instances show that TLH is efficient and outperforms the commercial solver CPLEX. Numerical results on PRP benchmark instances further confirm the good performance of TLH and show that it is very fast for large-sized instances. In particular, TLH finds 283 new best solutions for the benchmark instances.

2) A new MFPRP with packaging is addressed. For the problem, a MILP model is proposed. Then a hybrid matheuristic (HM) is developed to solve it. The heuristic has three components: 1) a two-phase iterative method to generate a good initial solution or useful information; 2) a fix-and-optimize procedure to repair and improve the solution; and 3) a route-based optimization to further improve the solution. Especially, the route-based optimization exploits the large number of routes generated by the two-phase iterative and the fix-and-optimize components. Numerical results demonstrate that HM is efficient and outperforms the commercial solver CPLEX.

3) A new BFPRP is investigated. The two objectives are to minimize the total cost and to maximize the average quality of food products received by customers. For the problem, a novel bi-objective MILP model is proposed. Then an $\epsilon$-constraint-based two-phase iterative heuristic ($\epsilon$-CTIH) that combines an $\epsilon$-constraint framework and a two-phase iterative heuristic is developed to generate a near-optimal Pareto solution set. The original bi-objective problem is transformed into a series of single-objective problems that are solved iteratively by the two-phase iterative heuristic. In particular, compared with the classic $\epsilon$-constraint method in which each transformed single-objective problem is only linked by $\epsilon$, the proposed $\epsilon$-CTIH utilizes useful information from the previous iteration to solve the transformed single-objective problem in the current iteration. The fuzzy logic decision method is applied to help decision makers select a preferred solution based on their preferences. Computational results on a case study indicate that the proposed model and solution method are able to solve a

real world case. In particular, the proposed approach can reduce the total cost by 10.77% compared with a three-phase constructive heuristic. Computational results on 185 randomly generated instances show that the proposed approach can provide better Pareto solution set with shorter computation time compared with CPLEX.

4) A FPRPTW is studied. This is still an ongoing work. For the problem, a MILP model is first presented. Then the model is directly solved by CPLEX. Experimental results on 45 randomly generated instances show that the problem is quite complex, and only small-sized instances can be solved optimally by CPLEX. Thus in future study, efficient algorithms should be developed to solve large-sized instances for the problem.

The remainder of this thesis is organized as follows.

Chapter 2 presents a systematic literature review. Firstly, existing studies on the PRP is reviewed. Then existing works on the integrated FSC planning is presented. Finally, we review solution methods for the PRP, followed by the introduction to bi-objective optimization.

Chapter 3 studies the MPRPOS that extends the classic PRP. For the problem, a new MILP model is presented and a three-level heuristic is developed. Computational experiments on 225 randomly generated MPRPOS instances and 1530 PRP benchmark instances are conducted to evaluate the performance of the proposed heuristic.

Chapter 4 addresses the MFPRP with packaging considerations. The problem is first formulated as a MILP model. Then hybrid matheuristic is presented to solve it. The performance of HM is evaluated by testing 320 randomly generated instances.

Chapter 5 investigates the BFPRP, in which the cost and food quality are simultaneously set as objectives. Based on the constructed bi-objective MILP, a new $\varepsilon$-constraint-based iterative heuristic and a fuzzy logic decision method are developed. Computational experiments on a case study and randomly 185 generated instances are conducted to evaluate the proposed methods.

Chapter 6 studies the FPRPTW. A MILP model is formulated and several valid inequalities are proposed to strengthen the model. Then the proposed model is directly solved by CPLEX.

Chapter 7 concludes this thesis and discusses perspectives for future research.

# Chapter 2

# Literature review

In this chapter, we first review recent developments on the PRP in Section 2.1. Next, we briefly review recent studies on integrated planning problems considering food quality and perishability in Section 2.2. Then, the solution methods for the PRP are reviewed in Section 2.3. We then introduce bi-objective optimization and its solution methods in Section 2.4, followed by the introduction to fuzzy logic-based decision for select a preferred solution among a set of alternative solutions Section 2.5. Finally, we conclude this chapter in Section 2.6.

## 2.1 The production routing problem

The classic PRP can be defined on a SC network where a single production plant produces and distributes products to spatially distributed customers in order to meet their time varying demand. The plant usually has a limited production capacity, a limited inventory capacity, and a fleet of capacitated vehicles. Each customer has a deterministic dynamic demand and a limited inventory capacity. A solution to the PRP corresponds to an integrated plan in which the production, inventory, distribution and routing decisions are jointly optimized to minimize the total system cost including the fixed and variable production cost at the plant, the inventory cost at the plant and customers, and the transportation cost for distributing the products to customers. The decisions to be made in each period are: 1) production quantity in the plant; 2) inventory held in the plant and each customer; 3) delivery quantity to each customer; and 4) vehicle routing for each planned delivery. In the literature, the studies on PRP can be characterized according to the following features: 1) single- or multi-plant; 2) single- or multi-product; 3) unlimited or limited production and/or inventory capacity at the plant; and 4) rich production or routing considerations. The basic version of the PRP (denoted as classic PRP) involves a single plant, a single product, multiple (limited) homogeneous vehicles, and multiple customers. Since the PRP was introduced by Chandra [31] in 1993, the classic PRP

with a single plant, a single product, and a limited production and inventory capacity has been most widely investigated with focusing on developing efficient algorithms [1], [3], [4], [18]–[20], [25]–[27], [85], [88], [92]. The single-plant multi-product PRP has been studied by [17], [28], [31], [32], [49], [78], [84]. Zhang *et al.* [102] address a multi-plant multi-product PRP with two time grids and different production modes. Most studies consider production and inventory capacity except a few works that assume unlimited production and inventory capacity [2], [15], [31], [49], [87]. Recently, the PRP is extensively studied by integrating important issues in supply chain management. Examples include lost sales [5], [82], backordering [28], greenhouse gas emission [82], startup cost [84], demand uncertainty [5], [7], [8], rich routing [64], [77], [78], reverse logistics and remanufacturing [81].

Existing studies on the PRP generally assume that the customer demand should always be met with in-house production. However, due to various reasons, e.g., limited capacities, emergency events and demand uncertainty, customer demand may frequently not be met in time with in-house production. In this case, a few studies deal with this issue by allowing lost sales or backordering [28], [82]. However, timely and reliable delivery of products is an important measure of company performance [62], [63]. Outsourcing consists of satisfying customer demand in time with out-house procurement. It is defined as the act of obtaining semi-finished products, finished products or services from an outside company to satisfy customer demand in time. In this way, customers can receive their demand without delay. Chu and Chu [36] and Gilley and Rasheed [52] conclude that outsourcing has many potential benefits to companies, such as improved financial performance, quick response to changes and better core competency. Outsourcing has been widely studied in SC planning applications. Lee *et al.* [63] investigate advanced planning and scheduling with considering outsourcing. A single-item capacitated dynamic lot-sizing problem (LSP) with backlogging and outsourcing is studied by [37]. Polynomial algorithms for single-item LSP with bounded inventory and backlogging or outsourcing are developed by [35], [36]. Lee and Lan [62] study an extended economic production quantity model under stochastic demand with a secondary facility. Haoues *et al.* [56] address a two-echelon SC network considering multi-outsourcers. Integrating outsourcing decisions in the PRP enables a company to further reduce system cost and enhance service level. We observe that the PRP has been widely studied with considering new aspects that are important in SC planning. However, there are still many aspects awaiting to be investigated and the multi-product PRP has not received enough attention. To the best of our knowledge, the multi-product PRP with outsourcing has not been studied yet.

## 2.2 The food production routing problem

Food quality and perishability play a vital role in the planning of the production, inventory and routing activities of a FSC. Existing integrated planning problems that integrate food quality and perishability can be generally classified as the production scheduling and vehicle routing problem (PSVRP), PDP, and IRP. The PSVRP jointly optimizes production scheduling and vehicle routing decisions, but no inventory is considered. To minimize the total production, inventory, and direct distribution costs, the PDP simultaneously considers production lot-sizing with direct shipments to customers, excluding vehicle routing. The IRP solution consists of jointly determining the inventory and vehicle routing decisions to minimize the total inventory and routing cost, while the production quantity in each period is assumed to be given. In the following, we review studies on the PSVRP, PDP, and IRP that consider food perishability and quality. And then introduce the FPRP.

For the PSVRP with perishable products, Geismar *et al.* [50] develop a two-phase heuristic that uses evolutionary algorithms for PSVRP. Chen *et al.* [33] formulate a nonlinear model to maximize the total profit and develop a decomposition-based algorithm. Belo-Filho *et al.* [21] propose an adaptive large neighborhood search framework that relies on MILP and tools. Amorim *et al.* [11] study a PSVRP with time windows. Devapriya *et al.* [44] propose an evolutionary algorithm-based heuristic for a perishable product. Lacomme *et al.* [60] design a greedy randomized adaptive search procedure with an evolutionary local search algorithm.

In terms of the PDP considering perishability, Ahuja *et al.* [9] develop a greedy heuristic and an efficient implementation of the very-large-scale-neighborhood-search method. Amorim *et al.* [12] study a bi-objective PDP to minimize the total cost and to maximize the freshness. Rong *et al.* [86] study a PDP in which the food quality is traced throughout the production, inventory and distribution processes. They also explicitly consider temperature control in the integrated planning model.

For the IRP dealing with perishable products, Le *et al.* [61] develop a MILP model and propose a column generation-based solution approach. Mizaei and Seifi [79] propose a hybrid metaheuristic with lost sale consideration. Soysal *et al.* [93] investigate an IRP with truckload-dependent distribution cost, service level consideration and demand uncertainty. Shaabani and Kamalabadi [90] present an IRP involving perishable products with a fixed shelf life and develop a population-based simulated annealing algorithm. Crama *et al.* [41] investigate an IRP with demand uncertainty. Coelho and Laporte [39] study an IRP with age-based product price to maximize the total revenue. They formulate the problem with an age index to explicitly trace the food quality. In addition, the final price of a perishable product is assumed to be

dependent on its age. Such formulation can trace the food quality throughout the planning process and capture the effects of product discount policies.

The FPRP, as a natural extension of the PDP and IRP, aims to provide an integrated food production, inventory and routing plan in which food quality and perishability are considered. Such integrated planning will save costs, reduce food wastes and losses, and improve service level. The FPRP is generally more complex than the classic PRP due to the additional complexity introduced by integrating food quality and perishability. Therefore, it is of significant value to study the FPRP. However, the FPRP has rarely been studied in the literature. Recently in 2018, Qiu *et al.* [83] study a single-plant PRP considering products with perishable inventory, but they did not specify the products as food products. Food products are relevant to people's health and life quality, therefore the food quality and perishability should be carefully handled in the FPRP.

Innovative preservation methods can reduce the deterioration rate of perishable food products and in turn affect the production, inventory and routing decisions. However, there is an implicit tradeoff between the cost and benefit brought by adopting a new preservation method, e.g., a SC planner faces the choice between more expensive preservation methods that can extend shelf life of their products and less expensive preservation methods that cannot. Recently, SC planning problems that integrate preservation methods have been receiving increasing attention. Hsu *et al.* [57] present a solution procedure to determine an optimal replenishment cycle, shortage period, order quantity, and preservation method to maximize the total profit. The effect of preservation technology investment on inventory decisions are studied by [45] and [75]. Zhang *et al.* [101] investigate a planning problem in which the pricing, replenishment cycle and preservation technology investment are jointly determined to maximize the total profit per unit time. Yang *et al.* [100] examine an optimal dynamic decision-making problem in which the optimal trade credit, preservation technology investment and replenishment strategies are simultaneously determined to maximize the total profit. Most recently, Li *et al.* [66] study a joint pricing, replenishment and preservation technology investment problem for non-instantaneous deteriorating items. Li *et al.* [67] investigate the coordination of inventory and packaging decisions in a retailing environment. They show that the choice of packaging methods depends on the customers' willingness to buy less fresh items as substitutes, and indicate that the adoption of better packaging can consistently reduce food waste. Thus, integrating food preservation decisions in the FPRP seems to be promising. The FPRP with packaging consideration has not been studied in the literature yet.

Nowadays, customers are more critical on food quality than ever before. As food quality decreases continuously, it may not meet customers' expectation when the food

reaches them. In this case, customer satisfaction is low, which in turn affects the company's goodwill and revenue. Therefore, in an integrated FSC planning, a company should not only focus on reducing cost, but also consider food quality that is quite related to customer satisfaction. Recently, companies have realized that customer satisfaction is a key to success in today's business environment under fierce competition. Therefore, in a FPRP, decision makers should not only aim to reduce cost, but also provide customers with high-quality food. Simultaneously optimizing these two objectives needs to solve a bi-objective optimization problem. Similar bi-objective problems have been studied by Amorim *et al.* [12] who study a bi-objective PDP that minimizes the total production and distribution cost and maximizes the food freshness. Amorim and Almada-Lobo [10] investigate a bi-objective vehicle routing problem with time windows (VRPTW) in which the total routing cost and the product freshness are simultaneously optimized. However, to the best of our knowledge, a bi-objective FPRP that simultaneously optimizes the total cost and food quality has not been investigated yet.

## 2.3 Solution methods for the production routing problem

In this subsection, we review existing solution methods for the PRP. They can be generally classified as exact method, constructive heuristic, branch-and-price (B&P) approach, Lagrangian-relaxation or benders decomposition methods, meta-heuristic, and mathematical-programming-based heuristic (matheuristic).

### 2.3.1 Mixed-integer linear programming method

### 2.3.2 Exact method

An exact solution method has the advantage in proving optimality of solutions whereas it is extremely time-consuming, thus it can only solve small-sized instances. Examples include dynamic programming (DP), branch-and-bound (B&B), and branch-and-cut (B&C). A few studies propose B&C algorithms to exactly solve the PRP. Ruokokoski *et al.* [87] propose a B&C for the PRP with a single uncapacitated vehicle and an uncapacitated plant. They compare several formulations of the PRP. Results show that instances with 40 customers and 15 periods or with 80 customers and 8 periods are solved to optimality within two hours. Archetti *et al.* [15] develop a B&C for the PRP with a single capacitated vehicle. The subtour elimination constraints are first relaxed and added only when they are violated during the branching process. The algorithm is evaluated on instances with 14 customers and 6 periods,

and results show that the heuristic can solve most of these instances to optimality in a few seconds. Adulyasak *et al.* [3] develop a B&C to solve the multi-vehicle PRP. They compare two formulations with and without vehicle index, and several valid inequalities are proposed to strengthen the two formulations. Results indicate that instances with 35 customers, 3 periods and 3 vehicles can be solved to optimality. Qiu *et al.* [84] develop a B&C to address the MPRP with startup cost. Their B&C uses row generation to deal with subtour elimination constraints. The proposed B&C can solve instances with up to 70 customers, 3 periods, 4 vehicles, and 3 products. A B&C based on logical, strengthened lot-sizing, and lifted MTZ-type (Miller-Tucker-Zemlin) valid inequalities is presented by [83], in which instances with up to 50 customers, 3 periods, and 4 vehicles are solved to optimality.

### 2.3.3 Constructive heuristic

A constructive heuristic is often designed based on the problem characteristics. It can quickly provide a feasible solution for a NP-hard optimization problem, but the solution is not guaranteed to be optimal. Chandra [31] proposes a three-phase constructive heuristic that consists of sequentially solving a warehouse replenishment problem, a resulting distribution problem, and a consolidation process which allows the potential move of the delivery date to a customer from one period to another. Their computational results show that the consolidation process can lead to cost reduction. Chandra and Fisher [32] design a similar three-phase heuristic. In the heuristic, a production planning problem is first solved to determine a production schedule that minimizes the total cost of setups and inventory. Then a distribution problem is solved to schedule a fleet of vehicles. Finally, a local improvement procedure that changes the production schedule and the delivery schedule to customers is applied. They show that the reduction of the total operating cost ranges from 3% to 20% accounting to coordination. Lei *et al.* [64] develop a two-phase heuristic in which the first phase solves a PDP and the second phase solves many VRPs. Boudia *et al.* [26] present an uncoupled heuristic and a coupled heuristic. The uncoupled one determines the production plan and the distribution plan sequentially.The coupled one applies local search to improve the solution by performing two kinds of moves, i.e., exchanges of customers and modification of the production schedule, delivery schedule, and routes.

### 2.3.4 Branch and price approach

A B&P approach combines the column generation and the branch and bound methods. Bard and Nananukul [18] propose a B&P in which a delivery schedule for one

day is defined as a column to form the restricted master problem. Then promising columns are generated and added to the restricted master problem iteratively. Then a branch and bound is used to obtain an integer solution when no column can be generated any more. Bard and Nananukul [20] then design a hybrid methodology that combines exact and heuristic procedures within a B&P framework. The method uses a new branching strategy to deal with the master problem degeneracy and obtains a feasible solution by combining a rounding heuristic and tabu search within a B&P. Qiu *et al.* [82] present a B&P heuristic that integrates a column-generation formulation based on Dantzig-Wolfe decomposition.

### 2.3.5 Lagrangian relaxation or benders decomposition methods

The Lagrangian relaxation method transforms the original problems into easier ones by dualizing the side constraints [48]. Fumero and Vercellis [49] develop a Lagrangian-relaxation-based heuristic that transforms the original MPRP into four subproblems, i.e., production, inventory, distribution, and routing subproblems. The Benders decomposition method is designed to tackle problems with complicating variables which, when temporarily fixed, yield a problem that is generally easier to solve than the original problem. The original problem is decomposed into a master problem and a subproblem, which are solved iteratively by generating and adding feasibility or optimality cuts. Adulyasak *et al.* [5] propose a Benders decomposition-based B&C to solve the PRP with demand uncertainty.

### 2.3.6 Metaheuristic

Metaheuristic is the most preferred method for solving the PRP since it can exploit solution space by a guided search procedure with accumulated search experience to avoid getting trapped into local optimum. Metaheuristics can find near-optimal solutions and can be easily adapted to solve similar problems. However, the quality of the obtained solutions often needs to be evaluated by other techniques. Many metaheuristics are developed to solve the PRP. A greedy randomized adaptive search procedure and two improved versions with either a reactive mechanism or a path relinking process are proposed by [25]. Boudia and Prins [27] propose a memetic algorithm in which the genetic algorithm is modified by a local search procedure to improve both its initial population and offsrpings. Their memetic algorithm starts with an initial solution constructed by a simple heuristic. Then a crossover is used to generate new offsprings. Finally, local search is performed to change both the production plan and the distribution plan. Bard and Nananukul [19] develop a reactive

tabu search heuristic. The heuristic starts from a solution generated by a two-phase method that solves a PDP with an approximate routing cost and a series of vehicle routing problem (VRP). Then the reactive tabu search method exploits neighborhoods defined by a swap procedure and a transfer procedure. The swap procedure involves a exchange of delivery quantities between two customers, and the transfer procedure tries to reassign the delivery quantity of a customer from one period to another. Armentano *et al.* [17] propose two tabu search heuristics, namely the basic tabu search (TS) and the tabu search with path relinking (TSPR), to solve the multi-product PRP and the classic PRP. The TS starts with the construction of an initial solution that may be infeasible, then searches for improvements by performing three move components, i.e., transference of the maximum quantity from one period to another, insertion of the quantity in a route, and determination of a new production plan. Then the TSPR makes use of path relinking that links every tabu search local optimum with the farthest solution of a pool of elite solutions. Adulyasak *et al.* [4] present an adaptive large neighborhood search heuristic. Firstly, initial solutions are generated by solving a PDP and many VRPs with applying the local branching inequalities to diversify the production setup schedule. Then the adaptive large neighborhood search is performed using the proposed selection and transformation operators. Each time a solution is changed during the transformation process, a minimum cost flow subproblem is solved to obtain the values of all continuous variables. Finally, the incumbent solution is further improved by solving a TSP for each individual route. Recently, Qiu *et al.* [85] develop a variable neighborhood search heuristic in which delivery and routing variables are handled by neighborhood search and binary variables for production setup and continuous variables are determined by solving a MILP.

### 2.3.7 Mathematical programming-based heuristic

Mathematical programming-based heuristics (Matheuristics), which enable standard MILP solvers to be applied to find optimal or near-optimal solutions, have shown good performance and flexibility to solve PRPs. Relaxation and decomposition techniques are often combined to transform the original problem into subproblems that are easier to solve, while trying to lose as less as possible information of the original problem. Brahimi and Tarik [28] develop a hybrid heuristic that combines a relax-and-fix and a local search. Firstly, a production-distribution problem is solved with relaxing the integrality of the routing variables of the original model. Then a series of MILP models are solved by fixing part of the integer variables, relaxing the integrality of partial integer variables, and optimizing the remaining integer variables and all continuous variables. Similar relax-and-fix method has been proposed by [78] to

solve the multi-product PRP. Zhang *et al.* [102] propose an iterative MILP-based heuristic in which a MILP model is solved iteratively with a restricted set of updated candidate routes. Absi *et al.* [1] propose a two-phase iterative heuristic in which the original PRP is decomposed into an PDP and a resulted routing problem. In the first phase, the PDP, in which an approximate visit cost is introduced, is solved to determine the production, inventory and delivery quantity. The second phase solves a series of VRPs or traveling salesman problems (TSPs) to determine the routes. Then the approximate visit cost is updated. This procedure is repeated until a stopping criterion is met. Then, Chitsaz *et al.* [34] present a three-phase iterative matheuristic similar to that of [1]. The first phase decides the production setup schedule with an approximate total visit cost. With the production setup schedule being fixed, the second phase determines the production, inventory and delivery quantity with an approximate visit cost related to each customer and period. Then the third phase solves a VRP for each period and updates the approximate visit cost. This process is repeated until the stopping criterion is met. Solyali süral [92] develop a five-phase heuristic for the PRP. The first phase establishes customer visiting sequence by solving a TSP. In the second phase, a restricted PRP is solved with a fixed customer visiting sequence established in the first phase. The third phase solves a VRP for each period. Infeasible solutions obtained in the third phase are repaired in the fourth phase. Finally, incumbent feasible solutions are further improved in the fifth phase by solving many TSPs. Russell [88] designs a set-partitioning approach and a multi-phase approach which share some similarities with [1]. He claims that the main differences are the artificial vehicle routing cost incorporation, predetermined route generation, and seed route approximate formulation.

## 2.4 Bi-objective combinatorial optimization

In SC planning applications, the most commonly used objective is to minimize total cost or to maximize total profit. However, decision makers often face the situation in which multiple objectives are pursued. Usually, these objectives are conflict and hard to be achieved simultaneously. In this context, multi-objective optimization shows its advantage since it has the ability to deal with complex objectives. To help decision makers find a balance among the multiple targets, the corresponding problems should be formulated and solved by the multi-objective optimization models and methods. In this thesis, Chapter 5 investigates a bi-objective food production routing problem (BFPRP) that simultaneously optimizes two objectives. Therefore, in the following subsections, we first state the principles of bi-objective optimization, which is the most common form of the multi-objective optimization. Then the solution

methods for multi-objective optimization are briefly reviewed. Next, the $\epsilon$-constraint method is presented. Finally, the evaluation metrics for bi-objective optimization is introduced.

### 2.4.1 Principles

A general bi-objective optimization problem can be formulated as follows (model $\mathcal{M}$):

$$\min f_1 = \varphi(\boldsymbol{x}) \tag{2.1}$$

$$\min f_2 = \omega(\boldsymbol{x}) \tag{2.2}$$

s.t.

$$\boldsymbol{x} \in \mathcal{X} \tag{2.3}$$

where $f_1$ and $f_2$ represent the two objectives that may often be conflict, i.e., there may not exist a solution that will lead to the minimum of $f_1$ and $f_2$ simultaneously; $\boldsymbol{x}$ represents a vector of all decision variables; $\varphi(\boldsymbol{x})$ and $\omega(\boldsymbol{x})$ correspond to the two objective functions, respectively. $\mathcal{X}$ is the feasible region of $\boldsymbol{x}$. A feasible solution $\boldsymbol{x} \in \mathcal{X}$ is said to cover another feasible solution $\boldsymbol{x}' \in \mathcal{X}$ if $\varphi(\boldsymbol{x}) \leq \varphi(\boldsymbol{x}')$ and $\omega(\boldsymbol{x}) \leq \omega(\boldsymbol{x}')$ , and $\boldsymbol{x}$ is said to dominate $\boldsymbol{x}'$ ( $\boldsymbol{x} \prec \boldsymbol{x}'$) if and only if $\boldsymbol{x}$ covers $\boldsymbol{x}'$ and at least one of the two inequalities is strict. Similarly, $\boldsymbol{x} \in \mathcal{X}$ is said to be non-dominated if there is no feasible solution $\boldsymbol{x}' \in \mathcal{X}$ such that $\boldsymbol{x}' \prec \boldsymbol{x}$. If a feasible solution is non-dominated, then we say that it is Pareto-optimal. Then its corresponding objective values $(\varphi(\boldsymbol{x}), \omega(\boldsymbol{x}))$ form a Pareto point. The Pareto-optimal solution set is defined as $P_s = \{\boldsymbol{x} \in \mathcal{X} | \boldsymbol{x}$ is Pareto-optimal$\}$, and the Pareto front is defined as $P_f = \{(\varphi(\boldsymbol{x}), \omega(\boldsymbol{x})) | \boldsymbol{x} \in P_s\}$.

### 2.4.2 Solution methods

Generally, methods for solving multi-objective optimization problems can be classified into preference-based methods and generating methods [42], [43]. The former; e.g., goal programming, goal-attainment, and global criterion methods, take into consideration preferences of a decision maker during a solution process. They provide a single solution for a decision maker, which is not flexible and lacks information about alternative solutions. In contrast, generating methods, e.g., weighted sum, $\epsilon$-constraint, and evolutionary methods [40], aim to generate a set of Pareto optimal points without any preference from a decision maker. These methods have the advantages to provide a set of solutions that can be selected flexibly by a decision maker and have been adopted in many real-world applications [95], [96]. Among these methods, weighted sum methods have several shortcomings. For instance, many combinations of weights

may result in the same solution, objective functions have to be scaled to a common scale before forming the weighted sum, and it is hard to control the expected number of Pareto solutions. Evolutionary methods [59] provide a set of approximate Pareto solutions that may be very far from the optimal Pareto solutions. The $\epsilon$-constraint method is well-known as one of the most effective approaches, especially for bi-objective optimization problems [23]. It transforms an original multi-objective problem into a set of mono-objective problems that can be solved to obtain a set of Pareto solutions. It is superior to the weighted sum method because it is able to produce non-extreme efficient solutions, does not require scaling, and needs less effort to obtain an expected number of Pareto solutions [76]. Therefore, the $\epsilon$-constraint method is used to solve the studied BFPRP in Chapter 5. In the following, we briefly introduce the principles of the $\epsilon$-constraint method.

#### 2.4.2.1 $\epsilon$-constraint method

The basic idea of an $\epsilon$-constraint method is to transform an initial bi-objective problem (say problem defined by (2.1)-(2.3)) into a sequence of mono-objective problems with one principal objective (say minimize $\varphi(\boldsymbol{x})$) by transforming the other objective (say minimize $\omega(\boldsymbol{x})$) as a constraint bounded by a parameter $\epsilon$: $\omega(\boldsymbol{x}) \leq \epsilon$. For a given value of $\epsilon$, the ($\epsilon$-parameterized) mono-objective problem can be written as:

$$\min \{\varphi(\boldsymbol{x}) | \omega(\boldsymbol{x}) \leq \epsilon, \boldsymbol{x} \in \mathcal{X}\} \tag{2.4}$$

after solving this problem, we obtain a solution $\boldsymbol{x}^*(\epsilon) \in \mathcal{X}$, $f_1(\epsilon) = \varphi(\boldsymbol{x}^*(\epsilon))$ and $f_2(\epsilon) = \omega(\boldsymbol{x}^*(\epsilon)) \leq \epsilon$. It can easily be proved that for any $\boldsymbol{x} \in P_s$, there is an $\epsilon$ such that $\boldsymbol{x} = \boldsymbol{x}^*(\epsilon)$. As a consequence, by considering all possible values of $\epsilon$ and solving the corresponding mono-objective problems (2.4), we can generate the set of all Pareto solutions $P_s$.

For model $\mathcal{M}$, we transform the second objective into a constraint, thus yielding the following mono-objective model $\mathcal{M}(\epsilon)$:

$$\min f_1 = \varphi(\boldsymbol{x}) \tag{2.5}$$

$$s.t.$$

$$\omega(\boldsymbol{x}) \leq \epsilon \tag{2.6}$$

$$x \in \mathcal{X} \tag{2.7}$$

In order to construct the set of Pareto-optimal solution, we need to know the set of all possible values of $\epsilon$, which is actually an interval. This interval can be determined by obtaining an ideal point $(f_1^I, f_2^I)$ and a nadir point $(f_1^N, f_2^N)$. The ideal and nadir points define lower and upper bounds on the objective values of Pareto-optimal

solutions, respectively [23]. They can be obtained by exactly solving the following mono-objective problems:

$$f_1^I = \min\{\varphi(\boldsymbol{x})|\boldsymbol{x} \in \mathcal{X}\} \tag{2.8}$$

$$f_2^I = \min\{\omega(\boldsymbol{x})|\boldsymbol{x} \in \mathcal{X}\} \tag{2.9}$$

$$f_1^N = \min\{\varphi(\boldsymbol{x})|\omega(\boldsymbol{x}) = f_2^I, \boldsymbol{x} \in \mathcal{X}\} \tag{2.10}$$

$$f_2^N = \min\{\omega(\boldsymbol{x})|\varphi(\boldsymbol{x}) = f_1^I, \boldsymbol{x} \in \mathcal{X}\} \tag{2.11}$$

the value of $\epsilon$ can be bounded by interval $[f_2^I, f_2^N]$. Then a step size $\Delta$ should be fixed to explore values of $\epsilon$ and form a series of mono-objective problems that are solved to obtain the Pareto front or its approximation. The solution to each mono-objective $\mathcal{M}(\epsilon)$ for a given value of parameter $\epsilon$, if not dominated, is a Pareto-optimal solution to the original problem. The objective values of all Pareto solutions $(\varphi(\boldsymbol{x}), \omega(\boldsymbol{x}))$ form a Pareto front.

Ideally, we expect to generate the exact Pareto front. This may require solving a large number of mono-objective problems $\mathcal{M}(\epsilon)$ that may be very complex and time consuming, which is impractical and unnecessary for decision makers. In practice, decision makers may expect some representative Pareto solutions within a reasonable amount of computation time. The obtained representative Pareto solutions form an approximate Pareto front $\mathcal{A}_F$ in which no Pareto point is not dominated by any other Pareto point in $\mathcal{A}_F$.

### 2.4.3 Performance evaluation

In single-objective optimization, the solution quality can be directly evaluated by comparing the obtained solution with the lower or upper bounds. While in bi-objective optimization, the solution quality evaluation of the approximate Pareto solution set $\mathcal{A}_F$ is less straightforward and is often evaluated based on a reference Pareto solution set denoted as $\mathcal{R}_F$. Generally, the quality of $\mathcal{A}_F$ can be evaluated from four aspects [80], namely the cardinality $|\mathcal{A}_F|$ and $|\mathcal{R}_F|$, hypervolume ratio $\mathcal{H}$, average e-dominance $\mathcal{D}$, and computation time $\mathcal{T}$. The cardinality, hypervolume, and e-dominance are stated as following:

Cardinality is the number of solutions in an obtained Pareto solution set. If $|\mathcal{A}_F| > |\mathcal{R}_F|$, we say $\mathcal{A}_F$ is better than $|\mathcal{R}_F|$ from the cardinality perspective.

Hypervolume ratio $\mathcal{H}$, which can be calculated by (2.12) is the ratio of hypervolumes of $\mathcal{A}_F$ and $\mathcal{R}_F$ with the denotation of $\mathcal{H}_\mathcal{A}$ and $\mathcal{H}_\mathcal{R}$, respectively. $\mathcal{H}_\mathcal{A}$ indicates the objective space covered by set $\mathcal{A}_F$. As shown in Fig. 2.1, each Pareto point in the set forms a rectangle shown by the shaded area with respect to a reference point (generally the Nadir Pareto point), and the hypervolume of the solution set is the

area of the union of all rectangles. The larger the hypervolume ratio, the better the algorithm. If $\mathcal{H} > 1$, then $\mathcal{A}_F$ is better than $\mathcal{R}_F$.

$$\mathcal{H} = \frac{\mathcal{H}_{\mathcal{A}}}{\mathcal{H}_{\mathcal{R}}} \tag{2.12}$$

Indicator e-dominance denotes the average distance between $\mathcal{A}_F$ and $\mathcal{R}_F$. An approximate Pareto point $(\varphi(\boldsymbol{x}), \omega(\boldsymbol{x})) \in \mathcal{A}_F$ is said to e-dominate $(\varphi(\boldsymbol{x}'), \omega(\boldsymbol{x}')) \in \mathcal{R}_F$ if $\varphi(\boldsymbol{x}) \leq e(\boldsymbol{x}')\varphi(\boldsymbol{x}')$ and $\omega(\boldsymbol{x}) \leq e(\boldsymbol{x}')\omega(\boldsymbol{x}')$, where the e-dominance indicator $e(x')$ for a given Pareto point $((\varphi(\boldsymbol{x}'), \omega(\boldsymbol{x}'))$ is calculated as:

$$e(x') = \min_{(\varphi(\boldsymbol{x}),\omega(\boldsymbol{x}))\in\mathcal{A}_F} max\{\frac{\varphi(\boldsymbol{x})}{\varphi(\boldsymbol{x}')}, \frac{\omega(\boldsymbol{x})}{\omega(\boldsymbol{x}')}\} \tag{2.13}$$

$e(x') < 1$ indicates that $(\varphi(\boldsymbol{x}'), \omega(\boldsymbol{x}'))$ is dominated by $(\varphi(\boldsymbol{x}), \omega(\boldsymbol{x}))$. The average e-dominance indicator is calculated as:

$$\mathcal{D} = \frac{1}{|\mathcal{R}_F|} \sum_{(\varphi(\boldsymbol{x}'),\omega(\boldsymbol{x}'))\in\mathcal{R}_F} e(x') \tag{2.14}$$

The closer $\mathcal{D}$ is to 1, the closer$\mathcal{A}_F$ is to $\mathcal{R}_F$.

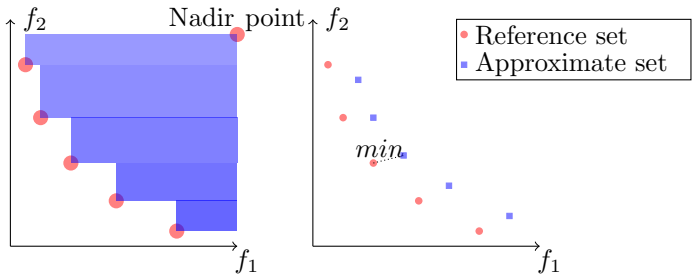



Fig. 2.1: Hypervolume indicator and e-dominance indicator

## 2.5 Selection of a preferred solution - fuzzy logic decision method

In a bi-objective decision making process, when a set $S$ of alternative solutions numbered from 1 to $|S|$ are obtained, decision makers may need to choose a best-compromised solution according to their preference. The fuzzy logic decision method can take into account their preference and indicate the optimality degree of the selected solution [46], [103]. For the $i^{th}$ objective ($i = 1, 2$ for a bi-objective problem) and the $s^{th}$ solution, where $s \in S$, a linear membership function $\eta_i(f_i^s)$ is defined for each of the two objectives as follows:

$$\eta_i(f_i^s) = \begin{cases} 1, & f_i^s \leq f_i^I \\ \frac{f_i^N - f_i^s}{f_i^N - f_i^I}, & f_i^I < f_i^s < f_i^N, i = 1, 2; 1 \leq s \leq S \\ 0, & f_i^s \geq f_i^N \end{cases} \tag{2.15}$$

where $f_i^I$ and $f_i^N$ denote the lower and upper limits of the $i^{th}$ objective function, respectively; $f_i^s$ is the $i^{th}$ objective value of the $s^{th}$ Pareto solution. The membership degree $\eta^s$ of the $s^{th}$ solution is calculated as follows:

$$\eta^s = \sum_{i=1}^{2} w_i \eta_i(f_i^s) \tag{2.16}$$

where $w_i$ denotes the weight of the $i^{th}$ objective with $\sum_{i=1}^{2} w_i = 1$. It can be determined based on decision makers' preference. The solution with the maximum value of $\eta^s$ is selected as their most preferred solution.

Note that this selection process presents some advantages over weighted sum method. Although both approach require modeling decision makers' preference, the meaning of weights is different. In a weighted sum method, a weight is directly attached to each objective, while these objectives are often of different units and scales. Modeling decision makers' preference among such objectives is challenging. Furthermore, if decision makers change their mind, a new set of weights has to be estimated to reflect their new preference and a new problem has to be solved again. The whole process thus may be very time consuming. In a fuzzy decision approach, however, a weight is attached to the membership degree of each objective and these membership degrees are already normalized to a scalar between 0 and 1 and the set of Pareto optimal solutions is already available at the selection stage. Such weights are easier to estimate in order to model decision makers' preference. More importantly, when decision makers change their mind, it is sufficient to estimate a new set of weights and select a new solution among the set of Pareto-optimal solutions. This makes the selection process much shorter.

## 2.6 Conclusions

In this chapter, we first review the studies on the PRP that is a hard combinatorial optimization problem since it generally contains two classic optimization problems, i.e., the LSP and VRP. It can be found that the PRP and its variants have been widely studied during the last two decades. However, the multi-product PRP has received less attention. Then we present the review on integrated FSC planning problems. We find that food perishability and quality have been extensively investigated in

the PSVRP, PDP, and IRP. However, to the best of our knowledge, the FPRP with perishability consideration has not been addressed yet. Especially, several critical issues such as packaging, customer satisfaction, and delivery time windows have not been investigated in the FPRP yet. Motivated by the above mentioned observations, this thesis is devoted to developing new models and efficient solution algorithms for the PRP and FPRP.

# Chapter 3

# Multi-product Production Routing Problem with Outsourcing

## 3.1 Introduction

Most studies on the PRP limit to a single type of product. However, in real applications, a company usually produces multiple products. In addition, due to limited capacities, outsourcing strategy has become an important practice to response quickly to market changes. The MPRP is generally less frequently addressed and the outsourcing aspect has not been integrated in a MPRP yet. Therefore, in this chapter, as a beginning work, we study a MPRP with Outsourcing (MPRPOS) that is a generalization of the classic PRP. The problem consists of a plant and a set of customers. The plant produces and distributes multiple types of products to these customers over a planning horizon. The customer demand must be satisfied on time by either in-house production or outsourcing.

This chapter focuses on developing an efficient heuristic to solve both the MPRPOS and the classic PRP. The MPRPOS is first formulated as a MILP, then a three-level mathematical-programming-based heuristic (TLH) is developed to solve it. Finally, extensive computational experiments on 225 randomly generated MPRPOS instances and 1530 classic PRP benchmark instances are conducted to evaluate the proposed model and the developed heuristic.

The rest of this chapter is structured as follows. Section 3.2 gives the problem description and mathematical formulation of MPRPOS. In Section 3.3, an efficient TLH is developed. Computational results are presented in Section 3.4. Section 3.5 concludes the chapter.

## 3.2 Problem description and formulation

The MPRPOS concerns a single plant that is responsible to produce and deliver multiple types of products to a set of retailers over a finite planning time horizon. It can be stated as follows:

Consider a complete digraph $G$=$\{N$,$A\}$ with a set of nodes $N$=$\{0, 1, \ldots, n\}$ and a set of arcs $A$=$\{(i, j) : i, j \in N, i \neq j\}$ . A plant that can produce a set of products $P = \{0, 1, \ldots, |P|\}$ with limited production capacity $C$ and storage capacity $U_0$ is located at node 0. A fleet of homogeneous vehicles $K$=$\{1, , 2, \ldots, |K|\}$ with capacity $V$ is available at the plant. A set of customers $R$=$\{1, 2, \ldots, n\}$ is dispersed on the graph. Each customer $i \in R$ has a limited storage capacity $U_i$ and a dynamic demand $d_i^{pt}$ for product $p \in P$ in period $t \in T$, where $T = \{0, 1, \ldots, |T|\}$ is a finite planning time horizon.

An MPRPOS consists of simultaneously determining production, storage, delivery and routing planning to satisfy customer demand with outsourcing consideration. The objective is to minimize the total production, inventory, routing and outsourcing costs. For each period of the planning horizon, several decisions have to be made jointly: 1) how much to produce; 2) how much to deliver to each retailer; 3) how much inventory to hold at the plant and each retailer; 4) how to arrange the vehicle routes for the planned deliveries; and 5) how much to outsource for each product and retailer. Note that once a product $p$ is outsourced, its production and transportation will be performed by another company with an unit outsourcing cost $e_p$. The MPRPOS is studied under the following assumptions: 1) each vehicle's route starts and ends at the plant; 2) each vehicle can perform at most one trip within each period; and 3) each customer can be visited at most once within each period, i.e., split delivery from the plant is not allowed. To formulate the problem, the following notation is defined:

Indices

$i, j$ index of node, $i, j \in N$;
$t$ index of period, $t \in T$;
$k$ index of vehicle, $k \in K$;
$p$ index of product, $p \in P$;

Parameters

$c_{ij}$ travel cost on arc $(i, j) \in A$;
$a_p$ unit production cost of product $p \in P$;
$e_p$ unit outsourcing cost of product $p \in P$;
$b_p$ production set up cost of product $p \in P$;
$C$ production capacity;
$d_i^{pt}$ demand for product $p \in P$ at customer $i \in R$ in period $t \in T$;

$U_i$ inventory capacity of $i \in N$;
$h_i^p$ unit inventory holding cost per period of product $p \in P$ at $i \in N$;
$I_i^{p0}$ initial inventory of product $p \in P$ held at $i \in N$;
$V$ vehicle capacity;

Decision variables

$\xi_{pt}$ production quantity of product $p$ in period $t$;
$z_i^{pt}$ outsourcing quantity of product $p$ for customer $i$ in period $t$;
$w_{pt}$ equal to 1 if product $p$ is produced in period $t$; otherwise 0;
$I_i^{pt}$ inventory of product $p$ held at customer $i$ at the end of period $t$;
$y_i^{pkt}$ quantity of product $p$ delivered to customer $i$ by vehicle $k$ in period $t$;
$v_i^{kt}$ equal to 1 if customer $i$ is visited by vehicle $k$ in period $t$; and 0 otherwise;
$x_{ij}^{kt}$ equal to 1 if arc $(i, j)$ is traversed by vehicle $k$ in period $t$; and 0 otherwise.

The considered problem can be formulated as the following MILP (model $\mathcal{P}$).

$$\mathcal{P}: \min \sum_{p\in P}\sum_{t\in T}(a_p\xi_{pt} + b_p w_{pt}) + \sum_{i\in R}\sum_{p\in P}\sum_{t\in T} e_p z_i^{pt} + \sum_{i\in N}\sum_{p\in P}\sum_{t\in T} h_i^p I_i^{pt} + \sum_{(i,j)\in A}\sum_{k\in K}\sum_{t\in T} c_{ij} x_{ij}^{kt} \tag{3.1}$$

s.t.

$$I_0^{pt} = I_0^{p,t-1} + \xi_{pt} - \sum_{i\in R}\sum_{k\in K} y_i^{pkt},\ \forall p \in P, t \in T \tag{3.2}$$

$$I_i^{pt} = I_i^{p,t-1} + z_i^{pt} + \sum_{k\in K} y_i^{pkt} - d_i^{pt},\ \forall i \in R, p \in P, t \in T \tag{3.3}$$

$$\sum_{p\in P} \xi_{pt} \le C,\ \forall t \in T \tag{3.4}$$

$$\xi_{pt} \le C w_{pt},\ \forall p \in P, t \in T \tag{3.5}$$

$$\sum_{p\in P} I_i^{pt} \le U_i,\ \forall i \in N, t \in T \tag{3.6}$$

$$\sum_{i\in R}\sum_{p\in P} y_i^{pkt} \le V,\ \forall k \in K, t \in T \tag{3.7}$$

$$\sum_{p\in P} y_i^{pkt} \le V v_i^{kt},\ \forall i \in R, k \in K, t \in T \tag{3.8}$$

$$\sum_{k\in K} v_i^{kt} \le 1,\ \forall i \in R, t \in T \tag{3.9}$$

$$\sum_{j\in N\setminus\{i\}} x_{ij}^{kt} = \sum_{j\in N\setminus\{i\}} x_{ji}^{kt},\ \forall i \in N, k \in K, t \in T \tag{3.10}$$

$$\sum_{j\in N\setminus\{i\}} x_{ij}^{kt} = v_i^{kt},\ \forall i \in N, k \in K, t \in T \tag{3.11}$$

$$\sum_{i \in R} x_{0i}^{kt} \leq 1, \ \forall k \in K, t \in T \tag{3.12}$$

$$\sum_{i \in S} \sum_{j \in S \setminus \{i\}} x_{ij}^{kt} \leq |S| - 1, \ \forall S \subseteq R, |S| \geq 2, k \in K, t \in T \tag{3.13}$$

$$\xi_{pt} \geq 0, \ \forall p \in P, t \in T \tag{3.14}$$

$$z_i^{pt} \geq 0, \ \forall i \in R, p \in P, t \in T \tag{3.15}$$

$$I_i^{pt} \geq 0, \ \forall i \in N, p \in P, t \in T \tag{3.16}$$

$$y_i^{pkt} \geq 0, \ \forall i \in R, p \in P, k \in K, t \in T \tag{3.17}$$

$$w_{pt} \in \{0, 1\}, \ \forall p \in P, t \in T \tag{3.18}$$

$$v_i^{kt} \in \{0, 1\}, \ \forall i \in R, k \in K, t \in T \tag{3.19}$$

$$x_{ij}^{kt} \in \{0, 1\}, \ \forall (i, j) \in A, k \in K, t \in T \tag{3.20}$$

Objective function (3.1) minimizes the total production, outsourcing, inventory and routing cost. Constraints (3.2) and (3.3) indicate the inventory flow balance at the plant and customers, respectively. Constraints (3.4) mean that the total production quantity of each period cannot exceed the production capacity. Constraints (3.5) indicate that the production quantity of a product must be 0 if its production is not launched. Constraints (3.6) limit the maximum inventory level to the capacity at the plant and customers. The vehicle capacity constraints are imposed by (3.7). Constraints (3.8) allow positive delivery quantity to a customer only if it is visited. Constraints (3.9) forbid split delivery from the plant. Constraints (3.10) correspond to the vehicle flow conservation. Constraints (3.11) link the arc routing variable to the node visit variable. Constraints (3.12) indicate that one vehicle can perform at most one trip in each period. Constraints (3.13) are the subtour elimination constraint. Constraints (3.14)-(3.20) provide the ranges of the decision variables. The proposed MPRPOS is NP-hard since it contains a VRP that is well known to be NP-hard [51]. Due to the complexity of the studied problem, a new efficient three-level heuristic (TLH) is developed in the next section.

## 3.3 Solution method

In this section, we present an efficient TLH to obtain near-optimal solutions to the proposed model $\mathcal{P}$, which can be easily adapted to solve the classic PRP that assumes the outsourcing quantity to be 0. The principle of TLH is presented as follows: 1) firstly, the original model $\mathcal{P}$ is decomposed into a PDP and a series of VRP($t$) for each period $t$. Then a two-phase iterative method (TIM) solves the above-mentioned two subproblems iteratively to obtain an initial solution that may be infeasible to $\mathcal{P}$; 2) in the second level, a repairing strategy is designed to find a feasible solution. In

the repairing strategy, a restricted PDP (RPDP) and a series of traveling salesman problem (TSP)$(t, k)$, one for each vehicle $k$ and each period $t$, are solved sequentially. In case that the obtained solution from Level 1 is feasible, Level 2 skips the RPDP step and directly solves a series of TSP$(t, k)$ to consolidate each route; and 3) In level 3, a fix-and-optimize procedure is applied iteratively to each customer $s \in R$ to further enhance the incumbent solution. Once the fix-and-optimize procedure stops, a series of TSP$(t, k)$ are solved to form the final solution. The general framework of TLH is shown in Fig. 3.1. We detail the three levels of TLH in Subsections 3.3.1 to 3.3.3 and outline TLH in Subsection 3.3.4.

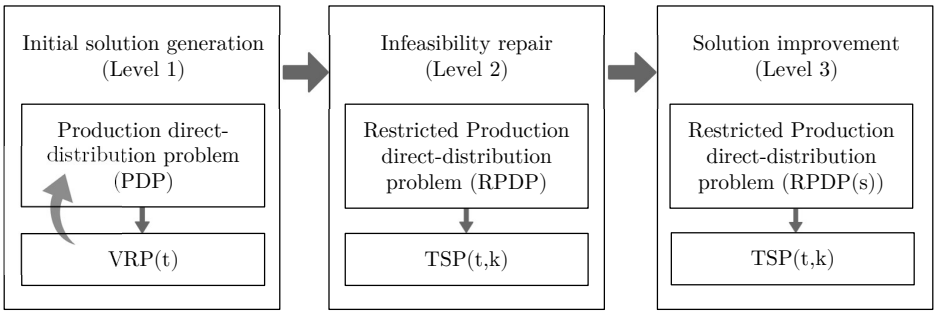



Fig. 3.1: General framework of TLH

### 3.3.1 Level 1: Initial solution generation

Level 1 aims to obtain a good initial solution to model $\mathcal{P}$ with a TIM. If the obtained solution by TIM is infeasible, it will be repaired in the next level. In Level 1, the original problem is decomposed into a PDP and a series of VRP$(t)$ that are iteratively solved by updating an approximate visit cost $\delta_{it}$ for each customer $i$ and each period $t$. During the iterative procedure, if the incumbent solution does not improve for a given number of iterations, a diversification mechanism is applied and the next iteration is called a diversification iteration. To save computation time, the production setup schedules are generated only in the first iteration and diversification iterations in TIM. The generated production setup schedules are then fixed and used in the subsequent iterations until the next diversification iteration. In TIM, for the $s^{th}$ iteration that is either the first or a diversification iteration, the first subproblem can be formulated as an MILP denoted as $\mathcal{P}_1(s)$ that comes from model $\mathcal{P}$ with several modifications. The vehicle routing constraints (3.7)-(3.13) are first relaxed and aggregate vehicle capacity constraints are added. Two new variables and a parameter are then introduced. Finally, the last term in objective function (3.1) is replaced with a term denoting the approximate routing cost. To formulate $\mathcal{P}_1(s)$, the following additional parameter and variables are defined:

Parameter

$\delta_{it}^s$ approximate visit cost from the plant to customer $i$ in period $t$ for the $s^{th}$ iteration. Initially, for $s = 1$ , $\delta_{it}^s$ is set to a specific value, while for $s > 1$, $\delta_{it}^s$ is updated based on the VRP($t$) solution in the previous iteration.

Decision variables

$y_i^{pt}$ quantity of product $p$ delivered to customer $i$ in period $t$;

$v_i^t$ equal to 1 if customer $i$ is visited in period $t$; and 0 otherwise.

Note that $v_i^t$ and $y_i^{pt}$ replace $v_i^{kt}$ and $y_i^{pkt}$ in model $\mathcal{P}$ by removing the vehicle index $k$. Thus the capacity of each individual vehicle is no longer restricted in PDP($s$). Instead, aggregate vehicle capacity constraints that limit the total delivery quantity within a period are introduced. For the $s^{th}$ iteration that is the first or a diversification iteration, PDP is formulated as a MILP (model $\mathcal{P}_1(s)$) as follows:

$$\mathcal{P}_1(s): \ \min \sum_{p\in P}\sum_{t\in T}(a_p\xi_{pt} + b_p w_{pt}) + \sum_{i\in R}\sum_{p\in P}\sum_{t\in T} e_p z_i^{pt} + \sum_{i\in N}\sum_{p\in P}\sum_{t\in T} h_i^p I_i^{pt} + \sum_{i\in R}\sum_{t\in T}\delta_{it}^s v_i^t \tag{3.21}$$

s.t.

$(3.4)-(3.6), (3.14)-(3.16), (3.18)$ and

$$I_0^{pt} = I_0^{p,t-1} + \xi_{pt} - \sum_{i\in R} y_i^{pt}, \ \forall p \in P, t \in T \tag{3.22}$$

$$I_i^{pt} = I_i^{p,t-1} + z_i^{pt} + y_i^{pt} - d_i^{pt}, \ \forall i \in R, p \in P, t \in T \tag{3.23}$$

$$\sum_{i\in R}\sum_{p\in P} y_i^{pt} \le |K|V, \ \forall t \in T \tag{3.24}$$

$$\sum_{p\in P} y_i^{pt} \le V v_i^t, \ \forall i \in R, t \in T \tag{3.25}$$

$$y_i^{pt} \ge 0, \ \forall i \in R, p \in P, t \in T \tag{3.26}$$

$$v_i^t \in \{0,1\}, \ \forall i \in R, t \in T \tag{3.27}$$

The objective function (3.21) minimizes the total production, inventory, outsourcing and visit cost. Constraints (3.22) and (3.23) correspond to the inventory flow balance at the plant and customers, respectively. Constraints (3.24) indicate that the total delivery quantity in each period cannot exceed the total vehicle capacity. Constraints (3.25) ensure delivery quantity to be 0 if a customer is not visited. Constraints (3.26) and (3.27) bound the variables. Model $\mathcal{P}_1(s)$ consists of determining production setup schedule $\hat{w}_{pt}$, production quantity $\hat{q}_{pt}$, outsourcing quantity $\hat{z}_i^{pt}$, inventory quantity $\hat{I}_i^{pt}$, delivery schedule $\hat{v}_i^t$, and delivery quantity $\hat{y}_i^{pt}$. Although $\mathcal{P}_1(s)$ is NP-hard, we

can obtain a relatively good feasible solution with a standard commercial solver. Once $\mathcal{P}_1(s)$ is solved, a production setup schedule $\mathcal{W} = \{\hat{w}_{pt} | p \in P, t \in T\}$ can be proposed. For a subsequent iteration $s$ that is not a diversification iteration, the first-phase problem becomes a RPDP denoted as $\mathcal{P}_1^{'}(s)$ can be formulated as follows:

$$\mathcal{P}_1^{'}(s): \ \min \sum_{p\in P}\sum_{t\in T} a_p \xi_{pt} + \sum_{i\in R}\sum_{p\in P}\sum_{t\in T} e_p z_i^{pt} + \sum_{i\in N}\sum_{p\in P}\sum_{t\in T} h_i^p I_i^{pt} + \sum_{i\in R}\sum_{t\in T} \delta_{it}^s v_i^t \tag{3.28}$$

s.t.

$$(3.4), (3.6), (3.14) - (3.16), (3.22) - (3.27), \text{ and}$$

$$\xi_{pt} \ \leq \ C\hat{w}_{pt}, \ \forall\, p \in P, t \in T \tag{3.29}$$

Solving $\mathcal{P}_1(s)$ or $\mathcal{P}_1^{'}(s)$ decides the delivery schedule $\hat{v}_i^t$ and the corresponding delivery quantity $\hat{y}_i^{pt}$. Then in the second phase of TIM, a series of VRP($t$) for each period $t \in T$ can be formulated and solved to determine the number of vehicles used, the customers each of them visits and the visit sequence. If the number of required vehicles in the solution to all VRP($t$) does not exceed the fleet size $|K|$, a feasible solution to $\mathcal{P}$ can be formed by the solutions of $\mathcal{P}_1(s)$ or $\mathcal{P}_1^{'}(s)$ and VRP($t$). Otherwise, TIM provides useful information to construct a feasible solution to $\mathcal{P}$ at the next level.

At the end of the $s^{th}$ iteration, the visit cost $\delta_{it}^{s+1}$ should be updated from the solution of VRP($t$). If the $(s+1)^{th}$ iteration is a diversification iteration, we generate $\delta_{it}^{s+1}$ with (3.30); otherwise, if customer $i$ is visited in period $t$, we calculate $\delta_{it}^{s+1}$ with (3.31); otherwise, it is calculated with (3.32). Equations (3.30)-(3.32) are listed below:

$$\delta_{it}^{s+1} = RGIF[\min_{j\in N}\{c_{ij}\}, c_{0i} + c_{i0}] \tag{3.30}$$

$$\delta_{it}^{s+1} = \mu(c_{i^-i} + c_{ii^+} - c_{i^-i^+}) + \rho(1 - L^{kt}/V)c_{0i} \tag{3.31}$$

$$\delta_{it}^{s+1} = \min_{k\in K}\{\mu\Delta + \rho(1 - L^{kt}/V)c_{0i}\} \tag{3.32}$$

where RGIF represents a randomly generated integer number from the interval; $\mu$ ($0 \leq \mu \leq 1$) and $\rho$ ($0 \leq \rho \leq 1$) are parameters. $i^-$ and $i^+$ denote the immediate predecessor and successor nodes of node $i$ in the visit sequence, respectively. $L^{kt}$ is the total delivery quantity by vehicle $k$ in period $t$ and $\Delta_i^{kt}$ is the cheapest insertion cost for inserting $i$ into an existing route performed by vehicle $k$ in period $t$. The update procedure attempts to better approximate the visit cost for a customer by considering both the customer clustering and the truckload utilization rate. Parameters $\mu$ and $\rho$ are used to adjust the weight of these two aspects. When solving $\mathcal{P}_1^{'}(s+1)$, where $s+1$

is a diversification iteration, equation (3.30) is used to restart the iterative procedure with some random values of $\delta_{it}^{s+1}$ to diversify the $\mathcal{P}_1'(s+1)$. The visit cost update procedure (3.31) and (3.32) tends to drive the solver to remove a customer from a route if $\delta_{it}^{s+1}$ is large and insert a customer into a route if $\delta_{it}^{s+1}$ is small.

The aforementioned process is repeated for a number of iterations, then TIM stops and outputs the best solution to be used in the next level.

The TIM has been developed based on the MIP-based iterative heuristic (IM-VRP) proposed by [1]. The common points among the TIM, the IM-VRP and the three-phase heuristic (CCJ-DH) recently developed by [34] are as follows: 1) the original problems are decomposed into subproblems that are solved sequentially and iteratively; 2) an approximate visit cost is introduced and it is updated in each iteration to serve as a surrogate of the real routing cost. However, our TIM differs from IM-VRP and CCJ-DH with the following new features: 1) For the production setup schedule, TIM and CCJ-DH generate it only in the first and diversification iterations to save computation time, while IM-VRP generates it in each iteration. In diversification iterations, TIM and IM-VRP diversify the search by re-generating the value of the approximate visit cost while CCJ-DH does that by adding local branching inequalities to change the setup schedule; 2) TIM allows infeasible solution and it is repaired only in Level 2 of TLH to save computation time, while IM-VRP and CCJ-DH attempt to repair each infeasible solutions once it is found; and 3) TIM uses a new update mechanism to update the approximate visit cost by considering both customer clustering and truckload utilization rate, while IM-VRP and CCJ-DH do that by only considering the customer clustering.

### 3.3.2 Level 2: Infeasibility repair

Level 2 focuses on repairing the infeasible solution provided by Level 1 and consolidating vehicle routes. It consists of solving an RPDP and a series of TSP$(t,k)$ to form a feasible solution to $\mathcal{P}$. In the solution obtained from Level 1, let $\mathcal{W} = \{\hat{w}_{pt} | p \in P, t \in T\}$ be the production setup schedule. The actual number of vehicles used in period $t \in T$ is denoted as $|\mathcal{K}_t|$, if there exists any period $t$ with that $|\mathcal{K}_t| > |K|$, the solution is infeasible to $\mathcal{P}$. In this case, the vehicles used in each period are sorted in non-increasing order of their delivery quantity to form a set of vehicles $K = \{1, \ldots, |K|\}$ and a set of dummy vehicles $\mathcal{V} = \{|K|+1, \ldots, |\mathcal{K}_t|\}$. Let $\mathcal{R}^{kt}$ be the set of customers served by vehicle $k$ in period $t$. Then a set of customers $\mathcal{B}$ consisting of all customers served by dummy vehicles is formed, i.e., $\mathcal{B} = \{i | i \in \mathcal{R}^{kt}, t \in T, k \in \mathcal{V}\}$. Let $\mathcal{D} = \{\hat{v}_i^{kt}\}$ be the delivery setup schedule formed by non-dummy vehicles, i.e., $\mathcal{D} = \{\hat{v}_i^{kt} | i \in R \setminus \mathcal{B}, k \in K, t \in T\}$.

Then the second level RPDP can be formulated as a MILP denoted as $\mathcal{P}_2$ that drives from $\mathcal{P}$ by relaxing its vehicle routing constraints (3.10)-(3.13) and fixing the production setup schedule to $\mathcal{W} = \{\hat{w}_{pt} | p \in P, t \in T\}$ and part of the delivery setup schedules to $\mathcal{D} = \{\hat{v}_i^{kt} | i \in R \setminus \mathcal{B}, k \in K, t \in T\}$. To formulate $\mathcal{P}_2$, an additional parameter $\sigma_i^{kt}$ is defined to represent the approximate visit cost to customer $i$ by vehicle $k$ in period $t$. It can be calculated based on the solution obtained from Level 1. In the best solution of Level 1, if customer $i$ is visited in period $t$, then $\sigma_i^{kt}$ is set to $c_{i^-i} + c_{ii^+} - c_{i^-i^+}$; otherwise, $\sigma_i^{kt}$ is set to $\Delta_i^{kt}$. $\mathcal{P}_2$ is formulated as follows:

$$\mathcal{P}_2: \min \sum_{p\in P}\sum_{t\in T} a_p \xi_{pt} + \sum_{i\in R}\sum_{p\in P}\sum_{t\in T} e_p z_i^{pt} + \sum_{i\in N}\sum_{p\in P}\sum_{t\in T} h_i^p I_i^{pt} + \sum_{i\in \mathcal{B}}\sum_{k\in K}\sum_{t\in T} \sigma_i^{kt} v_i^{kt} \tag{3.33}$$

s.t.

$(3.2)-(3.4), (3.6)-(3.7), (3.14)-(3.17)$ and

$$\xi_{pt} \le C\hat{w}_{pt}, \ \forall p \in P, t \in T \tag{3.34}$$

$$\sum_{p\in P} y_i^{pkt} \le V\hat{v}_i^{kt}, \ \forall i \in R \setminus \mathcal{B}, k \in K, t \in T \tag{3.35}$$

$$\sum_{p\in P} y_i^{pkt} \le V v_i^{kt}, \ \forall i \in \mathcal{B}, k \in K, t \in T \tag{3.36}$$

$$\sum_{k\in K} v_i^{kt} \le 1, \ \forall i \in \mathcal{B}, t \in T \tag{3.37}$$

$$v_i^{kt} \in \{0,1\}, \ \forall i \in \mathcal{B}, k \in K, t \in T \tag{3.38}$$

The objective function (3.33) minimizes the total cost. Constraints (3.34) indicate that the production quantity in each period must respect the given production setup schedule. Constraints (3.35) and (3.36) require the delivery quantity to each customer to be 0 if not visited. Constraints (3.37) forbid split delivery to customer $i \in \mathcal{B}$. Constraints (3.38) define the delivery setup variable as a binary variable.

With the delivery setup schedule $\mathcal{D} = \{\hat{v}_i^{kt} | i \in R, k \in K, t \in T\}$ proposed by $\mathcal{P}_2$, a series of TSP$(t,k)$ (up to $|T| \times |K|$) are solved to find a good vehicle visiting sequence. Then the solutions to model $\mathcal{P}_2$ and all TSP$(t,k)$ form a feasible solution to model $\mathcal{P}$.

If the solution obtained from Level 1 is feasible, a series of TSP$(t,k)$ are directly solved with $\mathcal{D} = \{\hat{v}_i^{kt} | i \in R, k \in K, t \in T\}$ to further improve it. Then the incumbent solution is updated and used in the next level.

### 3.3.3 Level 3: Incumbent solution improvement

Level 3 aims to improve the incumbent solution by using a fix-and-optimize procedure. Its basic idea is to iteratively solve a series of subproblems in which a subset of

binary variables are fixed so that the subproblems can be solved optimally by a MIP solver [89]. In the $s^{th}$ iteration, all continuous variables $(\xi_{pt}, z_i^{pt}, I_i^{pt}, y_i^{pkt})$ and binary variables $v_s^{kt}$ of a single customer $s \in R = \{1, 2, \ldots, n\}$ are optimized, while variables $v_i^{kt}, i \in R \setminus \{s\}, k \in K, t \in T$ are fixed to $\hat{v}_i^{kt}$, i.e., $v_i^{kt} = \hat{v}_i^{kt}$. The production setup schedule is fixed to $\mathcal{W} = \{\hat{w}_{pt} | p \in P, t \in T\}$. According to the incumbent solution provided by Level 2, we recalculate the approximate visit cost for customer $s$ according to the delivery setup schedule $\mathcal{D}$ as follows: if customer $s$ is visited in period $t$, $\sigma_s^{kt}$ is set to $c_{s^-s} + c_{ss^+} - c_{s^-s^+}$; otherwise, $\sigma_s^{kt}$ is set to $\Delta_s^{kt}$ which is the minimum insertion cost by inserting $s$ into the route of vehicle $k$ in period $t$, and the corresponding inserting position is recorded. The $s^{th}$ iteration of the procedure consists of formulating a new RPDP as a MILP as follows:

$$\mathcal{P}_3(s): \min \sum_{p\in P}\sum_{t\in T} a_p \xi_{pt} + \sum_{i\in R}\sum_{p\in P}\sum_{t\in T} e_p z_i^{pt} + \sum_{i\in N}\sum_{p\in P}\sum_{t\in T} h_i^p I_i^{pt} + \sum_{k\in K}\sum_{t\in T} \sigma_s^{kt} v_s^{kt} \tag{3.39}$$

s.t.

$$(3.2)-(3.4), (3.6)-(3.7), (3.14)-(3.17), (3.34), \text{ and}$$

$$\sum_{p\in P} y_i^{pkt} \le V \hat{v}_i^{kt}, \ \forall\, i \in R \setminus \{s\}, k \in K, t \in T \tag{3.40}$$

$$\sum_{p\in P} y_s^{pkt} \le V v_s^{kt}, \ \forall\, k \in K, t \in T \tag{3.41}$$

$$\sum_{k\in K} v_s^{kt} \le 1, \ \forall\, t \in T \tag{3.42}$$

$$v_s^{kt} \in \{0, 1\}, \ \forall\, k \in K, t \in T \tag{3.43}$$

The objective function (3.39) and constraints (3.40)-(3.43) are similar to the objective function (3.33) and constraints (3.35)-(3.38), respectively. Note that model $\mathcal{P}_3(s)$ focuses on iteratively improving the incumbent solution to model $\mathcal{P}$ by adjusting the delivery setup schedule of customer $s$ while model $\mathcal{P}_2$ in level 2 tries to insert a set of customers $\mathcal{B}$ into existing routes performed by non-dummy vehicles.

$\mathcal{P}_3(s)$ is solved to determine $v_s^{kt}$ and product flow $(\xi_{pt}, z_i^{pt}, I_i^{pt}, y_i^{pkt})$. Then all the routes are updated based on an insertion or removal strategy for customer $s$. Note that for each route of vehicle $k$ in period $t$, if customer $s$ is removed, a new route is directly formed by connecting the precedent and the successive customers; if customer $s$ is inserted, it would be in the position recorded when calculating $\sigma_s^{kt}$. Then a feasible solution to $\mathcal{P}$ is formed. According to the solution of the $s^{th}$ iteration, $\sigma_{s+1}^{kt}$ of the $(s+1)^{th}$ iteration is set to $c_{s^-s} + c_{ss^+} - c_{s^-s^+}$ if customer $s$ is visited in period $t$; otherwise, it is set to $\Delta_s^{kt}$. The procedure is repeated until all customers are enumerated. Finally, the obtained solution from the fix-and-optimize procedure

is further enhanced by solving a series of TSP$(t, k)$ for each period $t$ and each used vehicle $k$.

### 3.3.4 Three-level heuristic outline

The TLH can be summarized in Algorithm 3.1, in which the following parameters are defined. $sol^*$ and $sol$ are used to store the best-obtained solution and the current solution, while $obj^*$ and $obj$ denote the obtained best objective value and the current objective value, respectively. $s$ is the iteration counter. $M$ indicates the total number of iterations that are allowed in TIM. The notation $flag$ serves as an iteration indicator in the first level. $flag = 0$ indicates the first or a diversification iteration, while $flag = 1$ indicates other iterations.

In Algorithm 1, lines 1-20 correspond to the TIM in Level 1. Lines 1-9 solve a special lot-sizing problem and a routing problem to form a solution to $\mathcal{P}$ that may be infeasible. Lines 10-14 update the best solution and approximate visit cost. Lines 15-18 correspond to the diversification mechanism. Lines 21-26 represent Level 2. Particularly, lines 21-23 repair the infeasible solutions. A series of TSP$(t, k)$ are solved in line 24. Lines 27-33 are devoted to the iterative fix-and-optimize procedure in Level 3. Line 34 solves a series of TSP. Finally, the best solution is updated and returned in lines 35 and 36.

## 3.4 Computational experiments

In this section, 225 newly generated MPRPOS instances and 1530 PRP benchmark instances are tested to evaluate the performance of the proposed heuristic. TLH is implemented in C++, and all tests are performed on a PC with Intel Core i7 CPU (2.5 GHz) and 8 GB RAM. The tested instances and detailed computational results are available at `http://www.mediafire.com/file/cn4r6aqwm48gog4/DataAndResults.rar`. The initial approximate visit cost $\delta_{it}^1$ is set differently for MPRPOS and PRP based on the problem structure. For the MPRPOS, since outsourcing is considered, the approximate visit cost will greatly impact the production setup schedule and the production quantity. $\delta_{it}^1$ is generated using equation (3.30) to better approximate the real routing cost and generate a good production setup schedule. For the classic PRP, although the approximate visit cost impacts the production setup and quantity decisions, $\delta_{it}^1$ is set to 0 to quickly generate a production setup schedule and save computation time. Parameters $\mu$ and $\rho$ are set to 0.5 for all instances. The threshold for diversification mechanism is set to 3, which indicates that the diversification mechanism is applied after 3 successive iterations without objective value improvements after the previous diversification iteration. To make a trade-off between

**Algorithm 3.1** TLH for MPRPOS and for PRP

1. Initialize $flag = 0, s = 1, sol^* \leftarrow \phi, sol \leftarrow \phi, obj, obj^* \leftarrow +\infty$
   $M, \delta_{it}^s$ for all $i \in R, t \in T$
2. **while** $(s \leq M)$
3. **if** $(flag = 0)$
4. Solve $\mathcal{P}_1(s)$, output $\hat{w}_{pt}, \hat{\xi}_{pt}, \hat{z}_i^{pt}, \hat{I}_i^{pt}, \hat{y}_i^{pt}$ and $\hat{v}_i^t$
5. **else if**$(flag = 1)$
6. Solve $\mathcal{P}_1'(s)$, output $\hat{q}_{pt}, \hat{z}_i^{pt}, \hat{I}_i^{pt}, \hat{y}_i^{pkt}$ and $\hat{v}_i^t$
7. **end if**
8. Solve VRP$(t)$ for each period $t$ with $\hat{v}_i^t$ and $\hat{y}_i^{pt}$ known for all $i \in R, p \in P$
9. Form a solution $sol$ to $\mathcal{P}$ (probably infeasible) and calculate $obj$
10. **if** $(obj < obj^*)$
11. Set $obj^* \leftarrow obj$ and update the corresponding best solution $sol^* \leftarrow sol$
12. **end if**
13. Calculate visit cost $\delta_{it}^{s+1}$ with (3.31) and (3.32)
14. Set $flag = 1$
15. **if** the diversification condition is met **then**
16. Set $flag = 0$
17. Reset $\delta_{it}^{s+1}$ with (3.30)
18. **end if**
19. Set $s \leftarrow s + 1$
20. **end while**
21. **if** $sol^*$ is infeasible **then**
22. Solve $\mathcal{P}_2$ with $\sigma_i^{kt}$ and output $\hat{q}_{pt}, \hat{z}_i^{pt}, \hat{I}_i^{pt}, \hat{y}_i^{pkt}$ and $\hat{v}_i^{kt}$
23. **end if**
24. Solve a series of TSP$(t, k)$ for all $k \in K$ and $t \in T$ with $\hat{v}_i^{kt}$ known for all $i \in R$
25. Form a feasible solution $sol$ to $\mathcal{P}(s)$ and calculate $obj$
26. Set $sol^* \leftarrow sol$ and $obj^* \leftarrow obj$
27. Set $s = 1$ and calculate visit cost $\sigma_s^{kt}$ based on $sol^*$
28. **while** ( $s \leq n$)
29. Solve $\mathcal{P}_3(s)$ and output $\hat{v}_s^{kt}$
30. Perform insertion or removal strategy for customer $s$
31. Update the best solution $sol^*$ and visit cost $\sigma_{s+1}^{kt}$
32. Set $s \leftarrow s + 1$
33. **end while**
34. Solve TSP$(t, k)$ for each used vehicle in each period
35. Update $sol^*$ and $obj^*$
36. Return $sol^*$ and $obj^*$

solution quality and computation time, the number of iterations $M$ is set according to the size of instances. Particularly, for all MPRPOS instances, $M$ is set to 10; for PRP benchmark instances, $M$ is set to 200 for small-sized instance set A1, 50 for medium- and large-sized instances sets A2 and A3, and 10 for very large-sized instance set B. Note that subproblems PDP, RPDP, and RPDP(s) and in Fig. 1 are solved by the state-of-the-art commercial solver CPLEX. Since solving PDP is time-consuming, CPLEX outputs their solutions if the running time is larger than 500s or its gap is less than 0.01%, respectively. Subproblems VRP($t$) and TSP($t, k$) in Fig. 1 are solved by calling the state-of-the-art record-to-record metaheuristic (RTR) [65] and Lin-Kernighan Heuristic (LKH) [74], respectively. The corresponding codes of RTR and LKH are available at `https://www.coin-or.org/download/source/VRPH/` and `http://www.akira.ruc.dk/~keld/research/LKH/`.

In Section 3.4.1, we present the newly generated MPRPOS instances and compare the performance of TLH with that of CPLEX on these instances. In addition, we analyze how outsourcing impacts the performance of TLH. In subsection 3.4.2, we briefly introduce the tested PRP benchmark instances, report the implementation environment of the state-of-the-art algorithms and compare the performance of TLH with these algorithms on these benchmark instances. Finally, we analyze the performance of TLH in subsection 3.4.3.

## 3.4.1 Computational experiments for MPRPOS

### 3.4.1.1 Instance generation

As MPRPOS is a newly studied problem, there are no benchmark instances in the literature. To validate the proposed model and evaluate the performance of TLH, we first randomly generate 150 instances (instance set C) based on the PRP instance generation in [15]. These instances consist of 30 sets with 5 instances in each set that is characterized by the number of customers $n$, periods $|T|$, and products $|P|$. Specifically, $n$ is set to 10, 14, 50, 100, and 200; $|T|$ is set to 3 and 6; $|P|$ is set to 3, 6, and 12. Fleet size $|K|$ is set to 1 for instances with 10 and 14 customers, and 5, 10 and 20 for instances with 50, 100 and 200 customers, respectively. Customers or plant locations $(X_i, Y_i)$ , which correspond to the coordinates of node $i$ in graph $G$, are randomly generated from U[0,1000]. Travel cost $c_{ij}$ is set to the Euclidean distance between nodes $i$ and $j$. At the beginning of the planning horizon, the initial inventory $I_i^{p0}$ is set to 0 for all $i \in N$ and $p \in P$. The detailed generation of parameters for MPRPOS is given in Table 3.1.

Table 3.1: Parameters for MPRPOS model

| Parameters | Generation description |
|---|---|
| $c_{ij}$ | $= \lfloor \sqrt{(X_i - X_j)^2 + (Y_i - Y_j)^2} + 0.5 \rfloor$ |
| $d_i^{pt}$ | randomly generated from U[5, 25] |
| $C$ | $= g(\sum_{i \in R} \sum_{p \in P} \sum_{t \in T} d_i^{pt})/\lvert T \rvert$, where $g$ is randomly generated from U[2, 4] |
| $U_0$ | $= gC$,where $g$ is randomly generated from U[1, 3] |
| $U_i$ | $= g_i(\sum_{p \in P} \sum_{t \in T} d_i^{pt})/\lvert T \rvert$, where $g_i$ is randomly generated from U[2, 6], $i \in R$ |
| $h_i^p$ | randomly generated from U[0.5, 1.5] |
| $a_p$ | randomly generated from U[5, 10] |
| $e_p$ | $= g_p a_p$,where $g_p$ is randomly generated from U[1.5, 2], $p \in P$ |
| $b_p$ | $= 30a_p, 50a_p, 200a_p, 400a_p$, and $800a_p$, for $n = 10, 14, 50, 100$ and 200 |
| $V$ | $= g(\sum_{i \in R} \sum_{p \in P} \sum_{t \in T} d_i^{pt})/\lvert K \rvert/\lvert T \rvert$,where $g$ is randomly generated from U[1.5, 2] |

#### 3.4.1.2 Lower bound of MPRPOS

Our preliminary experiments show that CPLEX can only optimally solve instances with up to 14 customers and it has difficulty even in providing lower bounds for large-sized instances. To evaluate the performance of TLH, we propose a lower bound to MPRPOS based on [19] in which an allocation model is proposed and proved to be able to provide valid lower bounds for PRP. We present a similar model denoted as $\mathcal{P}_{LB}$ as follows:

$$\mathcal{P}_{LB}: \min \sum_{p \in P} \sum_{t \in T} (a_p \xi_{pt} + b_p w_{pt}) + \sum_{i \in R} \sum_{p \in P} \sum_{t \in T} e_p z_i^{pt} + \sum_{i \in N} \sum_{p \in P} \sum_{t \in T} h_i^p I_i^{pt} + \sum_{i \in R} \sum_{t \in T} \epsilon_i v_i^t + \sum_{i \in R} \sum_{p \in P} \sum_{t \in T} \frac{\epsilon_0 y_i^{pt}}{V} \tag{3.44}$$

s.t.

$$(3.4) - (3.6), (3.14) - (3.16), (3.18), (3.22) - (3.27)$$

The objective function (3.44) minimizes the total production, outsourcing, inventory and approximate routing cost which is denoted by the last two terms. In (3.44), $\epsilon_i$ denotes the minimum transportation cost from customer $i$ to any other customer or the plant, i.e., $\epsilon_i = \min_{j \in N \setminus \{i\}} c_{ij}$. Particularly, $\epsilon_0$ denotes the minimum cost from the plant to any customer. Optimally solving $\mathcal{P}_{LB}$ provides a valid lower bound to model $\mathcal{P}$. If $\mathcal{P}_{LB}$ is not optimally solved, a lower bound to $\mathcal{P}_{LB}$ is outputted. For more details, please refer to [19].

#### 3.4.1.3 Computational results on MPRPOS instances

In this subsection, we first compare the results of TLH and CPLEX for small-sized instances with up to 14 customers and 1 vehicle. To directly solve the MPRPOS, CPLEX default settings are used and the subtour elimination constraints (3.13) are iteratively added. Computation time for each iteration is limited to 3600s. At the end of each iteration, we check the solution: if there is no subtour, then an optimal solution is obtained; otherwise, if the time limit (3600s) is reached, CPLEX fails to provide any feasible solution (upper bound) and only a lower bound is obtained. If there is one or more subtours in the solution and the time limit is not reached, the relaxed model is re-solved by adding a set of subtour elimination constraints concerning these subtours. Note that the total computation time may exceed 3600s because the stopping criterion is only checked at the end of each iteration. Results provided by CPLEX and TLH for small-sized instances are reported in Table 3.2, where columns 1 to 3 denote the numbers of customers, periods and products, respectively. Columns "$obj^*$" and "$obj^C$" denote the best objective value obtained by TLH and the optimal objective value or a lower bound provided by CPLEX, respectively. Column "Gap" indicates the average gap between $obj^*$ and $obj^C$, where Gap=$(obj^* - obj^C)/obj^C * 100\%$. Columns "T-TLH" and "T-CPLEX" provide the computation times by TLH and CPLEX, respectively. We can see from Table 3.2

Table 3.2: MPRPOS computational results on instance up to 14 customers

| $n$ | $\|T\|$ | $\|P\|$ | $obj^*$ | $obj^C$ | Gap(%) | T-TLH(s) | T-CPLEX(s) |
|---|---|---|---|---|---|---|---|
| 10 | 3 | 3 | 15942 | 15810 | 0.85 | 1.3 | 149.5 |
| | | 6 | 32122 | 32063 | 0.19 | 1.8 | 330.5 |
| | | 12 | 52638 | 52550 | 0.18 | 10.7 | 1971.7 |
| | 6 | 3 | 32983 | 31789 | 3.69 | 7.8 | 3525.1 |
| | | 6 | 58697 | 56207 | 4.43 | 48.6 | 4196.8 |
| | | 12 | 103295 | 100573 | 2.73 | 506.2 | 4409.7 |
| 14 | 3 | 3 | 20726 | 20250 | 2.45 | 2.4 | 1619.6 |
| | | 6 | 39690 | 39452 | 0.62 | 6.9 | 1663.0 |
| | | 12 | 73163 | 72967 | 0.28 | 17.6 | 1227.5 |
| | 6 | 3 | 45553 | 43661 | 4.29 | 17.0 | 3845.6 |
| | | 6 | 78875 | 77409 | 1.93 | 50.5 | 2812.2 |
| | | 12 | 141033 | 137948 | 2.25 | 512.4 | 4074.8 |
| Average | | | 57893 | 56723 | 1.99 | 98.6 | 2486 |

that the average upper bound obtained by TLH and the average optimal objective value (or lower bound) provided by CPLEX are 57893 and 56723, respectively. TLH achieves relatively good-quality solutions with an average gap of 1.99% in less than 110s. The average computation time of CPLEX is over 2480s which is more than

20 times as long as that of TLH. We can observe from Columns 7 and 8 that for a given number of customers, the computation time of TLH and CPLEX increase with the number of periods and products, yet the latter increases much more quickly than the former. Take instances with 10 customers and 3 periods as an example, the average computation time of CPLEX increases from 149.5s to 1971.7s when the number of products increases from 3 to 12, while that of TLH increases from 1.3s to 10.7s; for instances with 10 customers and 3 products, the computation time of CPLEX increases from 149.5s to 3525.1s when the number of periods increases from 3 to 6, while that of TLH increases from 1.3s to 7.8s.

To evaluate the performance of TLH for large-sized instances, two lower bounds denoted as LB1 and LB2 are proposed by directly solving model $\mathcal{P}$ and $\mathcal{P}_{LB}$ with CPLEX. Computational results for instances with up to 200 customers are reported in Table 3.3, in which the first three columns represent the numbers of customers, periods and products, respectively. Column "$obj^*$" is the best upper bound obtained by TLH, while columns "LB1" and "LB2" are lower bounds provided by CPLEX through solving MPRPOS and MLB, respectively. Columns "Gap1" and "Gap2" give the gaps of $obj^*$ with respect to LB1 and LB2 respectively, where Gap1=($obj^*$-LB1)/LB1*100% and Gap2=($obj^*$-LB2)/LB2*100%. In particular, the numbers in bold indicate the better gap between the two obtained lower bounds. Columns "T-TLH", "T-LB1" and "T-LB2" represent the computation times to obtain $obj^*$, LB1 and LB2, respectively.

As can be seen from Table 3.3 that the average best upper bound obtained by TLH and the average lower bound by $\mathcal{P}_{LB}$ are 547931 and 496828, respectively. CPLEX has difficulties in directly solving model $\mathcal{P}$ to provide lower bounds for instances with more than 100 customers and 6 periods and it only provides better bounds in three sets of instances. $\mathcal{P}_{LB}$ can provide lower bounds for all large-sized instances and it generally obtains better lower bounds. This may because that $\mathcal{P}_{LB}$ is generally less complex than $\mathcal{P}$ and CPLEX can provide better bounds to $\mathcal{P}_{LB}$ on large-sized instances. Column "Gap2" shows that the average gap of $obj^*$ with respect to LB2 is 9.9%. Regarding the computation time, TLH needs on average 706s that is a relatively short computation time for such a complex problem. The average computation times for solving $\mathcal{P}$ and $\mathcal{P}_{LB}$ by CPLEX are more than 3600s and 900s, respectively. It increases noticeably with the number of periods and products.

To further compare the results of TLH and CPLEX, the MPRPOS instances are grouped by the number of customers, yielding 5 sets of instances with 10, 14, 50, 100, and 200 customers.. The average results for each set of instances with the same number of customers are shown in Fig. 3.2 (a), which indicates that the average gap increases with the number of customers, while the largest average gap does not exceed

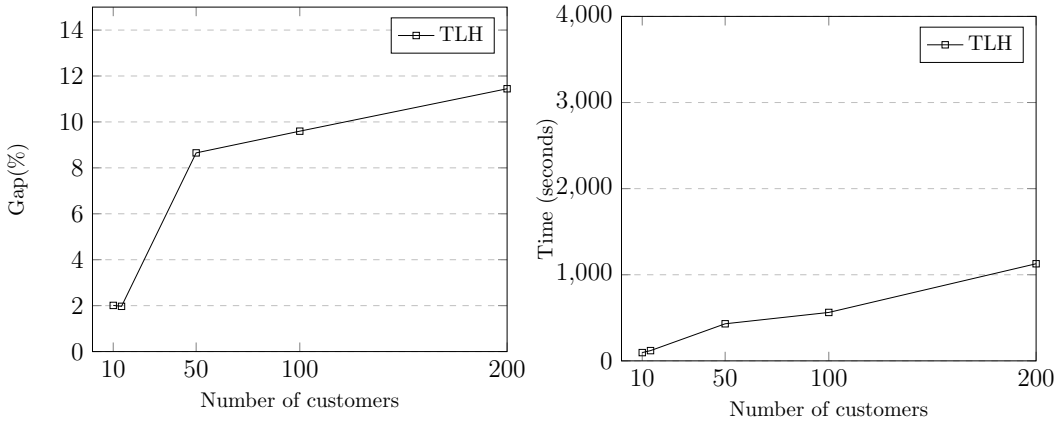
Table 3.3: MPRPOS computational results on instance up to 200 customers

| $n$ | $\|T\|$ | $\|P\|$ | $obj^*$ | LB1 | LB2 | Gap1 | Gap2 | T-TLH | T-LB1 | T-LB2 |
|---|---|---|---|---|---|---|---|---|---|---|
| 50 | 3 | 3 | 77806 | 70313 | 70051 | **10.66** | 11.07 | 11.6 | 3600.1 | 0.9 |
| | | 6 | 136737 | 126109 | 126008 | **8.43** | 8.52 | 31.6 | 4091.8 | 2.7 |
| | | 12 | 255275 | 244921 | 245617 | 4.23 | **3.93** | 286.4 | 3600.2 | 65.2 |
| | 6 | 3 | 153042 | 136723 | 137523 | 11.94 | **11.28** | 160.7 | 3600.2 | 41.6 |
| | | 6 | 278465 | 253216 | 256246 | 9.97 | **8.67** | 1031.4 | 3600.2 | 979.0 |
| | | 12 | 541375 | 506202 | 499273 | **6.95** | 8.43 | 1061.4 | 3600.4 | 3600.2 |
| 100 | 3 | 3 | 155884 | 136796 | 136894 | 13.95 | **13.87** | 37.6 | 3600.6 | 2.0 |
| | | 6 | 277329 | 255065 | 256414 | 8.73 | **8.16** | 117.9 | 3600.7 | 5.6 |
| | | 12 | 513744 | 483816 | 494759 | 6.19 | **3.84** | 632.9 | 3600.8 | 68.3 |
| | 6 | 3 | 304265 | - | 272599 | - | **11.62** | 438.2 | - | 77.2 |
| | | 6 | 524291 | - | 470080 | - | **11.53** | 1010.6 | - | 2141.6 |
| | | 12 | 1009348 | - | 929678 | - | **8.57** | 1136.9 | - | 3600.3 |
| 200 | 3 | 3 | 263952 | - | 230864 | - | **14.33** | 112.6 | - | 4.8 |
| | | 6 | 543135 | - | 506040 | - | **7.33** | 362.6 | - | 44.3 |
| | | 12 | 981553 | - | 939752 | - | **4.45** | 1404.0 | - | 162.4 |
| | 6 | 3 | 578216 | - | 497103 | - | **16.32** | 1237.6 | - | 489.2 |
| | | 6 | 1088271 | - | 978366 | - | **11.23** | 1479.6 | - | 3014.9 |
| | | 12 | 2180062 | - | 1895645 | - | **15.00** | 2169.4 | - | 3600.6 |
| Average | | | 547931 | - | 496828 | - | **9.90** | 706.8 | - | 994.5 |

12% for the largest-sized instances with 200 customers. Fig. 3.2 (b) shows that the average computation time of TLH is relatively short for all instances. In summary, our TLH provides good quality solutions within short computational times.

#### 3.4.1.4 Sensitivity analysis

To analyze the impact of different outsourcing cost on TLH, based on the standard instance set in Section 3.4.1.1, three new sets of 25 MPRPOS instances (75 new instances) each with $|T| = 6$ and $|P| = 6$ are generated by varying the value of $g_p$ among U[1.1, 1.4], U[2.0, 2.5], and U[2.5, 3.0], respectively. In total, 100 instances are tested including the basic set ($g_p$ =U[1.5, 2.0]) of instances with the same size. Computational results are presented in Table 3.4. The first three columns denote the number of customers, periods and products, respectively. The fourth column gives the $g_p$ value. The next four columns represent the upper bound (UB) obtained by TLH, the lower bound (LB) obtained by CPLEX, their gap calculated as gap=(UB-LB)/LB*100%, and the computation time of TLH, respectively. The last four columns present the average production, outsourcing, inventory, and routing costs, respectively. Fig. 3.3 shows the changes of total cost, production cost, outsourcing cost, inventory cost and routing cost with respect to the changes of $g_p$. Note that these values in Fig. 3.3 are calculated over all the instances with the same generation interval of $g_p$. We




(a) Average gaps for MPRPOS instances

(b) Computation times for MPRPOS instances

Fig. 3.2: Results for MPRPOS instances

can see from Table 3.4 that the average gap and computation time are 5.88% and 587.0s, respectively, which show that our TLH is relatively effective and efficient for solving MPRPOS instances with various outsourcing cost. From Table 3.4, we have the following observations: 1) When the outsourcing cost ($g_p$ =U[1.1,1.4]) is lower than that of the basic set, the MPRPOS instances can be solved to optimality for instances with up to 14 customers and all customers' demand is outsourced due to the small outsourcing cost. The gaps and computation time increase slowly with the number of customers. The maximum average gap and computation time do not exceed 3.09% and 627.9s, respectively. 2) When the outsourcing cost ($g_p$ =U[2.0,2.5] and $g_p$ =U[2.5,3.0]) is significantly higher than that of the basic set, the gaps and computation times are quite close for these two sets of instances. They increase with the number of customers except the set of instances with 200 customers. Most customers' demand is satisfied by in-house production for these instances. 3) For the set of basic instances ($g_p$ =U[1.5,2]), the average UB and gap are generally close to the instances with higher outsourcing cost. While for instances with 200 customers, the average UB, gap and computation time of the basic set are the highest compared with other instances. This may be because the tradeoff between the outsourcing and in-house production is far from obvious and has to be fine-tuned, which makes the problem more difficult. The above mentioned observations can be confirmed by Fig. 3.4 that shows the changes of different cost components as the value of $g_p$ changes. To sum up, the results in Table 3.4 indicate that outsourcing cost can impact the performance of TLH. If the outsourcing cost is low, and all customers' demand is satisfied by outsourcing, the problem is relatively easy to be solved. While the outsourcing cost and the in-house product cost are comparable, a strong tradeoff between production and outsourcing makes the problem hard to be solved. Finally, as the outsourcing

Table 3.4: Sensitivity analysis for MPRPOS instances with varying outsourcing cost

| $n$ | $\|T\|$ | $\|P\|$ | $g_p$ | UB | LB | gap(%) | time(s) | Pcost | Ocost | Icost | Rcost |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 10 | 6 | 6 | U[1.1,1.4] | 47171 | 47171 | 0.00 | 0.6 | 0 | 47171 | 0 | 0 |
| | | | U[1.5,2.0] | 58697 | 56207 | 4.43 | 48.6 | 35986 | 10425 | 4610 | 7677 |
| | | | U[2.0,2.5] | 58921 | 56761 | 3.82 | 47.3 | 43765 | 105 | 4839 | 10211 |
| | | | U[2.5,3.0] | 59089 | 56557 | 4.50 | 46.9 | 43609 | 104 | 5440 | 9936 |
| 14 | 6 | 6 | U[1.1,1.4] | 65358 | 65358 | 0.00 | 0.9 | 0 | 65358 | 0 | 0 |
| | | | U[1.5,2.0] | 78875 | 77409 | 1.93 | 50.5 | 56456 | 7334 | 6546 | 8539 |
| | | | U[2.0,2.5] | 79082 | 77516 | 2.05 | 74.6 | 62003 | 151 | 6755 | 10172 |
| | | | U[2.5,3.0] | 79634 | 77537 | 2.75 | 53.6 | 61919 | 270 | 7562 | 9884 |
| 50 | 6 | 6 | U[1.1,1.4] | 245043 | 243362 | 0.66 | 67.3 | 0 | 245043 | 0 | 0 |
| | | | U[1.5,2.0] | 278465 | 256246 | 8.70 | 1031.4 | 219310 | 4907 | 24129 | 30120 |
| | | | U[2.0,2.5] | 278635 | 256250 | 8.78 | 1026.3 | 224488 | 245 | 23481 | 30421 |
| | | | U[2.5,3.0] | 278373 | 256257 | 8.67 | 927.2 | 225002 | 6 | 22482 | 30884 |
| 100 | 6 | 6 | U[1.1,1.4] | 456779 | 444579 | 2.62 | 258.3 | 0 | 456779 | 0 | 0 |
| | | | U[1.5,2.0] | 524291 | 470080 | 11.71 | 1010.6 | 345450 | 94030 | 35691 | 49120 |
| | | | U[2.0,2.5] | 522751 | 470515 | 11.33 | 1007.3 | 420130 | 183 | 44916 | 57522 |
| | | | U[2.5,3.0] | 524455 | 470483 | 11.77 | 1111.4 | 418717 | 166 | 47812 | 57760 |
| 200 | 6 | 6 | U[1.1,1.4] | 968862 | 937183 | 3.09 | 627.9 | 0 | 968862 | 0 | 0 |
| | | | U[1.5,2.0] | 1088271 | 978366 | 11.19 | 1479.6 | 801918 | 104090 | 80857 | 101406 |
| | | | U[2.0,2.5] | 1074743 | 981848 | 9.70 | 1457.3 | 888061 | 0 | 74000 | 112682 |
| | | | U[2.5,3.0] | 1079264 | 983201 | 9.97 | 1411.5 | 892221 | 0 | 74987 | 112057 |
| Average | | | | 392338 | 363144 | 5.88 | 587.0 | 236952 | 100261 | 23205 | 31919 |

cost goes much higher than the in-house production cost, the problem can be treated as a classic PRP. The gaps and computation times for instances with different outsourcing cost does not change noticeably except the ones with very low outsourcing cost, as shown in Fig. 3.4. Overall, the results indicate that TLH is stable in solving MPRPOS instances with different outsourcing costs.

In summary, results on randomly generated MPRPOS instances show that TLH can quickly provide near-optimal solutions for small-sized instances. For large-sized instances with up to 200 customers, the average gap does not exceed 10% with about 1000s computation time. Note that the size of existing benchmark instances on the well-known NP-hard problem PRP is limited to 200 customers in the literature, and the multi-product MPRPOS is a generalization of the classic single-product PRP. Thus our generated instances are relatively large-scaled instances. In our experiments, we provide feasible solutions and lower bounds to evaluate the developed method with 225 randomly generated MPRPOS instances. Results show the effectiveness and efficiency as well as the stability of our method.

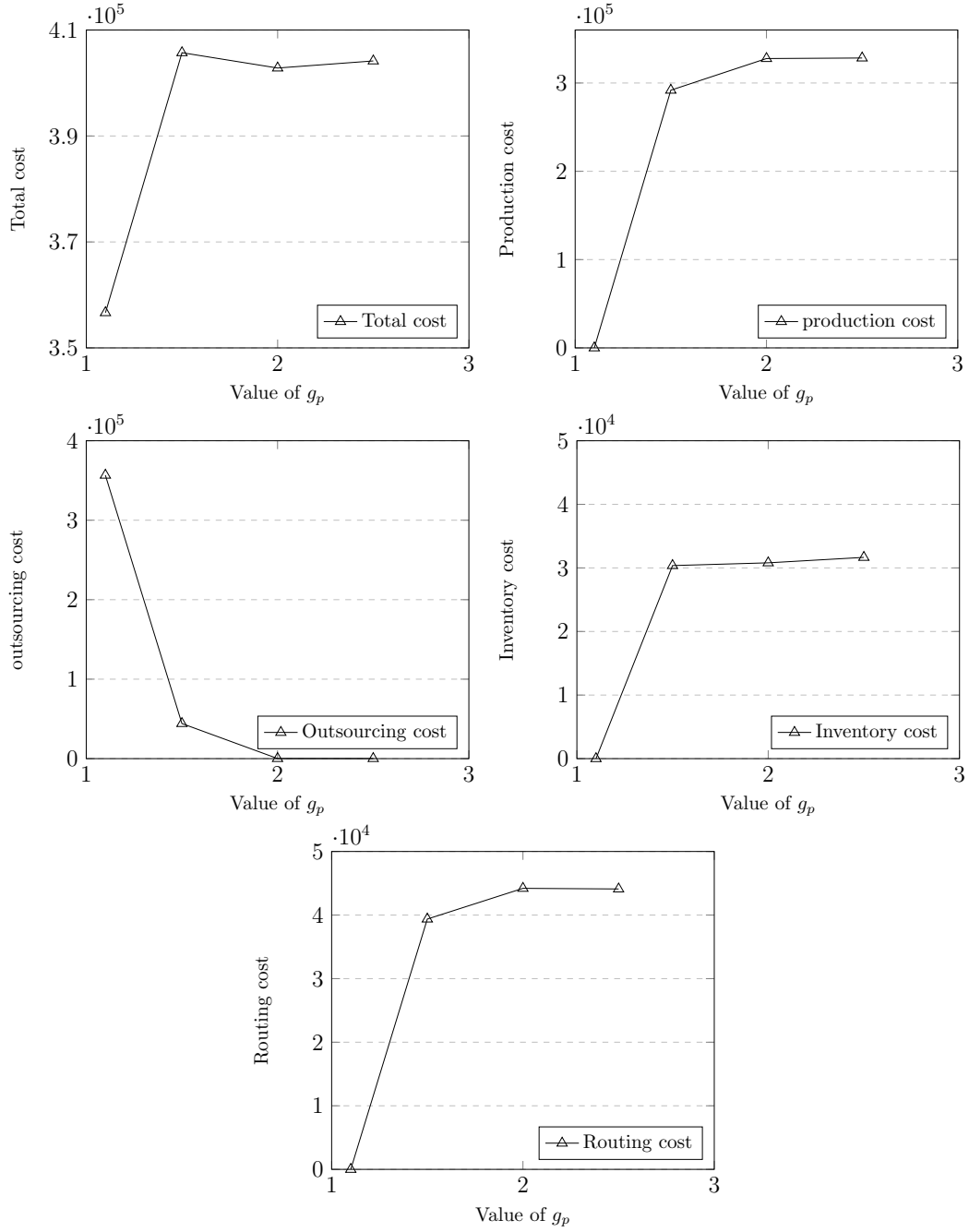



Fig. 3.3: Sensitivity analysis on outsourcing cost

### 3.4.2 Computational experiments for classic PRP

In this subsection, we briefly present the tested benchmark instances, existing algorithms, and their implementation environment. Then, we report the computational results obtained by these algorithms on these benchmark instances in terms of average total cost, number of best solutions found (including the best solutions proposed by TLH), the average gap with respect to the so-far-best-known solutions and average CPU time.

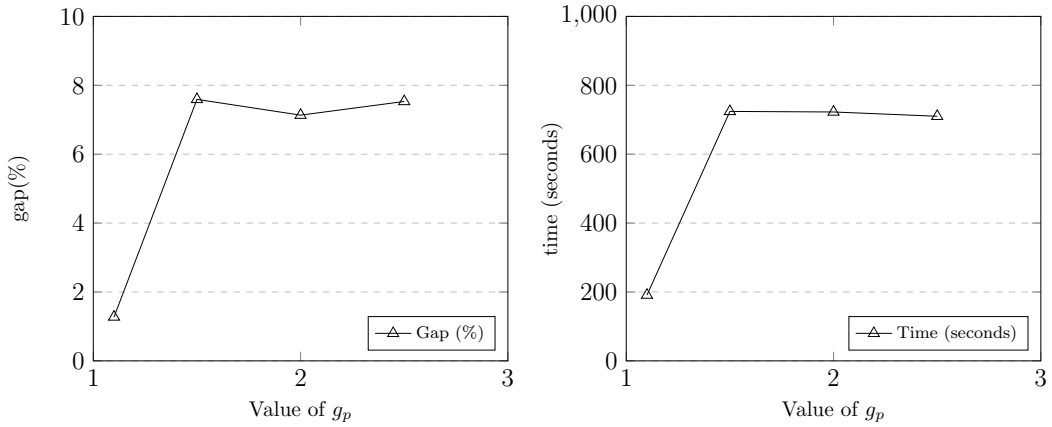



Fig. 3.4: Average gaps and computation times

Table 3.5: Summary of benchmark information

| Instance subset | A1 | A2 | A3 | B1 | B2 | B3 |
|---|---|---|---|---|---|---|
| Number of instances | 480 | 480 | 480 | 30 | 30 | 30 |
| Number of periods | 6 | 6 | 6 | 20 | 20 | 20 |
| Number of customers | 14 | 50 | 100 | 50 | 100 | 200 |
| Number of vehicles | 1 | $\infty$ | $\infty$ | 5 | 9 | 13 |
| Demand | C | C | C | V | V | V |
| Production capacity | $\infty$ | $\infty$ | $\infty$ | C | C | C |
| Plant inventory capacity | $\infty$ | $\infty$ | $\infty$ | C | C | C |
| Customer inventory capacity | C | C | C | C | C | C |
| Initial inventory at plant | 0 | 0 | 0 | V | V | V |
| Initial inventory at customers | V | V | V | 0 | 0 | 0 |
| Vehicle capacity | C | C | C | C | C | C |

*V: Varying, C: Constant, $\infty$: Unlimited [4]

#### 3.4.2.1 Benchmark instances description and existing algorithms implementation

In the literature, algorithm performance comparisons for classic PRP are usually conducted on two sets of widely used benchmark instance sets A and B summarized by [4] (see Table 3.5). Set A (1440 instances) consists of three subsets A1, A2 and A3 with 14, 50 and 100 customers, respectively. Customer demand on 6 periods is assumed to be static. Plant production and inventory capacity are assumed to be unlimited. Vehicle fleet size is set to 1 for A1, and unlimited for A2 and A3. Instances in each subset are further divided into four classes with 120 instances in each class according to their parameter settings. Class 1 corresponds to the instances with standard production, inventory and transportation cost. Class 2 and class 3 are characterized by high production and transportation variable costs, respectively. Class 4 has no inventory cost at customers. The benchmark instance set B (90 instances) is composed of

three subsets B1, B2 and B3 with 50, 100 and 200 customers, respectively. Customer demand on 20 periods is assumed to be dynamic. The production capacity, inventory capacity and vehicle fleet size are assumed to be constant.

Table 3.6: Information of the benchmark algorithms

| Algorithm | Reference | Running platform | Solver | Tested instances |
|---|---|---|---|---|
| ALNS | [4] | 2.10 GHz Duo CPU PC | CPLEX 12.2 | A, B |
| IM-TSP | [1] | 2.67 GHz PC | CPLEX 12.1 | A, B |
| IM-VRP | [1] | 2.67 GHz PC | CPLEX 12.1 | A2, A3, B |
| CCJ-DH | [34] | 3.07 GHz | CPLEX 12.6 | A, B |
| 5P | [92] | 2.4 GHz work station | CPLEX 12.5 | A, B |
| SP-VRP | [88] | 3.5 GHz PC | CPLEX 12.6 | A*, B1 |
| MP-VRP | [88] | 3.5 GHz PC | CPLEX 12.6 | B |
| VNS | [85] | 2.4 GHz PC | CPLEX 12.6 | A, B |
| TLH | This paper | 2.5 GHz PC | CPLEX 12.6 | A, B |

* Only 288 out of the 1440 instances in set A are tested with 96 for each set A1, A2 and A3

#### 3.4.2.2 Comparison of computational results on instance set A

For the set A, a comparison of algorithms on average total cost is presented in Table 3.7, in which the first column and the first row represent the name of instances and algorithms, respectively. Note the results for A1 and A are not available for IM-VRP and SP-VR, respectively. Because SP-VRP tests only a part of set A (288 out of 1440 instances), so its average total cost has not been counted. We can see from Table 3.7 that 5P proposes the best results on A1, and TLH provides the lowest average total cost for A2 and A3. That means TLH has improved the results in the literature on large-sized instances in A.

Table 3.7: Average total cost obtained for set A

| Ins | ALNS | IM-TSP | IM-VRP | CCJ-DH | 5P | SP-VRP | VNS | TLH |
|---|---|---|---|---|---|---|---|---|
| A1 | 181803 | 179424 | - | 179874 | **179314** | - | 179795 | 179490 |
| A2 | 590210 | 589774 | 588183 | 587706 | 587602 | - | 587612 | **587494** |
| A3 | 1089635 | 1090641 | 1086169 | 1086406 | 1084957 | - | 1085368 | **1084860** |

Tables 3.8-3.10 present the number of optimal or best-known solutions obtained for sets A1 - A3 by different algorithms. Particularly, BS and NBS mean the numbers of best solutions and new best solutions obtained by TLH, respectively. Note that [15] provides optimal solutions for 467 out of 480 small-sized instances in set A1. Table 3.8 shows that our heuristic TLH yields optimal solutions for 168 small-sized instances in set A1. It outperforms ALNS, CCJ-DH and VNS, while IM-TSP and 5P obtain optimal solutions for more instances. Compared with existing algorithms, TLH has

difficulties in finding optimal solutions for set A1 even with 200 iterations in the first level. This may be because the diversification mechanism in Level 1 is relatively simple. From Table 3.9, we see that TLH obtains 114 best solutions including 110 new best solutions for medium-sized instance set A2. Table 3.10 shows that TLH clearly outperforms all other algorithms. It provides 166 new best solutions among 167 best solutions found. Numerical results show that our TLH outperforms existing heuristics on medium- and large-sized instances in terms of the number of best solutions found.

Table 3.8: Number of best solutions found for small-sized instance set A1

| Classes | ALNS | IM-TSP | IM-VRP | CCJ-DH | 5P | SP-VRP | VNS | TLH | |
|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | BS | NBS |
| Class 1 | 1 | **78** | - | 10 | 71 | - | 30 | 45 | 0 |
| Class 2 | 0 | **78** | - | 10 | 71 | - | 33 | 45 | 0 |
| Class 3 | 0 | 52 | - | 4 | **67** | - | 20 | 35 | 0 |
| Class 4 | 1 | **85** | - | 20 | 70 | - | 27 | 43 | 0 |
| Total | 2 | **293** | - | 44 | 279 | - | 110 | 168 | 0 |

Table 3.9: Number of best solutions found medium-sized instance set A2

| Classes | ALNS | IM-TSP | IM-VRP | CCJ-DH | 5P | SP-VRP | VNS | TLH | |
|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | BS | NBS |
| Class 1 | 0 | 1 | 3 | **52** | 19 | 3 | 18 | 27 | **24** |
| Class 2 | 0 | 7 | 5 | 18 | 37 | 11 | 4 | **38** | **38** |
| Class 3 | 0 | 2 | 1 | **43** | 28 | 5 | 18 | 24 | **23** |
| Class 4 | 1 | 6 | 9 | **30** | 21 | 9 | 20 | 25 | **25** |
| Total | 1 | 16 | 18 | **143** | 105 | 28 | 60 | 114 | **110** |

Table 3.10: Number of best solutions found for large-sized instance set A3

| Classes | ALNS | IM-TSP | IM-VRP | CCJ-DH | 5P | SP-VRP | VNS | TLH | |
|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | BS | NBS |
| Class 1 | 0 | 0 | 6 | 31 | 26 | - | 8 | **47** | **47** |
| Class 2 | 0 | 2 | 13 | 27 | 27 | - | 6 | **45** | **44** |
| Class 3 | 0 | 0 | 1 | 11 | 37 | - | 15 | **43** | **43** |
| Class 4 | 0 | 2 | 13 | 22 | 23 | - | 14 | **32** | **32** |
| Total | 0 | 4 | 33 | 91 | 113 | - | 43 | **167** | **166** |

Tables 3.11-3.13 present the average gap with respect to the best solutions found so far for sets A1-A3 by different algorithms. Looking at Table 3.11, the average gap of TLH is 0.26% for small-sized instances in set A1 while IM-TSP and 5P achieve lower gaps of 0.18% and 0.06%, respectively. As shown in Tables 3.12 and 3.13, TLH achieves the smallest average gap among all algorithms for medium- and large-sized instances. Especially, TLH provides 0.14% and 0.10% average gaps for sets A2 and A3, respectively. The results demonstrate that TLH is more effective for

medium- and large-sized instances in terms of average gap. Similar to the benchmark algorithms, TLH generally yields larger gaps on the third class instances with high transportation cost. One possible reason is that SLS has difficulty in generating an optimal delivery plan due to the inclusion of the approximate visit cost which is to some extent inaccurate.

Table 3.11: Average gap with respect to the best solutions for small-sized instance set A1

| Classes | ALNS(%) | IM-TSP(%) | IM-VRP(%) | CCJ-DH(%) | 5P(%) | SP-VRP(%) | VNS(%) | TLH(%) |
|---|---|---|---|---|---|---|---|---|
| Class 1 | 1.7 | 0.09 | - | 0.47 | **0.03** | - | 0.28 | 0.15 |
| Class 2 | 0.36 | 0.01 | - | 0.08 | **0.00** | - | 0.05 | 0.03 |
| Class 3 | 8.43 | 0.57 | - | 2.20 | **0.18** | - | 1.52 | 0.76 |
| Class 4 | 0.95 | **0.03** | - | 0.25 | 0.05 | - | 0.57 | 0.09 |
| Average | 2.86 | 0.18 | - | 0.75 | **0.06** | - | 0.61 | 0.26 |

Table 3.12: Average gap with respect to the best solutions for medium-sized instance set A2

| Classes | ALNS(%) | IM-TSP(%) | IM-VRP(%) | CCJ-DH(%) | 5P(%) | SP-VRP(%) | VNS(%) | TLH(%) |
|---|---|---|---|---|---|---|---|---|
| Class 1 | 1.21 | 1.16 | 0.25 | 0.17 | 0.13 | - | 0.13 | **0.07** |
| Class 2 | 0.18 | 0.11 | 0.06 | 0.04 | **0.02** | - | 0.05 | **0.02** |
| Class 3 | 3.93 | 3.11 | 1.39 | 0.72 | 0.48 | - | 0.43 | **0.39** |
| Class 4 | 0.26 | 0.53 | 0.17 | 0.07 | 0.13 | - | **0.05** | 0.09 |
| Average | 1.40 | 1.23 | 0.47 | 0.25 | 0.19 | - | 0.16 | **0.14** |

Table 3.13: Average gap with respect to the best solutions for large-sized instance set A3

| Classes | ALNS(%) | IM-TSP(%) | IM-VRP(%) | CCJ-DH(%) | 5P(%) | SP-VRP(%) | VNS(%) | TLH(%) |
|---|---|---|---|---|---|---|---|---|
| Class 1 | 1.02 | 1.88 | 0.24 | 0.28 | 0.09 | - | 0.19 | **0.04** |
| Class 2 | 0.14 | 0.12 | 0.03 | 0.03 | 0.02 | - | 0.06 | **0.01** |
| Class 3 | 3.82 | 3.94 | 1.36 | 1.73 | **0.28** | - | 0.44 | 0.29 |
| Class 4 | 0.33 | 0.70 | 0.24 | 0.06 | 0.05 | - | 0.08 | **0.04** |
| Average | 1.33 | 1.66 | 0.47 | 0.52 | 0.11 | - | 0.19 | **0.10** |

Tables 3.14-3.16 present the average CPU times in seconds for sets A1-A3. These Tables show that all algorithms are relatively efficient on instances set A. Note that as the implementation and running environment of each algorithm is different, it is difficult to compare the computation time. Thus the following comparison is relatively rough. Table 3.14 reports the average computation time for set A1. It indicates that TLH is comparable to ALNS, CCJ-DH, 5P and VNS, which consume generally less than 30 seconds on A1. While IM-TSP needs more than 200 seconds. Table 14 for set A2 shows that TLH is much faster than CCJ-DH and IM-TSP, comparable to ALNS, and slightly slower than IM-VRP, 5P and VNS, although it is more effective than 5P and IM-VRP in terms of number of best solutions found and the average

gap. Table 3.15 indicates that for set A2, the computation times of IM-TSP, CCJ-DH and SP-VRP are generally longer, which exceed 300s, 200s and 600s, respectively. Computation times of ALNS, 5P, VNS and TLH are comparable, needing less than 50s. Moving to large-sized instances in set A3, Table 3.16 shows that TLH is relatively fast among the existing algorithms. VNS provides the best CPU time for large-sized instance set A3, but it performs badly in term of number of best solutions found and average gap.

In summary, TLH can find high-quality solutions within short computation time. Especially, TLH has found 276 new best solutions for instance set A.

Table 3.14: Average CPU times in seconds for small-sized instance set A1

| Classes | ALNS | IM-TSP | IM-VRP | CCJ-DH | 5P | SP-VRP | VNS | TLH |
|---|---|---|---|---|---|---|---|---|
| Class 1 | 9.2 | 251 | - | 14.4 | **5.0** | - | 12.3 | 29.3 |
| Class 2 | 8.9 | 214.2 | - | 13.6 | **4.9** | - | 12.8 | 27.8 |
| Class 3 | 7.6 | 237.2 | - | 12.8 | **4.7** | - | 12.2 | 28.0 |
| Class 4 | 8.7 | 216.9 | - | 14.9 | **5.2** | - | 12.6 | 25.1 |
| Average | 8.6 | 229.8 | - | 13.9 | **5.0** | - | 12.5 | 27.5 |

Table 3.15: Average CPU times in seconds for medium-sized instance set A2

| Classes | ALNS | IM-TSP | IM-VRP | CCJ-DH | 5P | SP-VRP | VNS | TLH |
|---|---|---|---|---|---|---|---|---|
| Class 1 | 50.2 | 338.5 | 25.6 | 328.8 | **16.4** | 736.8 | 25.2 | 43.1 |
| Class 2 | 49.5 | 235.7 | 21.7 | 272.8 | **13.8** | 661.2 | 21.3 | 36.5 |
| Class 3 | 42.7 | 317.9 | 22.6 | 252.6 | **15.8** | 699.0 | 24.5 | 39.3 |
| Class 4 | 44.1 | 375.8 | 27.7 | 271.5 | **25.4** | 466.8 | 28.4 | 42.1 |
| Average | 46.6 | 317.0 | 24.4 | 281.4 | **17.9** | 641 | 24.9 | 40.3 |

Table 3.16: Average CPU times in seconds for large-sized instance set A3

| Classes | ALNS | IM-TSP | IM-VRP | CCJ-DH | 5P | SP-VRP | VNS | TLH |
|---|---|---|---|---|---|---|---|---|
| Class 1 | 228.5 | 514.2 | 85.5 | 1313.1 | 323.5 | 2184.6 | **84.6** | 168.6 |
| Class 2 | 217.6 | 497.4 | 76.1 | 1062.7 | 50.6 | 2082.6 | **73.4** | 131.9 |
| Class 3 | 197.8 | 509.3 | 75.1 | 1016.9 | 349.6 | 2177.4 | **72.8** | 144.5 |
| Class 4 | 206.0 | 507.0 | 86.0 | 1056.6 | 125.1 | 2016.6 | **83.7** | 159.8 |
| Average | 212.5 | 506.2 | 80.7 | 1112.3 | 212.2 | 2115.3 | **78.6** | 151.2 |

#### 3.4.2.3 Comparison of computational results on instance set B

In this section, we summarize and compare the computational results on large-sized instance set B. Table 3.17 presents the average total costs obtained by different algorithms. It reveals that our TLH yields the better average total cost than ALNS, IM-TSP, IM-VRP, CCJ-DH, 5P, and VNS. While it is a little inferior to SP+MP-VRP. Tables 3.18 and 3.19 show that TLH provides new best solutions to 7 instances. In addition, Table 3.20 shows that TLH has an absolute advantage in computation

time. TLH needs only about 460s average computation time, while existing algorithms require about 2200s - 10100s. In detail, TLH needs only about 280s on B1 while CCJ-DH and 5P take more than 2000s and 3000s, respectively. For the largest-sized instance set B3, TLH consumes less than 700s, but existing algorithms require 4197s - 19270s.

Table 3.17: Average total costs for very large-sized instance set B

| Instance | ALNS | IM-TSP | IM-VRP | CCJ-DH | 5P | SP+MP-VRP | VNS | TLH |
|---|---|---|---|---|---|---|---|---|
| B1 | 346878 | 348175 | 348990 | 346186 | 343439 | **340850** | 345758 | 344295 |
| B2 | 636962 | 639272 | 636294 | 637826 | 631351 | **629931** | 634776 | 630339 |
| B3 | 876761 | 879447 | 865905 | 862872 | 882128 | **846431** | 864523 | 857207 |

Table 3.18: Number of best solutions for very large-sized instance set B

| Classes | ALNS | IM-TSP | IM-VRP | CCJ-DH | 5P | SP+MP-VRP | VNS | TLH | |
|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | BS | NBS |
| B1 | 0 | 0 | 0 | 0 | 3 | **27** | 0 | 0 | 0 |
| B2 | 1 | 1 | 0 | 0 | 7 | **14** | 0 | 7 | 7 |
| B3 | 0 | 0 | 0 | 0 | 0 | **30** | 0 | 0 | 0 |
| total | 1 | 1 | 0 | 0 | 10 | **71** | | 7 | 7 |

Table 3.19: Average gap with respect to the best solutions for very large-sized instance set B

| Instance | ALNS(%) | IM-TSP(%) | IM-VRP(%) | CCJ-DH(%) | 5P(%) | SP+MP-VRP(%) | VNS(%) | TLH(%) |
|---|---|---|---|---|---|---|---|---|
| B1 | 1.78 | 2.16 | 2.40 | 1.58 | 0.77 | **0.01** | 1.46 | 1.03 |
| B2 | 1.36 | 1.73 | 1.26 | 1.50 | 0.47 | **0.25** | 1.02 | 0.31 |
| B3 | 3.58 | 3.91 | 2.30 | 1.94 | 4.23 | **0.00** | 2.14 | 1.27 |
| Average | 2.24 | 2.60 | 1.99 | 1.68 | 1.82 | **0.09** | 1.54 | 0.87 |

In summary, TLH generally outperforms most of the existing heuristics on for the large-sized instance set B. Although it is a little inferior to SP+MP-VRP in terms of the average solution quality, it is the fastest among all heuristics and it needs only 6% of the time consumed by SP+MP-VRP. Particularly, it contributes new best solutions for 7 out of 90 instances. The excellent performance of TLH on these instances may due to three aspects: 1) partial fixed production setup schedule in Level 1; 2) infeasibility repair procedure only in Level 2; 3) further improvement obtained by the fix-and-optimize procedure in Level 3.

### 3.4.3 Performance analysis of TLH

In this subsection, we analyze in detail the performance of TLH. The first two levels of TLH provide a feasible solution to MPRPOS and the third level further improves the incumbent solution by a fix-and-optimize procedure. In Table 3.21, the improvements

Table 3.20: Average CPU times in seconds for very large-sized instance set B

| Instance | ALNS | IM-TSP | IM-VRP | CCJ-DH | 5P | SP-VRP | MP-VRP | VNS | TLH |
|---|---|---|---|---|---|---|---|---|---|
| B1 | 481 | 1653 | 551 | 978 | 2464 | 2140 | 3796 | 488 | **277** |
| B2 | 1570 | 9484 | 2054 | 5441 | 7488 | - | 7510 | 1767 | **493** |
| B3 | 5794 | 19270 | 4197 | 13693 | 16365 | - | 10727 | 5884 | **614** |
| Average | 2615 | 10136 | 2267 | 6704 | 8772 | - | 7344 | 2713 | **461** |

obtained by Level 3 and the corresponding computation time are reported. The first two columns denote the instance sets and the number of customers. Columns "gap2" and "gap3" indicate the gaps of the objective value obtained by Level 2 and Level 3 with respect to the best-known solutions. Column "Improve" presents the improvements obtained by Level 3, which can be calculated as: Improve=gap2−gap3. "T2" denotes the computation time at the end of Level 2 and "T3" is the computation time in Level 3. From Table 3.21, we can see that good feasible solutions are already obtained at the end of Level 2, and the average gaps are 0.22%, 1.11%, and 0.20% for instance sets A, B and C, respectively. 0.05%, 0.24% and 0.20% average improvements for the three sets can be achieved by Level 3, which shows that the fix-and-optimize procedure is effective. Especially, "0.00" in column "gap3" means the best solutions are obtained by TLH. In terms of computation time, the first two levels together generally take longer time than the third level and the computation times increase with the number of customers. In particular, the computation time increases with the number of customers more quickly in Level 3, since the more customers needs, the more iterations are needed in the fix-and-optimize procedure.

Table 3.21: Improvements of the fix-and-optimize procedure

| Instance | $n$ | gap2(%) | gap3(%) | Improve(%) | T2(s) | T3(s) |
|---|---|---|---|---|---|---|
| | 14 | 0.33 | 0.26 | 0.07 | 27.3 | 0.2 |
| A | 50 | 0.19 | 0.14 | 0.05 | 36.3 | 4.0 |
| | 100 | 0.13 | 0.10 | 0.03 | 118.2 | 33.0 |
| Average | | 0.22 | 0.17 | 0.05 | 60.6 | 12.4 |
| | 50 | 1.46 | 1.03 | 0.43 | 270.9 | 6.0 |
| B | 100 | 0.45 | 0.31 | 0.14 | 456.2 | 36.8 |
| | 200 | 1.42 | 1.27 | 0.15 | 396.0 | 218.0 |
| Average | | 1.11 | 0.87 | 0.24 | 374.3 | 86.9 |
| | 10 | 0.14 | 0.00 | 0.14 | 95.6 | 0.4 |
| | 14 | 0.15 | 0.00 | 0.15 | 100.4 | 0.7 |
| C | 50 | 0.17 | 0.00 | 0.17 | 419.0 | 11.5 |
| | 100 | 0.12 | 0.00 | 0.12 | 505.8 | 56.6 |
| | 200 | 0.43 | 0.00 | 0.43 | 746.6 | 381.0 |
| Average | | 0.20 | 0.00 | 0.20 | 373.5 | 90.0 |

To evaluate the effectiveness of the new update mechanism of the approximate visit cost in Level 1 of TLH, we compare it with the update mechanism proposed by [1]. To do this, the update mechanism of [1] is implemented in our TLH to form

a new algorithm called TLHA that is tested on the newly proposed instances and the benchmarks. The comparison results between TLH and TLHA are presented in Table 3.22, in which "diff2" denotes the differences between the objective values of Level 2 obtained by TLH and TLHA. Similarly, " diff3 " denotes the differences between the objective values of the final solution provided by TLH and TLHA. Let $obj^2$ and $obj^*$ denote the objcetive values provided by the second-level solution and the final solution, respectively, then "diff2" and "diff3" can be calculated using equations (3.45) and (3.46), respectively. T2 and T3 denote the computation times at the end of Level 2 and Level 3, respectively. It is shown in Table 3.22 that our newly proposed update mechanism generally provides better results for instance sets A and C, while providing slightly inferior results for instance set B compared with that proposed by [1]. The average differences between two mechanisms at the end of Level 2 are 0.25%, -0.03%, and 0.24% for set A, B and C, respectively. These differences do not change noticeably during Level 3, with the final differences being 0.25%, -0.03%, and 0.21%, respectively. In terms of the computation time, TLH generally takes longer time than TLHA. Overall, the results in Table 3.22 indicates that the newly proposed update mechanism is effective for most of the tested instances.

$$\text{diff2} = \frac{\text{obj}^2(\text{TLHA}) - \text{obj}^2(\text{TLH})}{\text{obj}^2(\text{TLH})} \times 100\% \tag{3.45}$$

$$\text{diff3} = \frac{\text{obj}^*(\text{TLHA}) - \text{obj}^*(\text{TLH})}{\text{obj}^*(\text{TLH})} \times 100\% \tag{3.46}$$

Table 3.22: omparison of different visit cost update mechanisms

| Instance | $n$ | dif2(%) | dif3(%) | T2(s) | | T3(s) | |
|---|---|---|---|---|---|---|---|
| | | | | TLHA | TLH | TLHA | TLH |
| A | 14 | 0.10 | 0.12 | 28.8 | 27.3 | 29.0 | 27.5 |
| | 50 | 0.25 | 0.29 | 32.6 | 36.3 | 35.4 | 40.3 |
| | 100 | 0.39 | 0.33 | 93.5 | 118.2 | 114.1 | 151.2 |
| Average | | 0.25 | 0.25 | 51.6 | 60.6 | 59.5 | 73.0 |
| B | 50 | -0.18 | -0.14 | 149.0 | 270.9 | 154.1 | 276.8 |
| | 100 | -0.00 | -0.01 | 220.1 | 456.2 | 250.7 | 493.0 |
| | 200 | 0.10 | 0.06 | 338.3 | 396.0 | 547.7 | 614.0 |
| Average | | -0.03 | -0.03 | 235.8 | 374.3 | 317.5 | 461.3 |
| C | 10 | 0.89 | 0.74 | 92.7 | 95.6 | 93.2 | 96.1 |
| | 14 | 0.38 | 0.33 | 112.0 | 100.4 | 112.5 | 101.1 |
| | 50 | 0.39 | 0.32 | 314.8 | 419.0 | 323.4 | 430.5 |
| | 100 | 0.07 | 0.03 | 431.6 | 505.8 | 475.7 | 562.4 |
| | 200 | -0.53 | -0.40 | 670.2 | 746.6 | 966.7 | 1127.6 |
| Average | | 0.24 | 0.21 | 324.3 | 373.5 | 394.3 | 463.5 |

To sum up, the good performance of TLH is a joint effect of the three levels. Firstly, Level 1 develops a two-phase iterative method and provides a good feasible

solution or useful information to be used in Level 2. It generally consumes longer computation time in solving the specific lot-sizing problem (SLS in Fig. 1) by CPLEX, yet the computation time can be reduced by allowing infeasibility and fixing the production setup schedule for most iterations when solving SLS. Secondly, Level 2 introduces a restricted production distribution model (RPD in Fig. 1) to repair the infeasible solutions from Level 1 and further improves them with a simple procedure, thus the computation time in Level 2 is very short. Finally, Level 3 explores further improvement by applying a fix-and-optimize procedure on each customer. Accordingly, a higher improvement is achieved and the computation time increases with the number of customers.

## 3.5 Conclusions

This chapter investigates a new multi-product production routing problem with outsourcing (MPRPOS) that is a generalization of the classic PRP. Firstly, a mixed integer linear program is formulated for the problem. Then a three-level mathematical-programming-based heuristic (TLH) is developed to solve both MPRPOS and the classic PRP. In TLH, the original MPRPOS is decomposed into two subproblems that are solved iteratively to generate an initial solution (probably infeasible); then a restricted production direct-distribution problem is solved to repair the infeasible solution and a route consolidation procedure further improves the incumbent solution; finally, a fix-and-optimize procedure is used to improve the incumbent solution iteratively. To evaluate the performance of TLH, 225 newly generated MPRPOS instances with up to 200 customers, 20 vehicles, 6 periods, and 12 products are first tested, followed by the tests on 1530 widely used PRP benchmark instances with up to 200 customers, 13 vehicles, and 20 periods. Computational results on MPRPOS instances show that TLH can efficiently find feasible solutions with average gaps of 1.99% and 9.90% with computation times of 98.6s and 706.8s for instances with up to 14 customers and 200 customers, respectively. Extensive experimental results on PRP benchmark instances indicate that TLH generally outperforms most of the existing heuristics for PRP in terms of both solution quality and computation time. In particular, TLH finds 283 new best solutions for 1530 tested benchmark instances. The corresponding work has been published in the following paper.

Y. Li, F. Chu, C. Chu, and Z. Zhu. An Efficient Three-level Heuristic for the Large-scaled Multi-product Production Routing Problem with Outsourcing. *European Journal of Operational Research*, 272(3):914-927, 2019.

DOI: https://doi.org/10.1016/j.ejor.2018.07.018.

# Chapter 4

# Multi-plant Food Production Routing Problem with Packaging Consideration

## 4.1 Introduction

A FSC has some particular characteristics compared to general supply chains. Especially, product perishability plays an important role and impacts the overall performance of FSCs. Precisely, the limited shelf life of food products can significantly affect the production, inventory and transportation activities. A longer product shelf life means a better quality. As the residual shelf life reduces, the product quality deteriorates. Innovative food packaging (preservation), such as antimicrobial packaging methods, may be more expensive, but can extend the product shelf life by slowing down the decay rate. Integrating such packaging decisions in the food production routing optimization is important because it can in turn influence production, inventory, and routing decisions. In this chapter, we address a novel multi-plant food production routing problem (MFPRP) that considers food perishability and packaging simultaneously. The problem, which is an extension of the classic PRP, consists of determining production, inventory and distribution planning for each plant on a time horizon. The objective is to maximize the total profit that is equal to the selling revenue of all retailers minus the total production, packaging, inventory, and transportation costs. MFPRP is firstly formulated as a MILP. Then it is solved by a hybrid matheuristic (HM) that has three components: 1) a two-phase iterative method (TI) method to obtain a good initial solution or useful information; 2) a fix-and-optimize (FO) procedure to repair infeasibility and to improve the solution; and 3) a route-based optimization (RO) to further improve the incumbent solution by exploiting the useful information provided by TI and FO. Finally, HM is evaluated by computational experiments on 320 randomly generated instances.

The remainder of this chapter is organized as follows. Section 4.2 presents the problem description and mathematical formulation. In Section 4.3, a hybrid matheuristic is developed. Computational experiments are conducted in Section 4.4. Finally, Section 4.5 concludes this chapter.

## 4.2 Problem description and formulation

The considered MFPRP is defined on a complete digraph $G$=$\{N,\ A\}$. The set $N$ of nodes is comprised of a subset $P = \{1, 2, \ldots, m\}$ of $m$ plants and a subset $R = \{m + 1, m + 2, \ldots, m + n\}$ of $n$ retailers, i.e., $N = P \cup R = \{1, 2, \ldots, m, m + 1, \ldots, m + n\}$. The set $A$ of arcs links all nodes, i.e., $A$=$\{(i, j) : i, j \in N, i \neq j\}$. Each plant $p \in P$ has a limited production capacity $C_p$ and a fleet $K_p = \{1, 2, \ldots, |K_p|\}$ of homogeneous vehicles with capacity $V$. Within a planing horizon $T = \{0, 1, \ldots, |T|\}$, the set of plants are responsible for producing and distributing a single perishable product to satisfy the dynamic retailer demand $D_{it}$ ($i \in R$, $t \in T$). Both plants and retailers have a storage capacity $U_i$ where $i \in N$. The retailers' demand has to be met in time, i.e., backlogging is not allowed. A set $B = \{1, 2, \ldots, |B|\}$ of packages that will lead to different product shelf lives exist and a packaged product with the package $u \in B$ has a shelf life $s = \theta(u)$. A residual shelf of a product indicates the number of time periods the product can be stored after it is packaged. Note that the package types are ordered by their quality such that $\theta(1) < \theta(2) <, \ldots, < \theta(|B|)$. Correspondingly, package $u \in B$ incurs a packaging cost $e_u$ such that $e_1 < e_2 <, \ldots, < e_{|B|}$, i.e., a better package with higher packaging cost leads to a longer shelf life. The shelf life of a product varies in the set $S = \{0, 1, \ldots, |S|\}$, where a shelf life 0 means that the product should be consumed in the current period and cannot be stored to meet future demand, and $|S|$ may take a value of up to $\theta(|B|)$, i.e., the shelf life when using the best package.

The MFPRP is studied under the following assumptions: 1) products with different packages have different shelf lives; 2) the final price of a product is dependent on its residual shelf life; 3) the customers' demand can be satisfied by products with different residual shelf lives and packages; 4) each vehicle's route starts and ends at the same plant; 5) each vehicle can perform at most one trip in each period; and 6) each retailer can be visited at most once in a period, i.e., split delivery is not allowed.

A solution to the problem consists of determining: 1) production quantity in each period in each plant; 2) package selection for each produced product; 3) delivery date and quantity to each customer from the plants in each period; 4) vehicle routing; and 5) selling decisions to satisfy customers' demand. The objective is to maximize the total profit, which is equal to the total selling revenue minus the fixed and variable

production, packaging, inventory, and routing costs. To formulate the problem, the following notation is defined:

Indices:

$i, j, p$: index of a node, $i, j, p \in N$;
$t$: index of a period, $t \in T$;
$k$: index of a vehicle, $k \in K_p, p \in P$;
$s$: index of a shelf life, $s \in S$;
$u$: index of a package type, $u \in B$.

Parameters:

$D_{it}$: demand of retailer $i \in R$ in period $t \in T$;
$a_p$: unit production cost at plant $p \in P$;
$b_p$: production set up cost at plant $p \in P$;
$e_u$: unit packaging cost for package $u \in B$;
$g_{is}$: selling price for a unit of a product with residual shelf life $s \in S$ at retailer $i \in R$;
$C_p$: production capacity of plant $p \in P$;
$U_i$: inventory capacity of node $i \in N$;
$h_i$: unit inventory holding cost per period at node $i \in N$;
$I_{is0}$: initial inventory of product with residual shelf life $s \in S$ at node $i \in N$;
$V$: vehicle capacity;
$c_{ij}$: travel cost on arc $(i, j) \in A$;
$|K_p|$: fleet size at plant $p \in P$;
$\theta(u)$: shelf life of a newly produced product with package $u \in B$;
$Q_t$: a big number equal to $Q_t = \sum_{i \in R} \sum_{t'=t}^{|T|} D_{it'}$;
$W_{it}$: a big number equal to $W_{it} = \min(U_i + D_{it}, \sum_{t'=t}^{|T|} D_{it'})$.

Decision variables:

$\xi_{pt}$: production quantity in period $t$ at plant $p$;
$z_{p,\theta(u),t}$: packaging quantity with shelf life $\theta(u)$ using package type $u$ in period $t$ at plant $p$;
$w_{pt}$: equal to 1 if there is any production in period $t$ at plant $p$, and 0 otherwise;
$I_{ist}$: inventory of product with residual shelf life $s$ at node $i$ at the end of period $t$;
$\vartheta_{ist}$: quantity of product with residual shelf life $s$ used to meet the demand at retailer $i$ in period $t$;
$y_{piskt}$: delivery quantity of product with residual shelf life $s$ from plant $p$ to retailer $i$ by vehicle $k$ in period $t$;
$v_{pikt}$: equal to 1 if retailer $i$ is visited by vehicle $k$ from plant $p$ in period $t$, and 0 otherwise;

$x_{ijkt}$: equal to 1 if arc $(i,j)$ is traversed by vehicle $k$ in period $t$, and 0 otherwise.

The proposed problem can be formulated as follows (model $\mathcal{P}$):

$$\mathcal{P}: \max \sum_{i\in R}\sum_{s\in S}\sum_{t\in T} g_{is}\vartheta_{ist} - \sum_{p\in P}\sum_{t\in T}(a_{pt}\xi_{pt} + b_{pt}w_{pt}) - \sum_{p\in P}\sum_{u\in B}\sum_{t\in T} e_u z_{p,\theta(u),t}$$
$$- \sum_{i\in N}\sum_{s\in S}\sum_{t\in T} h_i I_{ist} - \sum_{p\in P}\sum_{(i,j)\in A}\sum_{k\in K_p}\sum_{t\in T} c_{ij}x_{ijkt} \tag{4.1}$$

s.t.

$$I_{pst} = I_{p,s+1,t-1} + z_{p,\theta(u),t} - \sum_{i\in R}\sum_{k\in K_p} y_{piskt},$$
$$\forall\, p\in P, s\in\{\theta(u)|u\in B\}, t\in T \tag{4.2}$$

$$I_{pst} = I_{p,s+1,t-1} - \sum_{i\in R}\sum_{k\in K_p} y_{piskt},\ \forall\, p\in P, s\in S\setminus\{\theta(u)|u\in B\}, t\in T \tag{4.3}$$

$$I_{ist} = I_{i,s+1,t-1} + \sum_{p\in P}\sum_{k\in K_p} y_{piskt} - \vartheta_{ist},\ \forall\, i\in R, s\in S, t\in T \tag{4.4}$$

$$\sum_{u\in B} z_{p,\theta(u),t} = \xi_{pt},\ \forall\, p\in P, t\in T \tag{4.5}$$

$$\xi_{pt} \le C_p w_{pt},\ \forall\, p\in P, t\in T \tag{4.6}$$

$$\sum_{p\in P}\xi_{pt} \le Q_t,\ \forall\, t\in T \tag{4.7}$$

$$\sum_{s\in S} I_{ist} \le U_i,\ \forall i\in N, t\in T \tag{4.8}$$

$$\sum_{s\in S}\vartheta_{ist} = D_{it},\ \forall\, i\in R, t\in T \tag{4.9}$$

$$\sum_{i\in R}\sum_{s\in S} y_{piskt} \le V,\ \forall\, p\in P, k\in K_p, t\in T \tag{4.10}$$

$$\sum_{s\in S} y_{piskt} \le W_{it}v_{pikt},\ \forall\, p\in P, i\in R, k\in K_p, t\in T \tag{4.11}$$

$$\sum_{p\in P}\sum_{k\in K_p} v_{pikt} \le 1,\ \forall\, i\in R, t\in T \tag{4.12}$$

$$\sum_{j\in N\setminus\{i\}} x_{ijkt} = \sum_{j\in N\setminus\{i\}} x_{jikt},\ \forall\, p\in P, i\in N, k\in K_p, t\in T \tag{4.13}$$

$$\sum_{j\in R} x_{pjkt} + \sum_{j\in N\setminus\{i\}} x_{jikt} \ge 2v_{pikt},\ \forall\, p\in P, i\in R, k\in K_p, t\in T \tag{4.14}$$

$$\sum_{j\in N\setminus\{i\}} x_{ijkt} = \sum_{p\in P} v_{pikt},\ \forall\, i\in R, k\in K_p, t\in T \tag{4.15}$$

$$\sum_{p\in P}\sum_{i\in R} x_{pikt} \le 1,\ \forall\, k\in K_p, t\in T \tag{4.16}$$

$$x_{ijkt} = 0,\ \forall\, i\in P, j\in P, k\in K, t\in T \tag{4.17}$$

$$\sum_{i\in R} x_{pikt} = 0,\ \forall\, p\in P, t\in T, k\in K_{p'}, p'\in P\setminus\{p\}, \tag{4.18}$$

$$\sum_{i\in H}\sum_{j\in H\setminus\{i\}} x_{ijkt} \leq |H|-1,\ \forall H\subseteq R, |H|\geq 2, p\in P, k\in K_p, t\in T \tag{4.19}$$

$$w_{pt} \in \{0,1\},\ \forall p\in P, t\in T \tag{4.20}$$

$$\xi_{pt} \geq 0,\ \forall p\in P, t\in T \tag{4.21}$$

$$z_{p,\theta(u),t} \geq 0,\ \forall p\in P, u\in B, t\in T \tag{4.22}$$

$$\vartheta_{ist} \geq 0,\ \forall i\in R, s\in S, t\in T \tag{4.23}$$

$$I_{ist} \geq 0,\ \forall i\in N, s\in S, t\in T \tag{4.24}$$

$$y_{piskt} \geq 0,\ \forall p\in P, i\in R, s\in S, k\in K_p, t\in T \tag{4.25}$$

$$v_{pikt} \in \{0,1\},\ \forall p\in P, i\in R, k\in K_p, t\in T \tag{4.26}$$

$$x_{ijkt} \in \{0,1\},\ \forall (i,j)\in A, k\in K_p, t\in T. \tag{4.27}$$

The objective function (4.1) maximizes the total profit, i.e., the total revenue minus the total production, packaging, inventory and routing costs. Constraints (4.2)-(4.4) indicate the inventory flow balance for products with different residual shelf lives at each plant and retailer, where $I_{i,|S|+1,t}=0$. Constraints (4.5) represent the flow conservation of the produced and packaged products. Constraints (4.6) and (4.7) mean that the production quantity at a plant cannot exceed its capacity if the plant is set for production, in particular, the total production quantity of all plants should be limited to the total remaining demand of all retailers. Constraints (4.8) indicate that the inventory at each plant and retailer cannot exceed their inventory capacities. Constraints (4.9) indicate that the customer demand must be met, and it can be met by products with different residual shelf lives. Constraints (4.10) mean that a vehicle cannot deliver more than its capacity. Constraints (4.11) allow positive delivery quantity by a vehicle originating from a plant to a retailer only if the retailer is visited by that vehicle. Constraints (4.12) forbid split delivery. Constraints (4.13) correspond to the vehicle flow conservation. Constraints (4.14) and (4.15) link the arc routing variables to the delivery schedule. Particularly, if a retailer $i$ is visited by vehicle $k$ from plant $p$ in period $t$, then vehicle $k$ must depart from plant $p$ and traverse an arc linked to node $i$. Constraints (4.16) state that a vehicle can perform at most one route in a period. Constraints (4.17) indicate that vehicles cannot travel from one plant to another. Constraints (4.18) mean that a plant can only use its owned fleet, i.e., a vehicle cannot depart from a plant if it does not belong to that plant. Constraints (4.19) forbid subtours. Constraints (4.20)-(4.27) provide the ranges of the decision variables.

The proposed MFPRP is NP-hard since it contains a VRP that is well known to be NP-hard [51]. It is hard for a commercial solver to solve model $\mathcal{P}$ due to the inclusion of the subtour elimination constraints (4.19) whose number increases exponentially

with the number of nodes. In the next section, an efficient hybrid matheuristic is developed to solve the considered MFPRP.

## 4.3 Solution approach

In this section, we develop a hybrid matheuristic (HM) to solve the MFPRP. HM combines a two-phase iterative (TI) procedure, a fix-and-optimize (FO) strategy, and a route-based optimization (RO) process. The three components of HM are constructed based on mathematical formulations. In particular, useful information provided by TI and FO is exploited to find better solutions in RO. Generally, TI consists of solving a production direct-distribution problem (PDP) and a series of VRPs for each plant and time period iteratively to generate an initial solution to MFPRP that may be infeasible. Then FO repairs and improves the initial solution iteratively by solving a series of restricted PDPs (RPDPs). During the execution of TI and FO, a large number of vehicle routes will be generated and stored. Finally, to exploit the stored routes in the RO, a route-based model is formulated and an adaptive kernel search method (AKS) is developed to solve it. The general framework of HM is illustrated in Fig. 4.1. In the following subsections, the main components of HM are presented in detail.

### 4.3.1 Two-phase iterative method

TI consists of decomposing model $\mathcal{P}$ into an PDP and a series of VRPs and solving them iteratively. Parameter $\delta_{pit}$ representing the approximate visit cost from plant $p$ to retailer $i$ in period $t$ is introduced in the objective function of PDP. Once PDP is solved, the production setup variables and all continuous variables are determined, and the only decisions left are the routing ones. Then with the delivery schedules known for each plant $p$ and period $t$, a series of VRP$(p,t)$ is solved to form a solution to MFPRP. $\delta_{pit}$ is then updated based on the routing solution. This process is repeated until a stopping criterion is met. To formulate PDP, the vehicle routing constraints (4.10)-(4.19) of model $\mathcal{P}$ are first relaxed and aggregate vehicle capacity constraints are added. The last term in objective function (4.1) is replaced with the sum of approximate routing cost. The following new parameters and variables are introduced:

Parameters:

$\delta^j_{pit}$: approximate visit cost from plant $p \in P$ to retailer $i \in R$ in period $t \in T$ of the $j$th iteration. For $j = 0$, $\delta^j_{pit}$ is set to 0; for $j \geq 1$, it is updated based on the solution of the $(j-1)$th iteration.

Decision variables:

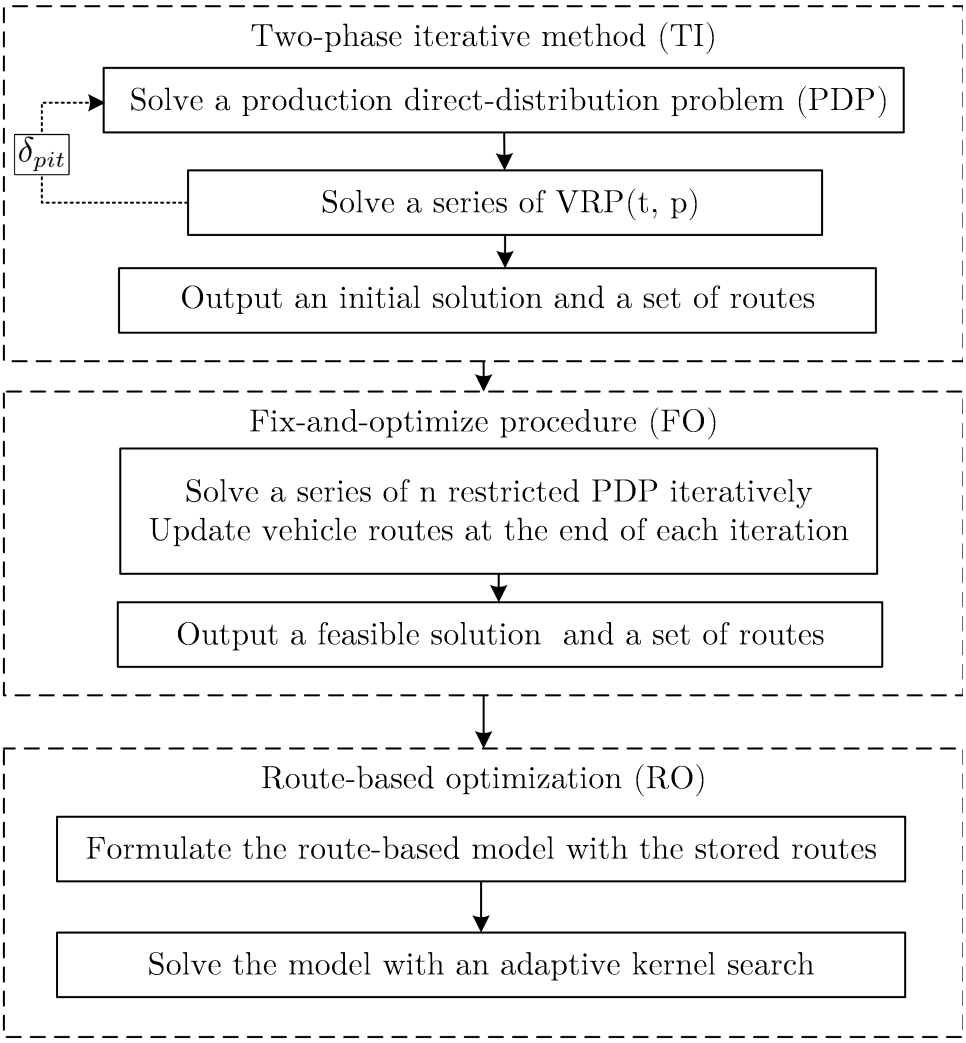



Fig. 4.1: General framework of HM

$y_{pist}$: delivery quantity of product with residual shelf life $s$ from plant $p$ to retailer $i$ in period $t$;

$v_{pit}$: equal to 1 if retailer $i$ is replenished by plant $p$ in period $t$, and 0 otherwise.

For the $j^{th}$ iteration, PDP is formulated as follows (model $\mathcal{P}_1(j)$):

$$\mathcal{P}_1(j): \max \sum_{i\in R}\sum_{s\in S}\sum_{t\in T} g_{is}\vartheta_{ist} - \sum_{p\in P}\sum_{t\in T}(a_{pt}\xi_{pt} + b_{pt}w_{pt}) - \sum_{p\in P}\sum_{u\in B}\sum_{t\in T} e_u z_{p,\theta(u),t} - \sum_{i\in N}\sum_{s\in S}\sum_{t\in T} h_i I_{ist} - \sum_{p\in P}\sum_{i\in R}\sum_{t\in T}\delta^j_{pit} v_{pit} \quad (4.28)$$

s.t.

$(4.5)-(4.9), (4.20)-(4.24)$, and

$$I_{pst} = I_{p,s+1,t-1} + z_{p,\theta(u),t} - \sum_{i\in R} y_{pist},\ \forall\, p\in P, s\in\{\theta(u)|u\in B\}, t\in T \quad (4.29)$$

$$I_{pst} = I_{p,s+1,t-1} - \sum_{i\in R} y_{pist},\ \forall\, p\in P, s\in S\setminus\{\theta(u)|u\in B\}, t\in T \quad (4.30)$$

$$I_{ist} = I_{i,s+1,t-1} + \sum_{p \in P} y_{pist} - \vartheta_{ist}, \ \forall\, i \in R, s \in S, t \in T \tag{4.31}$$

$$\sum_{i \in R} \sum_{s \in S} y_{pist} \leq |K_p| V, \ \forall\, p \in P, t \in T \tag{4.32}$$

$$\sum_{s \in S} y_{pist} \leq W_{it} v_{pit}, \ \forall\, i \in R, p \in P, t \in T \tag{4.33}$$

$$\sum_{p \in P} v_{pit} \leq 1, \ \forall\, i \in R, t \in T \tag{4.34}$$

$$y_{pist} \geq 0, \ \forall\, p \in P, i \in R, s \in S, t \in T \tag{4.35}$$

$$v_{pit} \in \{0, 1\}, \ \forall\, p \in P, i \in R, t \in T. \tag{4.36}$$

The objective function (4.28) maximizes the total profit that is equal to the revenue minus the production, packaging, inventory and approximate visit costs. Constraints (4.29)-(4.31) are the inventory flow balance constraints. Constraints (4.32) limit the total delivery quantity from a plant with the total capacity of its owned fleet size. Constraints (4.33) and (4.34) correspond to (4.11) and (4.12) in model $\mathcal{P}$, respectively, while these constraints are not specified to vehicle index.

Model $\mathcal{P}_1(j)$ consists of determining decisions including the production setup schedule $w_{pt}$, production quantity $\xi_{pt}$, packaging quantity $z_{p,\theta(u),t}$, inventory quantity $I_{ist}$, delivery schedule $v_{pit}$, delivery quantity $y_{pist}$, and selling quantity $\vartheta_{ist}$. Once $\mathcal{P}_1(j)$ is solved, with $v_{pit}$ and $y_{pist}$ known, a series of VRP$(p, t)$ for each plant $p \in P$ and each period $t \in T$ can be formulated and solved to determine the number of used vehicles and their routes. Then the approximate visit cost $\delta_{pit}^{j+1}$ for the $(j+1)$th iteration should be updated according to the solution of VRP$(p, t)$ in the $j$th iteration as: if retailer $i$ is served by plant $p$ in period $t$, we calculate $\delta_{pit}^{j+1}$ with (4.37); otherwise, it is calculated with (4.38). In particular, if the overall best solution does not improve for a number of $\tau$ iterations, the $(j+1)$th iteration is called a diversification iteration, and we re-generate $\delta_{pit}^{j+1}$ with (4.39). Equations (4.37)-(4.39) are:

$$\delta_{pit}^{j+1} = c_{i^- i} + c_{i i^+} - c_{i^- i^+} \tag{4.37}$$

$$\delta_{pit}^{j+1} = \min_{k \in K} \Delta_{pikt} \tag{4.38}$$

$$\delta_{pit}^{j+1} = RGIF[\min_{i' \in N}\{c_{ii'}\}, c_{pi} + c_{ip}] \tag{4.39}$$

where $i^-$ and $i^+$ denote the immediate predecessor and successor nodes of node $i$ in a given route, respectively. $\Delta_{pikt}$ is the cheapest insertion cost for inserting $i$ into an existing route performed by vehicle $k$ from plant $p$ in period $t$. $RGIF$ represents a randomly generated integer number from the interval.

The solution provided by TI may be infeasible because of the inclusion of the aggregate vehicle capacity constraints (4.32) for each plant $p \in P$. Because (4.32)

cannot guarantee the total delivery quantity by a plant to be packed into its owned fleet. In this case, it will be repaired and improved by FO. In addition, generated routes by solving VRP$(p,t)$ are stored in the set $\mathcal{R}$.

### 4.3.2 Fix-and-optimize procedure

As the solution provided in Section 4.3.1 may be infeasible, this section focuses on designing a fix-and-optimize procedure to: 1) repair infeasibility; 2) improve the incumbent solution. The procedure consists of iteratively forming and solving a RPDP. Each RPDP aims to optimize the delivery schedule for one retailer and the delivery schedules for the remaining retailers are fixed to the schedules provided by the previous iteration. Once a RPDP is solved, vehicle routes are updated by an insertion or removal strategy.

To formulate the RPDP for retailer $j$, where $j = \{1,\dots,n\}$, the delivery schedule variables $v_{pikt}$ of retailers $i \in R \setminus \{j\}$ in model $\mathcal{P}$ are fixed to the delivery schedule $\hat{v}_{pikt}$ provided by the previous iteration. Especially, in the first RPDP, for all retailers $i = \{2,\dots,n\}$, their delivery schedule $v_{pikt}$ is fixed to $\hat{v}_{pikt}$ provided by TI. Secondly, to better approximate the routing cost from depot $p$ to retailer $j$ by vehicle $k$, a similar approximate visit cost $\sigma_{pjkt}$ is introduced. Note that the approximate visit cost $\delta_{pjt}$ in model $\mathcal{P}_1$ is not specific to any vehicle. $\sigma_{pjkt}$ is calculated based on solution of the previous iteration: if retailer $j$ is visited by $k$ from $p$ in period $t$, $\sigma_{pjkt}$ can be calculated by (4.37); otherwise, $\sigma_{pjkt} = \Delta_{pjkt}$. Finally, to respect the vehicle fleet size constraint in RPDP, vehicles exceeding the fleet size of plant $p$ in the infeasible solution are placed into a set $\mathcal{V}_p$. Let $|\mathcal{K}_{pt}|$ be the number of vehicles used by plant $p$ in period $t \in T$. If there exists any $|\mathcal{K}_{pt}| > |K_p|$, the vehicles are sorted in non-increasing order of their delivery quantity in an ordered list. Then an oreded set of exceeded vehicles is put into $\mathcal{V}_{pt}$, where $\mathcal{V}_{pt} = \{|K_p|+1,\dots,|\mathcal{K}_{pt}|\}$. Thus the set of exceeded vehicles of plant $p$ can be denoted as $\mathcal{V}_p = \{K_p+1,\dots,K_p+|\mathcal{V}_p|\}$, where $|\mathcal{V}_p| = \max_{t\in T}\{|\mathcal{V}_{pt}|\}$. Note that in $\mathcal{P}_2(j)$, the vehicle fleet $K_p$ for plant $p$ is extended to include the set $\mathcal{V}_p$, i.e., $K_p = K_p \cup \mathcal{V}_p$.

The $j^{th}$ RPDP can be formulated as follows ($\mathcal{P}_2(j)$) :

$$\begin{aligned}\mathcal{P}_2(j):\ \max & \sum_{i\in R}\sum_{s\in S}\sum_{t\in T} g_{is}\vartheta_{ist} - \sum_{p\in P}\sum_{t\in T}(a_{pt}\xi_{pt}+b_{pt}w_{pt}) - \sum_{p\in P}\sum_{u\in B}\sum_{t\in T} e_u z_{p,\theta(u),t} \\ & - \sum_{i\in N}\sum_{s\in S}\sum_{t\in T} h_i I_{ist} - \sum_{p\in P}\sum_{k\in K_p}\sum_{t\in T}\sigma_{pjkt}v_{pjkt} \qquad (4.40)\end{aligned}$$

s.t.

$(4.2)-(4.10), (4.20)-(4.25)$, and

$$\sum_{s\in S} y_{piskt} \ \le\ W_{it}\hat{v}_{pikt},\ \forall\, p\in P, i\in R\setminus\{j\}, t\in T, k\in K_p \qquad (4.41)$$

$$\sum_{s \in S} y_{pjskt} \leq W_{jt} v_{pjkt}, \ \forall p \in P, k \in K_p, t \in T \tag{4.42}$$

$$v_{pjkt} = 0, \ \forall p \in P, k \in \mathcal{V}_p, t \in T \tag{4.43}$$

$$\sum_{p \in P} \sum_{k \in K_p} v_{pjkt} \leq 1, \ \forall t \in T \tag{4.44}$$

$$v_{pjkt} \in \{0, 1\}, \ \forall p \in P, k \in K_p \setminus \mathcal{V}_p, t \in T. \tag{4.45}$$

The objective function (4.40) maximizes the total profit. Constraints (4.41) indicate that all retailers $i \in R \setminus j$ should respect a given deliery schedule, and the delivery quantity should not exceed its remaining demand. Constraints (4.42) mean that the delivery quantity to $j$ should be 0 if it is not visited and should not exceed its total remaining demand. Constraints (4.43) indicate that retailer $j$ cannot be served by exceeded vehicles. Constraints (4.44) forbid split delivery to $j$.

Once $\mathcal{P}_2(j)$ is solved, the routes concerning $j$ are updated by an insertion or removal strategy according to the value of $v_{pjkt}$. Especially, the infeasibility in which retailer $j$ is visited by an exceeded vehicle is repaired.

Note that in FO, any newly generated routes will also be stored in the set $\mathcal{R}$.

### 4.3.3 Route-based optimization

The route-based optimization RO aims to improve the incumbent solution by exploiting the useful information collected from TI method and the FO procedure. This is done through formulating and solving a route-based model. Similar route-based models have been used to solve IRPs by [16] and [22]. Precisely, the route-based model is constructed with the set $\mathcal{R}$ of stored routes to avoid the large sizes of vehicle routing variables $x_{ijkt}$ and subtour elimination constraints (4.19). Note that for a given route, its cost, its originated plant and its visited retailers are known. For the model, an adaptive kernel search AKS is developed to solve it. With the following new parameters and variables, the route-based model is presented to exploit the set $\mathcal{R}$ of routes.

New parameters:

$\mathcal{R}$: set of routes generated by TI and FO;
$c_r$: cost to perform route $r \in \mathcal{R}$;
$\alpha_{ir}$: equal to 1 if retailer $i$ is in route $r$, and 0 otherwise;
$\beta_{pr}$: equal to 1 if route $r$ originates from plant $p$, and 0 otherwise.

New decision variables:

$y_{rist}$: delivery quantity with residual shelf life $s$ by route $r$ to retailer $i$ in period $t$;
$x_{rt}$: equal to 1 if route $r \in \mathcal{R}$ is used in period $t$, and 0 otherwise.

The route-based model can be formulated as follows (model $\mathcal{P}_3$):

$$(\mathcal{P}_3): \max f = \sum_{i\in R}\sum_{s\in S}\sum_{t\in T} g_{is}\vartheta_{ist} - \sum_{p\in P}\sum_{t\in T}(a_{pt}\xi_{pt} + b_{pt}w_{pt}) - \sum_{p\in P}\sum_{u\in B}\sum_{t\in T} e_u z_{p,\theta(u),t} - \sum_{i\in N}\sum_{s\in S}\sum_{t\in T} h_i I_{ist} - \sum_{r\in\mathcal{R}}\sum_{t\in T} c_r x_{rt} \quad (4.46)$$

s.t.

$(4.5)-(4.9), (4.20)-(4.24)$, and

$$I_{pst} = I_{p,s+1,t-1} + z_{p,\theta(u),t} - \sum_{r\in\mathcal{R}}\sum_{i\in R}\beta_{pr}\alpha_{ir}y_{rist}, \quad \forall\, p\in P, s\in\{\theta(u)|u\in B\}, t\in T \quad (4.47)$$

$$I_{pst} = I_{p,s+1,t-1} - \sum_{r\in\mathcal{R}}\sum_{i\in R}\beta_{pr}\alpha_{ir}y_{rist}, \forall\, s\in S\setminus\{\theta(u)|u\in B\}, t\in T \quad (4.48)$$

$$I_{ist} = I_{i,s+1,t-1} + \sum_{r\in\mathcal{R}}\alpha_{ir}y_{rist} - \vartheta_{ist}, \forall\, i\in R, s\in S, t\in T \quad (4.49)$$

$$\sum_{s\in S} I_{ist} \geq (1 - \sum_{r\in\mathcal{R}}\sum_{\tau=t+1}^{\tau=t'}\alpha_{ir}x_{r\tau})\sum_{\tau=t+1}^{\tau=t'} D_{i\tau},\ \forall\, i\in R, t\in T, t<t'\leq|T| \quad (4.50)$$

$$\sum_{i\in R}\sum_{s\in S}\alpha_{ir}y_{rist} \leq V x_{rt},\ \forall\, r\in\mathcal{R}, t\in T \quad (4.51)$$

$$\sum_{s\in S} y_{rist} \leq W_{it}\alpha_{ir},\ \forall\, r\in\mathcal{R}, i\in R, t\in T \quad (4.52)$$

$$\sum_{r\in\mathcal{R}}\alpha_{ir}x_{rt} \leq 1,\ \forall\, i\in R, t\in T \quad (4.53)$$

$$\sum_{r\in\mathcal{R}}\beta_{pr}x_{rt} \leq |K_p|,\ \forall\, p\in P, t\in T \quad (4.54)$$

$$y_{rist} \geq 0,\ \forall\, r\in\mathcal{R}, i\in R, s\in S, t\in T \quad (4.55)$$

$$x_{rt} \in \{0,1\},\ \forall\, r\in\mathcal{R}, t\in T. \quad (4.56)$$

The objective function (4.46) maximizes the total profit. Constraints (4.47)-(4.49) impose inventory balance. Constraints (4.50) indicate that the inventory of retailer $i$ in period $t$ must be sufficient to meet the demand in the following $t'-t$ periods if $i$ is not visited during these periods. Constraints (4.51) mean the vehicle capacity must be respected. Constraints (4.52) ensure that the delivery quantity to retailer $i$ by route $r$ must be 0 if $i$ is not visited by route $r$. Constraints (4.53) forbid split delivery. Constraints (4.54) state that the number of used vehicles cannot exceed the fleet size of each plant and each period. Constraints (4.55) and (4.56) bound the variables.

Our preliminary experiments show that model $\mathcal{P}_3$ cannot be optimally solved by a commercial solver when the number of routes increases. Thus we develop an adaptive kernel search (AKS) heuristic for it.

The kernel search (KS) algorithm is designed to solve mixed binary programs (MBP). Since its introduction by [13] for the multi-dimensional knapsack problem, it has been successfully applied to single-source capacitated facility location [55], alternative-fuel station location [97], and bus lane reservation problems [98], among others. The bisic idea of KS is to carefully build and solve a set of restricted MBPs iteratively by fixing a subset of binary variables to obtain a near-optimal solution of the original problem. Since model $\mathcal{P}_3$ aims to select a good combination of the routes in $\mathcal{R}$ to improve the incumbent solution, we select variables $x_{rt}$ to form the kernel. Our AKS generates a near-optimal solution to $\mathcal{P}_3$ through the following steps.

Firstly, the linear relaxation (LP) of model $\mathcal{P}_3$ is solved to optimality: if all binary variables $w_{pt}$ and $x_{rt}$ take integer values in the solution, then AKS terminates with providing an optimal solution to $\mathcal{P}_3$. Otherwise, all variables $x_{rt}$ are sorted in an ordered list $\mathcal{B}$. In the list, variables $x_{rt}$ that take value 1 in the solution provided by Section FO are put in the beginning of the list, and the remaining variables $x_{rt}$ are first sorted in non-increasing order of their values from the solution of the LP and then in non-increasing order of their reduced costs.

Secondly, our preliminary experiments show that it is difficult to search an appropriate kernel for model $\mathcal{P}_3$. Thus an *adaptive process* is developed to determine the initial kernel $\mathcal{K}_1$. Firstly, the first $L$ variables in list $\mathcal{B}$ are selected as a temporal kernel $\mathcal{K}$, and an expected gap of the upper and lower bounds and a time limit are set. Then, a MBP($\mathcal{K}$) is formulated by fixing all $x_{rt}$ variables that are not in $\mathcal{K}$ to 0 in model $\mathcal{P}_3$, i.e., $x_{rt} = 0, \forall x_{rt} \notin \mathcal{K}$. If MBP($\mathcal{K}$) is optimally solved or the gap between the obtained upper and a lower bounds is lower or equal to the expected one in the time limit, $\mathcal{K}_1$ is set as $\mathcal{K}$ and the process stops. Otherwise, $L$ is halved and $\mathcal{K}$ is updated, and a new MBP($\mathcal{K}$) is formed. This process is repeated until MBP($\mathcal{K}$) is optimally solved or the stopping criterion is met.

Then the first $L$ variables in set $\mathcal{B}$ form the first kernel $\mathcal{K}_1$ and the remaining variables are divided into $m$ buckets of size $L$, where $m = \lceil \frac{|\mathcal{B}|-L}{L} \rceil$. The $m-1$ buckets have the same length $L$ and the $m$th bucket may take a length smaller than $L$. The $m$ MBPs are solved iteratively. The $l$th MBP denoted as MBP($\mathcal{K}_l \cup \mathcal{B}_l$), in which the variables $x_{rt}$ except for those in $\mathcal{K}_l$ and $\mathcal{B}_l$ are fixed to 0, is formulated as follows:

$$\text{MBP}(\mathcal{K}_l \cup \mathcal{B}_l): \ \max f \tag{4.57}$$

s.t.

$$(4.5) - (4.9), (4.20) - (4.24), (4.47) - (4.56), \text{and}$$

$$x_{rt} = 0, \ \forall x_{rt} \notin \mathcal{K}_l \cup \mathcal{B}_l \tag{4.58}$$

$$f \geq z_{l-1}^{LB} \tag{4.59}$$

$$\sum_{x_{rt} \in \mathcal{B}_l} x_{rt} \geq 1 \tag{4.60}$$

where $z_{l-1}^{LB}$ is the best lower bound obtained so far, especially, $z_0^{LB}$ denotes the lower bound obtained by solving $\mathcal{K}_1$. Constraints (4.58) restrict MBP($\mathcal{K}_l \cup \mathcal{B}_l$ ) by fixing the $x_{rt}$ variables that are not in the current kernel $\mathcal{K}_l$ and bucket $\mathcal{B}_l$ to 0. Constraint (4.59) guarantees that the obtained objective value is not worse than the current best lower bound. Constraint (4.60) ensures that at least one variable in the current bucket $\mathcal{B}_l$ is equal to 1 in a feasible solution. Constraints (4.59) and (4.60) are introduced to alleviate the computational effort for solving MBP($\mathcal{K}_l \cup \mathcal{B}_l$).

Once MBP($\mathcal{K}_l \cup \mathcal{B}_l$) is solved, $\mathcal{K}_{l+1}$ and $z_{l+1}^{LB}$ are updated as follows: if MBP($\mathcal{K}_l \cup \mathcal{B}_l$) is infeasible, $\mathcal{K}_{l+1}$ and $z_{l+1}^{LB}$ stay unchanged. Otherwise, $\mathcal{K}_{l+1}$ is set to $\mathcal{K}_l \cup \mathcal{B}_l^+ \setminus \mathcal{K}_l^-$, where $\mathcal{B}_l^+$ consists of $x_{rt} \in \mathcal{B}_l$ taking value 1 in the solution of MBP($\mathcal{K}_l \cup \mathcal{B}_l$), and $\mathcal{K}_l^-$ contains $x_{rt} \in \mathcal{K}_l$ taking value 0 in the solution of the $\tau$ previous iterations, where $\tau$ is an integer parameter. Finally, AKS stops when all MBP($\mathcal{K}_l \cup \mathcal{B}_l$), $l = 1, \ldots, m$ are solved.

### 4.3.4 Hybrid matheuristic outline

Our HM can be summarized in Algorithm 4.1, in which the following parameters are defined. $sol^*$ (resp. $obj^*$) and $sol$ (resp. $obj$) denote the best-obtained solution (resp. best objective value) and the current solution (resp. current objective value), respectively. $m$ is the number of buckets in AKS. $j$ is the iteration counter and $J$ indicates the total number of iterations that are allowed in TI of HM. $\mathcal{R}$ is the set of stored routes.

In Algorithm 1, lines 1-11 correspond to TI in which model $\mathcal{P}_1$ and a series of VRP($p, t$) are iteratively solved. Lines 12-21 represent FO where model $\mathcal{P}_2(j)$ is solved for each retailer $j$. Lines 22-33 present the RO. The linear relaxation of model $\mathcal{P}_3$ is optimally solved. Then an adaptive procedure is developed to construct the kernel $\mathcal{K}_1$, and a number of $m$ buckets are constructed. Finally a sequence of MBP($\mathcal{K}_l \cup \mathcal{B}_l$) is solved to obtain the final solution.

## 4.4 Computational study

In this section, numerical experiments on 320 randomly generated instances (64 sets of 5 instances) are carried out to evaluate the performance of the developed hybrid matheuristic. The generated instances and detailed computational results are available online at `http://www.mediafire.com/file/9zgsbgnisl5an0l/MFPRP.rar`. The proposed HM is implemented in C++ in Windows 10 operating system and compiled under Visual C++ 2015 Community Edition. All tests are performed on

**Algorithm 4.1** HM for MFPRP

1. Initialize $j = 1, sol^* \leftarrow \phi, sol \leftarrow \phi, obj, obj^* \leftarrow 0, M, \delta_{pit}^m$ for all $p \in P, i \in R, t \in T$,$\mathcal{R} \leftarrow \phi$
2. **while** $(j \leq J)$
3. Solve $\mathcal{P}_1$, output $\hat{w}_{pt}, \hat{q}_{pt}, \hat{z}_{p,\theta(u),t}, \hat{I}_{ist}, \hat{y}_{pist}$, $\hat{d}_{ist}$ and $\hat{v}_{pit}$
4. Solve VRP$(p, t)$ for each plant $p$ and period $t$ with $\hat{v}_{pit}$ and $\hat{y}_{pist}$ known
5. Form a solution $sol$ to MFPRP (potentially infeasible), calculate $obj$ and add all routes to $\mathcal{R}$
6. **if** $(obj > obj^*)$
7. Set $obj^* \leftarrow obj$ and update the corresponding best solution $sol^* \leftarrow sol$
8. **end if**
9. Update $\delta_{pit}^{j+1}$ with (4.37), (4.38) or (4.39)
10. Set $j \leftarrow j + 1$
11. **end while**

12. Set $j = 1$, get the delivery schedule $\hat{v}_{pikt}$ for all retailers $i$, calculate visit cost $\sigma_{pjkt}$ for the $j$th retailer based on $sol^*$
13. **while** $(j \leq n)$
14. Solve model $\mathcal{P}_2(j)$ with all $v_{pikt}$ fixed to $\hat{v}_{pikt}$, where $i = R \setminus \{j\}$
15. Perform insertion or removal procedure for retailer $j$, update $sol$, calculate $obj$ and
add routes to $\mathcal{R}$
16. **if** $(obj > obj^*)$
17. Set $obj^* \leftarrow obj$ and $sol^* \leftarrow sol$
18. **end if**
19. Set $j \leftarrow j + 1$
20. Calculate visit cost $\sigma_{pjkt}$ based on $sol^*$: if $\hat{v}_{pjkt} = 1$, then $\sigma_{pjkt}$ is calculated by (4.37); otherwise, $\sigma_{pjkt} = \Delta_{pjkt}$
21. **end while**

22. Formulate model $\mathcal{P}_3$ with the set of stored routes $\mathcal{R}$
23. Solve LP$(\mathcal{P}_3)$, sort the binary variables $x_{rt}$ with the predetermined criterion
24. Determine the kernel $\mathcal{K}_1$ and bucket length $L$ with the *adaptive process*
25. Construct $m$ buckets, where $m = \lceil \frac{|\mathcal{B}|-L}{L} \rceil$
26. **for**$(l = 1, \ldots, m)$ **do**
27. Solve MBP$(\mathcal{K}_l \cup \mathcal{B}_l)$ formed by kernel $\mathcal{K}_l$ and bucket $\mathcal{B}_l$
28. Update the kernel $\mathcal{K}_{l+1} = \mathcal{K}_l \cup \mathcal{B}_l^+ \setminus \mathcal{K}_l^-$
29. **if** $(obj > obj^*)$
30. Set $obj^* \leftarrow obj$ and $sol^* \leftarrow sol$
31. **end if**
32. **end for**
33. Return $sol^*$ and $obj^*$

a PC with Intel Core i7 CPU (2.5 GHz) and 8 GB RAM. CPLEX 12.7.1 with default settings is called to solve subproblems in HM, i.e., $\mathcal{P}_1$, $\mathcal{P}_2(j)$, LP($\mathcal{P}_3$), MBP($\mathcal{K}$), and MBP($\mathcal{K}_l \cup \mathcal{B}_l$). All VRPs are solved with the VRPH packages [53] with the record-to-record algorithm implementation [65].

In Section 4.4.1, we first detail the parameter settings, instance generation and two implementations to solve the integrated model $\mathcal{P}$ by CPLEX. Computational results are presented in Section 4.4.2. Section 4.4.3 is devoted to examining the impacts of different product discount policies. Finally, the performance HM is analyzed in Section 4.4.4.

### 4.4.1 Parameter settings and instance generation

The parameters in HM are set as: 1) for the two-phase iterative method, the initial approximate visit cost $\delta^0_{pit}$ is set to 0; the number of iterations allowed in TI is set to 30; the diversification mechanism is applied after 2 consecutive iterations without improvement; model $\mathcal{P}_1$ is solved with 100s time limit; 2) for the adaptive kernel search, MBP($\mathcal{K}$) and MBP($\mathcal{K}_l \cup \mathcal{B}_l$) are solved with 500s time limit, and $\tau$ is set to 2.

Since MFPRP is a newly studied problem, there are no benchmark instances and algorithms available in the literature. Thus the 320 instances are randomly generated based on the IRP instance generation of [39] and the PRP instance generation of [15]. The number of plants $|P|$, retailers $n$, periods $|T|$ and packages $|B|$ are set to {2, 4}, {6, 8, 10, 15, 20, 30, 40, 50}, {3, 6} and {2, 3}, respectively. Fleet sizes $|K_p|$ of each plant $p \in P$ are set equally to 1, 2, and 3 for instances with 6-20, 30-40, and 50 customers, respectively. Product shelf life $\theta(u)$ of the newly-produced product with package $u$ is set to {1, 2} and {1, 2, 3} for instances with 2 and 3 types of packages, respectively. The coordinate of a node $i$ in graph $G$ that corresponds to a plant or a retailer location $(X_i, Y_i)$ is randomly generated from U[0,1000]. At the beginning of the planning horizon, the initial inventory $I_{is0}$ is set to 0 for all $i \in N$ and $s \in S$. The detailed generation of parameters for MFPRP is given in Table 4.1.

To evaluate the obtained results by HM, lower and upper bounds of MFPRP are generated with CPLEX. Model $\mathcal{P}$ cannot be directly solved by CPLEX due to the inclusion of the subtour elimination constraints (4.19), whose number increases with the number of nodes. Thus we introduce two ways to generate bounds of MFPRP. The basic idea is to relax the subtour elimination constraint (4.19) and to add them iteratively as needed. We denote the relaxed model of $\mathcal{P}$ as RP($\mathcal{P}$) that is defined by the objective function (4.1) and constraints (4.2)-(4.18) and (4.20)-(4.26). The two implementations are detailed as follows. In the first implementation, denoted as CP1, model RP($\mathcal{P}$) is solved to optimality. If there exists one or more subtours, the corresponding subtour elimination constraints are added to model RP($\mathcal{P}$), and RP($\mathcal{P}$)

Table 4.1: Parameters for MFPRP model

| Parameters | Generation description |
|---|---|
| $c_{ij}$ | $= \lfloor \sqrt{(X_i - X_j)^2 + (Y_i - Y_j)^2} + 0.5 \rfloor$ |
| $D_{it}$ | randomly generated from U[30, 210] |
| $C_p$ | $= \beta_p(\sum_{i \in R} \sum_{t \in T} D_{it})/\lvert T \rvert/\lvert P \rvert$, where $\beta$ is randomly generated from U[2, 4], $p \in P$ |
| $U_p$ | $= \beta_p C_p$, where $\beta_p$ is randomly generated from U[1.5, 2.0], $p \in P$ |
| $U_i$ | $= \beta_i \max_{t \in T}\{D_{it}\}$, where $\beta_i$ is randomly generated from U[2, 3], $i \in R$ |
| $h_i$ | randomly generated from U[0.2, 0.6] |
| $a_p$ | randomly generated from U[4, 7] |
| $e_u$ | $= \beta_u \theta(u)$, where $\beta_u$ is randomly generated from U[1, 2], $u \in B$ |
| $b_p$ | $= \beta C_p$, where $\beta$ is randomly generated from U[0.3, 0.5] |
| $g_{is}$ | $= \beta_i - (\lvert S \rvert - s + 1)^{\epsilon}$, where $\beta_i$ is randomly generated from U[16, 21] and $\epsilon = 1$ |
| $V$ | $= \beta(\sum_{i \in R} \sum_{t \in T} D_{it})/\sum_{p \in P} \lvert K_p \rvert/\lvert T \rvert$, where $\beta$ is randomly generated from U[1.5, 3.0] |

with the added subtours is then re-solved. This process is repeated until there are no subtours in the solution to RP($\mathcal{P}$) or a given time limit is reached. Alternatively, in the second implementation, denoted as CP2, a lazy constraint callback feature of CPLEX [58] is applied when RP($\mathcal{P}$) is solved by the branch-and-cut algorithm (B&C) in CPLEX. The principle of the feature consists of checking the existence of subtours whenever a feasible solution to RP($\mathcal{P}$) is provided by solving its subproblem in the search tree of B&C. If one or more subtours exist, the corresponding subtour elimination constraints with new cuts are added to form a new subproblem. The process continues until B&C terminates or a given time limit is rearched. The difference of CP1 and CP2 is that CP1 solves RP($\mathcal{P}$) multiple times while CP2 solves RP($\mathcal{P}$) only once. We set a time limit of 3600s for each implementation.

### 4.4.2 Computational results

In this section, we report the computational results on the 64 sets of instances (each containing 5 instances of the same size). The results are compared with those provided by CP1 and CP2. Tables 4.2 and 4.3 show the average results of each set. In the two tables, columns 1-5 denote the instance set number, numbers of plants, periods, retailers and packages, respectively. Columns "LB1", "LB2", and "LB3" are the lower bounds (feasible solutions) obtained by CP1, CP2, and HM, respectively. Note that the two numbers in "$^{(,)}$" indicate the numbers of instances for which an optimal and a feasible solution are found, respectively. The upper (dual) bounds provided by CP1 and CP2 are shown in columns "UB1" and "UB2". The computation times

in seconds for CP1, CP2, and HM are shown in columns "T1", "T2" and "T3", respectively. Column "Dev(%)" indicates the deviations of the solution obtained by HM with respect to the better solution provided by CP1 and CP2. A negative value of "Dev(%)" means that HM finds a better solution compared to CP1 and CP2. "Gap(%)" denotes the gap between the solution provided by HM and the best upper bound provided by CPLEX. The calculation of "Dev(%)" and "Gap(%)" is as follows:

$$\text{Dev}(\%) = \frac{\max(\text{LB1}, \text{LB2}) - \text{LB3}}{\max(\text{LB1}, \text{LB2})} \times 100\% \tag{4.61}$$

$$\text{Gap}(\%) = \frac{\min(\text{UB1}, \text{UB2}) - \text{LB3}}{\text{LB3}} \times 100\%. \tag{4.62}$$

Table 4.2 presents the numerical results for 32 instance sets (160 instances in total) with up to 4 plants and 15 retailers. From its last row, we can see that CP1 finds optimal solutions in 75 instances, and it cannot find any feasible solution in instances with 4 plants, 6 periods, and 8 retailers or more within the time limit. Note that for some sets of instances that are not all solved optimally, the value of LB1 is bigger than UB1. This is because the value of LB1 is the average objective value of instances that are solved to optimality, and UB1 is the average upper bounds for all the five instances. As for CP2, column "LB2" shows that it obtains optimal and feasible solutions in 70 and 154 out of the 160 tested instances, respectively. CP2 generally outperforms CP1 in terms of lower bound but it is a little inferior in finding optimal solutions or upper bounds. From column "LB3", we can observe that HM is able to provide solutions for all instances, finding optimal ones for 30 instances. The average deviation between the provided lower bounds by HM and CPLEX is -1.26%, which demonstrates that HM is superior to CPLEX in terms of solution quality. The average gap between the lower bound provided by HM and the best upper bound obtained by CPLEX is 5.09%, which reveals that HM is able to consistently find good quality solutions. Looking at the computation times, the average computation times of CP1, CP2 and HM are 2262.9 s, 2211.0 s and 106.0s, respectively. It indicates that HM consumes less than 5% of the computation time of CPLEX. The overall results demonstrate that HM performs well for instances with up to 4 plants and 15 retailers.

The results for instances with up to 4 plants and 50 retailers are presented in Table 4.3. We can observe from column "LB1" that CP1 only finds 6 optimal solutions and it cannot find any feasible solution for the remaining 154 instances within the time limit. CP2 finds 6 optimal solutions among 33 obtained feasible ones. Again, CP2 provides better lower bounds than CP1, while CP1 is superior to CP2 in generating upper bounds. For HM, it succeeds to provide feasible solutions for all instances, especially, one optimal solution is found. The average deviation is -3.04%, which indicates that HM is again superior to CPLEX in providing better feasible solutions.

Table 4.2: Results for instances with up to 4 plants and 15 retailers

| NO. | $m$ | $\|T\|$ | $n$ | $\|B\|$ | CP1 | | | CP2 | | | HM | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | LB1 | UB1 | T1(s) | LB2 | UB2 | T2(s) | LB3 | T3 (s) | Dev(%) | Gap(%) |
| 1 | 2 | 3 | 6 | 2 | $13187^{(5,5)}$ | 13187 | 15.8 | $13187^{(5,5)}$ | 13187 | 4.4 | $13187^{(5,5)}$ | 5.7 | 0.00 | 0.00 |
| 2 | 2 | 3 | 6 | 3 | $10611^{(5,5)}$ | 10611 | 22.1 | $10611^{(5,5)}$ | 10611 | 4.2 | $10533^{(4,5)}$ | 6.3 | 0.66 | 0.69 |
| 3 | 2 | 3 | 8 | 2 | $20772^{(5,5)}$ | 20772 | 63.9 | $20772^{(5,5)}$ | 20772 | 12.1 | $20669^{(3,5)}$ | 5.8 | 0.48 | 0.48 |
| 4 | 2 | 3 | 8 | 3 | $18441^{(5,5)}$ | 18441 | 436.4 | $18441^{(5,5)}$ | 18441 | 60.3 | $18089^{(2,5)}$ | 7.4 | 1.98 | 2.08 |
| 5 | 2 | 3 | 10 | 2 | $26664^{(4,4)}$ | 27425 | 915.8 | $27398^{(5,5)}$ | 27398 | 403.0 | $27311^{(3,5)}$ | 7.6 | 0.29 | 0.30 |
| 6 | 2 | 3 | 10 | 3 | $19316^{(4,4)}$ | 21624 | 811.9 | $21562^{(5,5)}$ | 21562 | 107.5 | $21507^{(3,5)}$ | 8.7 | 0.27 | 0.28 |
| 7 | 2 | 3 | 15 | 2 | $43916^{(4,4)}$ | 43552 | 1129.3 | $43487^{(4,5)}$ | 43676 | 813.9 | $43324^{(0,5)}$ | 9.2 | 0.45 | 0.61 |
| 8 | 2 | 3 | 15 | 3 | $34892^{(4,4)}$ | 34904 | 2080.5 | $34627^{(3,5)}$ | 35230 | 1680.0 | $34357^{(0,5)}$ | 20.4 | 0.76 | 1.58 |
| 9 | 2 | 6 | 6 | 2 | $28916^{(5,5)}$ | 28916 | 379.3 | $28916^{(4,5)}$ | 28954 | 835.8 | $28723^{(2,5)}$ | 22.9 | 0.57 | 0.57 |
| 10 | 2 | 6 | 6 | 3 | $23944^{(5,5)}$ | 23944 | 827.2 | $23939^{(4,5)}$ | 24117 | 1274.0 | $23601^{(0,5)}$ | 60.6 | 1.52 | 1.55 |
| 11 | 2 | 6 | 8 | 2 | $37971^{(3,3)}$ | 40275 | 2081.0 | $39993^{(2,5)}$ | 40691 | 2547.0 | $39783^{(2,5)}$ | 43.4 | 0.58 | 1.25 |
| 12 | 2 | 6 | 8 | 3 | $38123^{(4,4)}$ | 36129 | 1151.0 | $36027^{(4,5)}$ | 36424 | 1579.5 | $35737^{(1,5)}$ | 53.5 | 0.79 | 1.17 |
| 13 | 2 | 6 | 10 | 2 | - | 48935 | 3600.0 | $47520^{(1,5)}$ | 49753 | 3464.0 | $47637^{(0,5)}$ | 207.1 | -0.24 | 2.61 |
| 14 | 2 | 6 | 10 | 3 | $52744^{(1,1)}$ | 44537 | 3455.3 | $40969^{(1,5)}$ | 45789 | 2976.4 | $41671^{(0,5)}$ | 114.4 | -2.40 | 7.71 |
| 15 | 2 | 6 | 15 | 2 | - | 87712 | 3600.0 | $86578^{(0,5)}$ | 89572 | 3600.0 | $86776^{(0,5)}$ | 298.0 | -0.22 | 1.11 |
| 16 | 2 | 6 | 15 | 3 | $76140^{(1,1)}$ | 80739 | 2922.5 | $79214^{(1,5)}$ | 81875 | 3432.9 | $79672^{(1,5)}$ | 111.5 | -0.57 | 1.38 |
| 17 | 4 | 3 | 6 | 2 | $12275^{(5,5)}$ | 12275 | 467.4 | $12275^{(5,5)}$ | 12275 | 68.4 | $12079^{(1,5)}$ | 20.4 | 1.78 | 1.84 |
| 18 | 4 | 3 | 6 | 3 | $12101^{(5,5)}$ | 12101 | 235.5 | $12101^{(5,5)}$ | 12101 | 75.0 | $11893^{(0,5)}$ | 14.2 | 1.84 | 1.92 |
| 19 | 4 | 3 | 8 | 2 | $19087^{(3,3)}$ | 20163 | 1859.4 | $19971^{(3,5)}$ | 20395 | 2204.9 | $19753^{(2,5)}$ | 18.5 | 1.33 | 2.13 |
| 20 | 4 | 3 | 8 | 3 | $15368^{(1,1)}$ | 16361 | 3158.2 | $15887^{(3,5)}$ | 16514 | 2409.9 | $15607^{(1,5)}$ | 22.8 | 1.99 | 4.74 |
| 21 | 4 | 3 | 10 | 2 | - | 27091 | 3600.0 | $24844^{(0,5)}$ | 27942 | 3600.0 | $25033^{(0,5)}$ | 30.0 | -1.07 | 9.74 |
| 22 | 4 | 3 | 10 | 3 | - | 21544 | 3600.0 | $19959^{(0,5)}$ | 21882 | 3600.0 | $20023^{(0,5)}$ | 63.6 | -0.91 | 8.02 |
| 23 | 4 | 3 | 15 | 2 | - | 40861 | 3600.0 | $37178^{(0,5)}$ | 41555 | 3600.0 | $38722^{(0,5)}$ | 25.4 | -4.50 | 5.57 |
| 24 | 4 | 3 | 15 | 3 | - | 37183 | 3600.0 | $34765^{(0,5)}$ | 38094 | 3600.0 | $35741^{(0,5)}$ | 32.9 | -3.30 | 4.29 |
| 25 | 4 | 6 | 6 | 2 | $31437^{(1,1)}$ | 29754 | 3600.0 | $27172^{(0,5)}$ | 30597 | 3600.0 | $27158^{(0,5)}$ | 145.4 | -0.06 | 11.15 |
| 26 | 4 | 6 | 6 | 3 | - | 26751 | 3600.0 | $23878^{(0,5)}$ | 26954 | 3600.0 | $23579^{(0,5)}$ | 271.6 | 1.34 | 13.80 |
| 27 | 4 | 6 | 8 | 2 | - | 46910 | 3600.0 | $42715^{(0,5)}$ | 47853 | 3600.0 | $43753^{(0,5)}$ | 76.1 | -2.37 | 7.12 |
| 28 | 4 | 6 | 8 | 3 | - | 36474 | 3600.0 | $29779^{(0,5)}$ | 37076 | 3600.0 | $31208^{(0,5)}$ | 414.0 | -4.94 | 16.94 |
| 29 | 4 | 6 | 10 | 2 | - | 57429 | 3600.0 | $47491^{(0,5)}$ | 59133 | 3600.0 | $50251^{(0,5)}$ | 248.7 | -5.83 | 14.56 |
| 30 | 4 | 6 | 10 | 3 | - | 42706 | 3600.0 | $33419^{(0,5)}$ | 43713 | 3600.0 | $36924^{(0,5)}$ | 627.0 | -10.31 | 16.20 |
| 31 | 4 | 6 | 15 | 2 | - | 84849 | 3600.0 | $64864^{(0,4)}$ | 85868 | 3600.0 | $78128^{(0,5)}$ | 199.1 | -19.04 | 8.86 |
| 32 | 4 | 6 | 15 | 3 | - | 83483 | 3600.0 | - | 85227 | 3600.0 | $75929^{(0,5)}$ | 198.8 | - | 9.51 |
| Average | | | | | $28205^{(75,75)}$ | 36801 | 2262.9 | $32888^{(70,154)}$ | 37351 | 2211.0 | $34949^{(30,160)}$ | 106.0 | -1.26 | 5.09 |

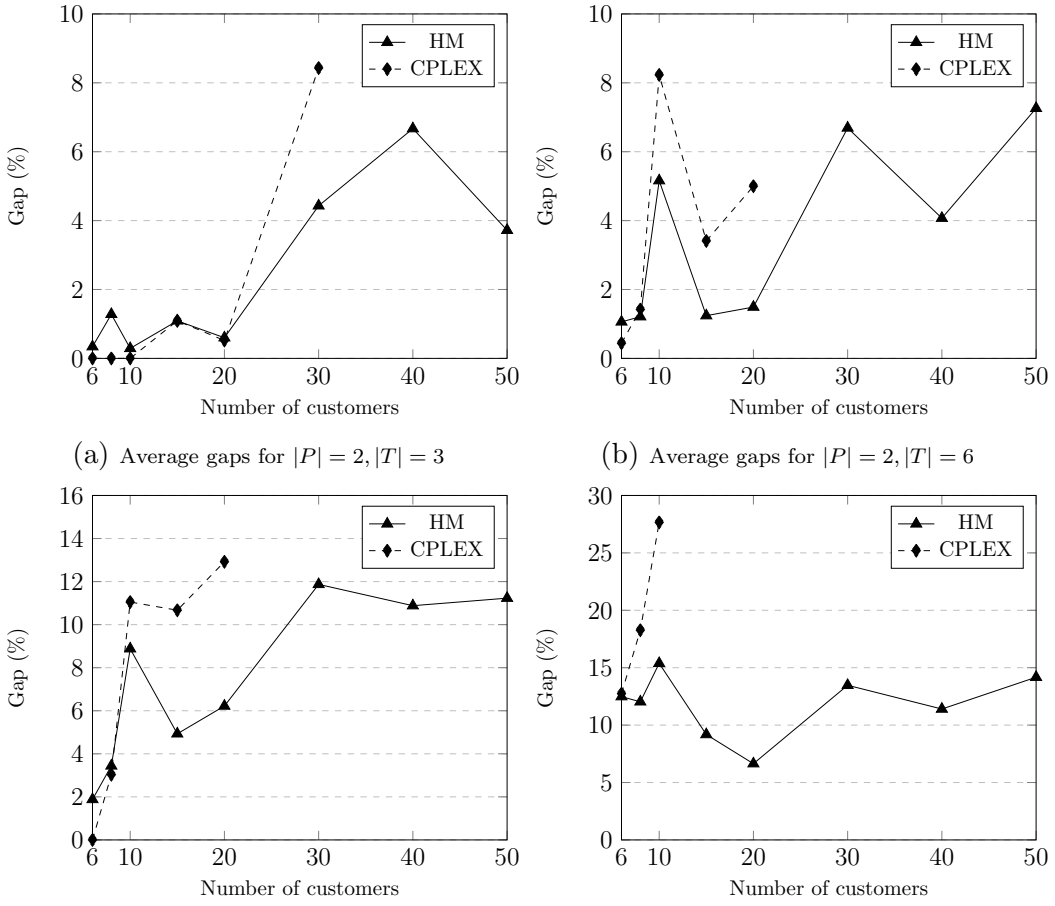
The proved gap between the obtained lower bound by HM and the upper bound by CPLEX is 7.55%, which shows that HM is able to provide good-quality solutions for large-sized instances. In addition, the computation time (463.9s) of HM is only 13.3% that of CPLEX (3478.6s and 3478.8s for CP1 and CP2, respectively) that reaches the time limit in nearly all instances.

To further compare the results of HM and CPLEX, the instances are grouped by the number of plants and periods, yielding 8 sets of 40 instances that only vary in

Table 4.3: Results for instances with up to 4 plants and 50 retailers

| NO. | $m$ | $\|T\|$ | $n$ | $\|B\|$ | CP1 | | | CP2 | | | HM | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | LB1 | UB1 | T1(s) | LB2 | UB2 | T2(s) | LB3 | T3(s) | Dev(%) | Gap(%) |
| 33 | 2 | 3 | 20 | 2 | $54761^{(2,2)}$ | 57152 | 2219.0 | $56984^{(2,5)}$ | 57518 | 2189.5 | $56720^{(1,5)}$ | 13.0 | 0.46 | 0.78 |
| 34 | 2 | 3 | 20 | 3 | $54092^{(4,4)}$ | 52207 | 1040.1 | $52198^{(4,5)}$ | 52255 | 1061.6 | $51969^{(0,5)}$ | 18.2 | 0.40 | 0.42 |
| 35 | 2 | 3 | 30 | 2 | - | 88992 | 3600.0 | $78068^{(0,2)}$ | 90114 | 3600.0 | $84334^{(0,5)}$ | 31.4 | -9.53 | 5.66 |
| 36 | 2 | 3 | 30 | 3 | - | 85332 | 3600.0 | $85749^{(0,1)}$ | 86982 | 3600.0 | $82791^{(0,5)}$ | 33.7 | -3.26 | 3.20 |
| 37 | 2 | 3 | 40 | 2 | - | 123971 | 3600.0 | - | 124843 | 3600.0 | $119072^{(0,5)}$ | 54.4 | - | 4.07 |
| 38 | 2 | 3 | 40 | 3 | - | 107365 | 3600.0 | - | 107660 | 3600.0 | $98585^{(0,5)}$ | 117.3 | - | 9.26 |
| 39 | 2 | 3 | 50 | 2 | - | 163671 | 3600.0 | - | 166863 | 3600.0 | $159198^{(0,5)}$ | 54.6 | - | 2.82 |
| 40 | 2 | 3 | 50 | 3 | - | 150592 | 3600.0 | - | 154425 | 3600.0 | $144022^{(0,5)}$ | 88.2 | - | 4.63 |
| 41 | 2 | 6 | 20 | 2 | - | 121546 | 3600.0 | $117463^{(0,5)}$ | 123986 | 3600.0 | $119899^{(0,5)}$ | 66.3 | -2.21 | 1.43 |
| 42 | 2 | 6 | 20 | 3 | - | 87307 | 3600.0 | $85151^{(0,5)}$ | 88937 | 3600.0 | $86008^{(0,5)}$ | 532.4 | -0.94 | 1.55 |
| 43 | 2 | 6 | 30 | 2 | - | 192300 | 3600.0 | - | 195647 | 3600.0 | $182241^{(0,5)}$ | 329.1 | - | 5.81 |
| 44 | 2 | 6 | 30 | 3 | - | 157951 | 3600.0 | - | 161700 | 3600.0 | $147554^{(0,5)}$ | 489.4 | - | 7.57 |
| 45 | 2 | 6 | 40 | 2 | - | 271684 | 3600.0 | - | 275464 | 3600.0 | $261081^{(0,5)}$ | 358.1 | - | 4.10 |
| 46 | 2 | 6 | 40 | 3 | - | 230860 | 3600.0 | - | 233873 | 3600.0 | $221942^{(0,5)}$ | 394.4 | - | 4.05 |
| 47 | 2 | 6 | 50 | 2 | - | 332460 | 3600.0 | - | 333982 | 3600.0 | $312978^{(0,5)}$ | 1146.1 | - | 6.18 |
| 48 | 2 | 6 | 50 | 3 | - | 282691 | 3600.0 | - | 285856 | 3600.0 | $261105^{(0,5)}$ | 998.9 | - | 8.35 |
| 49 | 4 | 3 | 20 | 2 | - | 59087 | 3600.0 | $53190^{(0,5)}$ | 59703 | 3600.0 | $55353^{(0,5)}$ | 29.3 | -4.24 | 6.46 |
| 50 | 4 | 3 | 20 | 3 | - | 54252 | 3600.0 | $48870^{(0,5)}$ | 55515 | 3600.0 | $51197^{(0,5)}$ | 38.0 | -4.96 | 5.97 |
| 51 | 4 | 3 | 30 | 2 | - | 96742 | 3600.0 | - | 97282 | 3600.0 | $86024^{(0,5)}$ | 59.2 | - | 12.49 |
| 52 | 4 | 3 | 30 | 3 | - | 91630 | 3600.0 | - | 92252 | 3600.0 | $82443^{(0,5)}$ | 77.1 | - | 11.25 |
| 53 | 4 | 3 | 40 | 2 | - | 129389 | 3600.0 | - | 129750 | 3600.0 | $118463^{(0,5)}$ | 138.0 | - | 9.68 |
| 54 | 4 | 3 | 40 | 3 | - | 117026 | 3600.0 | - | 117540 | 3600.0 | $104736^{(0,5)}$ | 169.6 | - | 12.09 |
| 55 | 4 | 3 | 50 | 2 | - | 153618 | 3600.0 | - | 154151 | 3600.0 | $139777^{(0,5)}$ | 158.0 | - | 10.00 |
| 56 | 4 | 3 | 50 | 3 | - | 149981 | 3600.0 | - | 151013 | 3600.0 | $133600^{(0,5)}$ | 315.1 | - | 12.46 |
| 57 | 4 | 6 | 20 | 2 | - | 127986 | 3600.0 | - | 130286 | 3600.0 | $120999^{(0,5)}$ | 108.2 | - | 5.82 |
| 58 | 4 | 6 | 20 | 3 | - | 110590 | 3600.0 | - | 112822 | 3600.0 | $102916^{(0,5)}$ | 748.2 | - | 7.46 |
| 59 | 4 | 6 | 30 | 2 | - | 185478 | 3600.0 | - | 185867 | 3600.0 | $165076^{(0,5)}$ | 800.1 | - | 12.47 |
| 60 | 4 | 6 | 30 | 3 | - | 171635 | 3600.0 | - | 172283 | 3600.0 | $150575^{(0,5)}$ | 1360.6 | - | 14.48 |
| 61 | 4 | 6 | 40 | 2 | - | 260684 | 3600.0 | - | 261262 | 3600.0 | $234972^{(0,5)}$ | 707.1 | - | 11.05 |
| 62 | 4 | 6 | 40 | 3 | - | 231643 | 3600.0 | - | 232505 | 3600.0 | $207322^{(0,5)}$ | 842.0 | - | 11.75 |
| 63 | 4 | 6 | 50 | 2 | - | 283139 | 3600.0 | - | 283615 | 3600.0 | $243823^{(0,5)}$ | 1920.2 | - | 16.76 |
| 64 | 4 | 6 | 50 | 3 | - | 303653 | 3600.0 | - | 304199 | 3600.0 | $272772^{(0,5)}$ | 2647.9 | - | 11.59 |
| Average | | | | | $54427^{(6,6)}$ | 157269 | 3476.8 | $72209^{(6,33)}$ | 158755 | 3476.6 | $145611^{(1,160)}$ | 463.9 | -3.04 | 7.55 |

the number of customers. The average results of solution quality and computation time are shown in Fig. 4.2 and Fig. 4.3, respectively. Fig. 4.2(a) shows that HM obtains near-optimal solutions for instances with up to two plants, three periods and 20 customers, and CPLEX performs well in solving these small-sized instances. Fig. 4.2(b) and 4.2(c) indicate that for instances with up to two plants and 6 periods, HM clearly outperforms CPLEX. The good performance of HM can be further confirmed by the results for instances with up to four plants, as shown in Fig. 4.2(d). CPLEX




(a) Average gaps for $|P| = 2, |T| = 3$

(b) Average gaps for $|P| = 2, |T| = 6$

(c) Average gaps for $|P| = 4, |T| = 3$

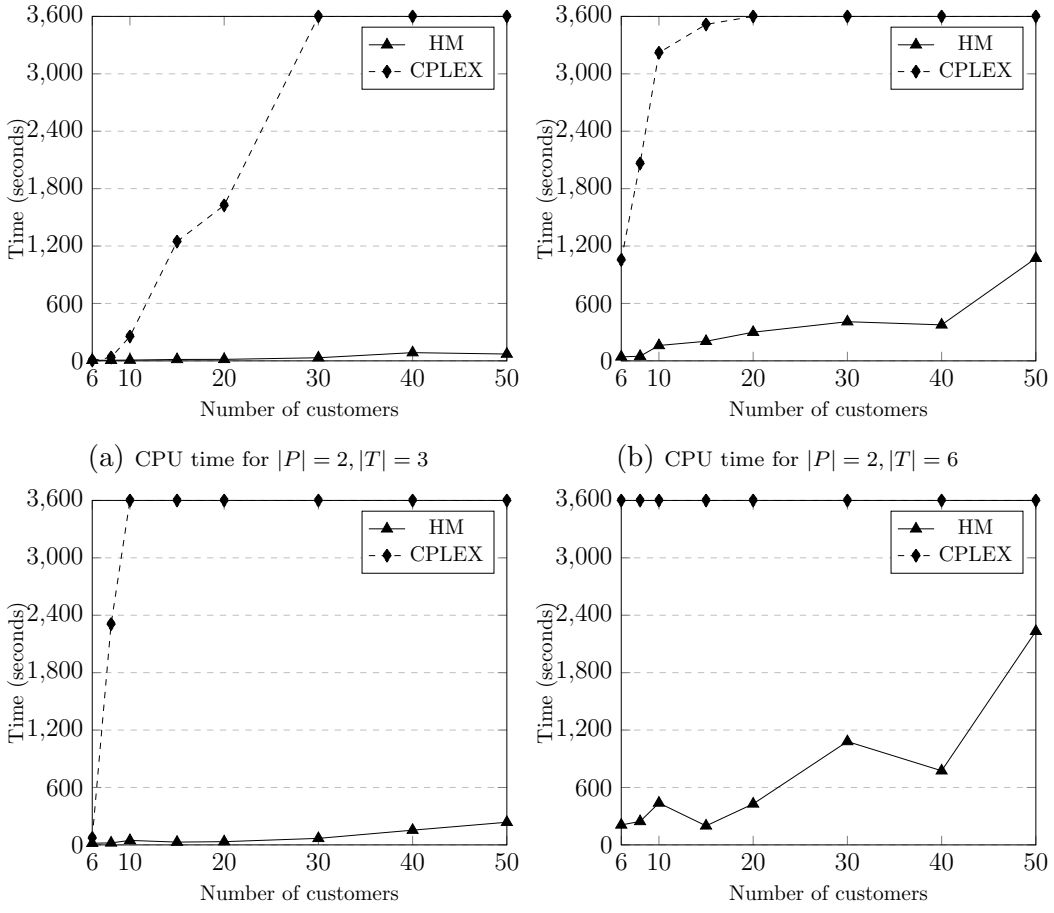
(d) Average gaps for $|P| = 4, |T| = 6$

Fig. 4.2: Comparison on solution quality (gaps(%))

cannot provide any feasible solutions for instances with up to 4 plants, 6 periods and 15 customers. HM can provide feasible solutions for all tested instances. In terms of the computation times, Fig. 4.5 indicates that HM is much faster than CPLEX. The computation times of HM increase relatively slowly while that of CPLEX increases much faster, which mean that HM is more efficient. In summary, numerical results show that HM is able to find near-optimal solutions within very short computation time. It clearly outperforms CPLEX in terms of solution quality and computation time.

### 4.4.3 Impacts of discount policies

In this paper, we consider a realistic situation in which the perishable food products have to be sold with discounts as the residual shelf life reduces. Different discount policies may lead to different profit. To investigate the impact of discount policies on the total profit, we test instances with different discount policies while all other parameters remain the same. The set of instances with $m = 2$ plants, $|T| = 6$




(a) CPU time for $|P| = 2, |T| = 3$

(b) CPU time for $|P| = 2, |T| = 6$

(c) CPU time for $|P| = 4, |T| = 3$

(d) CPU time for $|P| = 4, |T| = 6$

Fig. 4.3: Comparison on computation times

periods, $n = 20$ retailers, and $|B| = 3$ packages is selected. In the instance generation introduced in Section 4.4.1, $\epsilon$ (a parameter that determines the shape of the discount) is set to 1. Taking the instances with $\epsilon = 1$ as a basis, we generate three new sets of instances with $\epsilon$ being 0.5, 1.5 and 2.0, respectively. For a product with shelf life of 3 and initial price of 17, the discount policies for $\epsilon$ being 0.5, 1.0, 1.5 and 2.0 can be illustrated in Fig. 4.4.

These instances are then solved by HM. Upper bounds are generated by CP1 and CP2 with a 3600s time limit, and the obtained best upper bounds are reported. Average results for the set of instances are presented in Table 4.4, in which the first column shows the values of $\epsilon$. The next two columns show the upper bound obtained by CPLEX (including CP1 and CP2) and the lower bound provided by HM, respectively. Columns 4 and 5 give the gap between the lower and upper bounds, and the computation time for HM, respectively. Finally, the revenue and cost components, i.e., production cost, packaging cost, inventory cost and transportation cost, are presented in the last five columns.

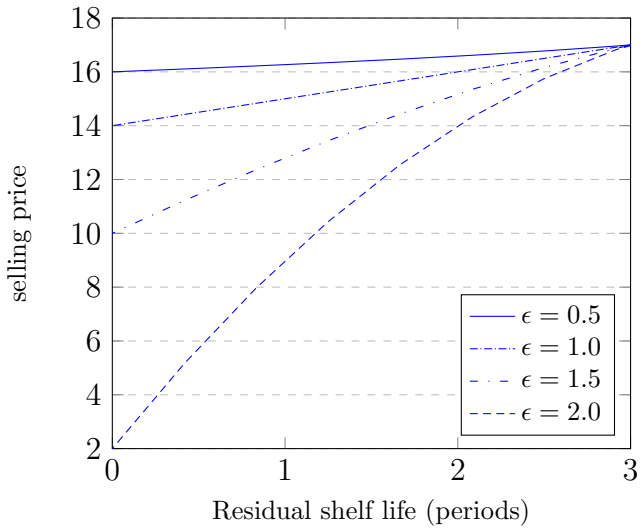



Fig. 4.4: Illustrative example of different discount policies

Table 4.4: Results for MFPRP with different discount policies

| $\epsilon$ | UB | LB (profit) | Gap (%) | Time (s) | Revenue | Procost | Packcost | Invcost | Transcost |
|---|---|---|---|---|---|---|---|---|---|
| 0.5 | 108563 | 105954 | 2.44 | 209.6 | 243930 | 95640 | 22810 | 1765 | 17762 |
| 1.0 | 87307 | 86008 | 1.55 | 532.4 | 230259 | 95702 | 28130 | 903 | 19517 |
| 1.5 | 66537 | 65829 | 1.21 | 114.1 | 221797 | 96020 | 38083 | 403 | 21462 |
| 2.0 | 60879 | 60197 | 1.24 | 47.6 | 254162 | 96785 | 74449 | 188 | 22543 |
| Average | 80821 | 79497 | 1.61 | 225.9 | 237537 | 96037 | 40868 | 815 | 20321 |

We can see from Table 4.4 that HM is able to find near-optimal solutions with an average gap of 1.61% for the 20 instances and an average computation time of only 225.9s. The small gap and short computation time indicate that HM is effective and stable in solving MFPRP with different discount policies. In addition, when the discount policies become steeper, the gaps and computation times generally decrease. Fig. 4.5 shows the changes of total revenue, profit, and different cost components when the discount policies changes. As $\epsilon$ increases (discount becomes steeper), we have the following observations: 1) the total profit obviously reduces; 3) the total revenue decreases at first and then increases; 4) the production, packaging, and transportation costs increase; and 4) the inventory cost decreases. The above observations can be explained by the fact that as the discount becomes steeper: 1) better packages with higher prices are more often used to prevent the food products from decaying; 2) less inventory is held since the selling price goes down quickly; and 3) the plants setup and deliveries to retailers become more frequent to produce and sell fresher food. In summary, the overall results show that discount policies significantly impact the total profit. HM consistently performs well for MFPRP instances with different discount policies.

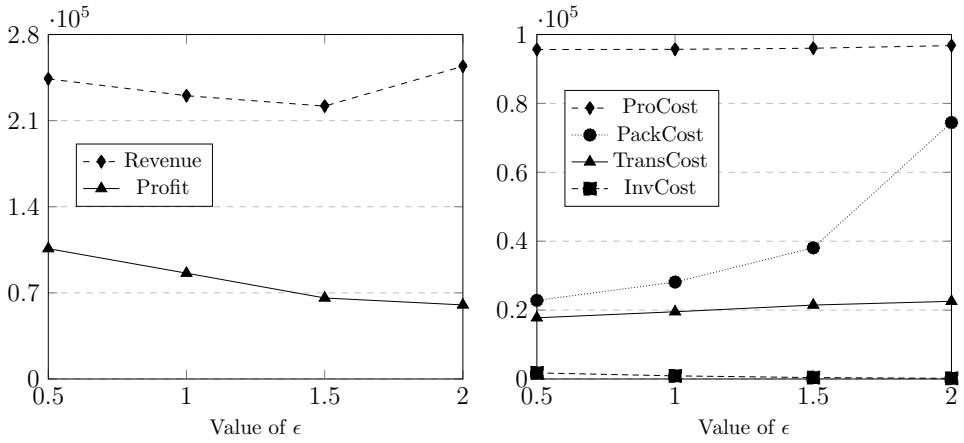



Fig. 4.5: Sensitivity analysis on discount policies

### 4.4.4 Performance analysis of HM

This section analyzes the performance of HM by reporting the improvements obtained by its three components and the corresponding computation times. The 320 instances are grouped by the number of retailers, resulting into 8 sets of 40 instances. The results are shown in Table 4.5 and Fig. 4.6. In Table 4.5 , the first two columns give the instance set number and the number of retailers in each set. Column "feas." shows the number of instances where a feasible solution is found by TI. Columns 4-7 present the average and maximum improvements obtained by FO with respect to TI, and the improvements achieved by RO with respect to FO, respectively. The improvements are calculated with equations (4.63) and (4.64), where obj1, obj2, and obj3 denote the objective values of the solutions provided by TI, FO and RO, respectively. Note that the percentage improvement by FO is calculated on the subset of instances for which a feasible solution has been found by TI since the comparison makes no sense for infeasible ones. The last three columns show the computation times required by the three components of HM, respectively. Fig. 4.6 shows the average and maximum improvements obtained by FO and RO, and the computation times required by the three components.

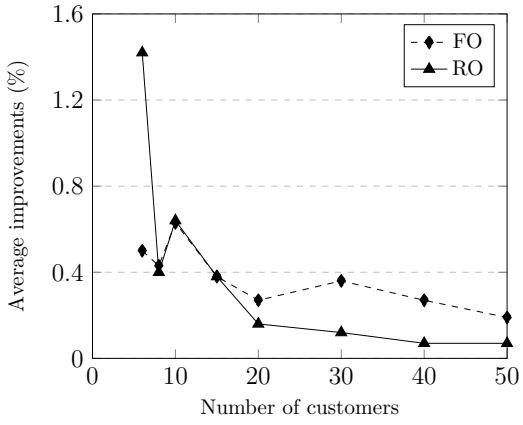
$$\%\,\text{improvement by FO} = \frac{\text{obj2} - \text{obj1}}{\text{obj1}} \times 100\% \tag{4.63}$$

$$\%\,\text{improvement by RO} = \frac{\text{obj3} - \text{obj2}}{\text{obj2}} \times 100\%. \tag{4.64}$$

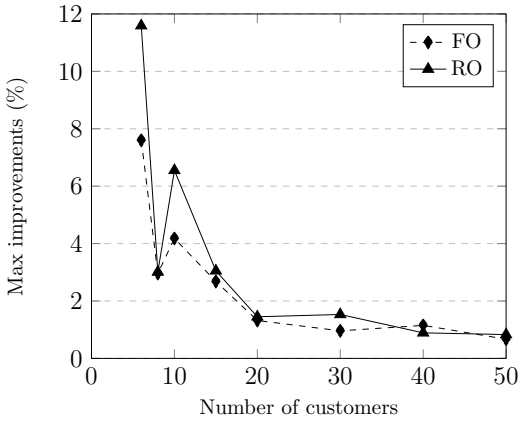
Table 4.5 shows that feasible solutions for 273 out of the 320 instances are provided by TI. The average improvement by FO with respect to TI for these 273 instances is 0.38%, and the average maximum improvement is 2.69%. This indicates that FO can generally improve the feasible solution provided by TI. In addition, FO succeeds to find feasible solutions for all the 320 instances. In terms of RO, an average

Table 4.5: Performance analysis of HM

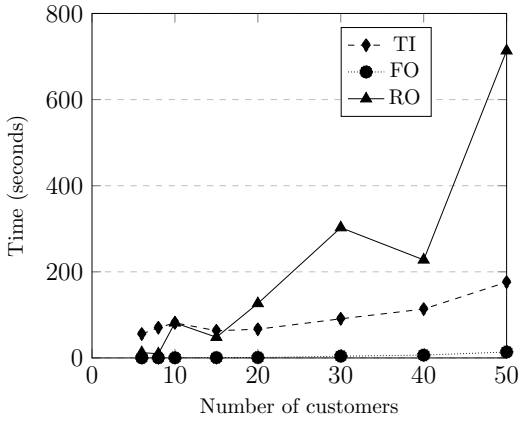
| NO. | $n$ | feas. | % improvement by FO | | % improvement by RO | | T1 (s) | T2 (s) | T3 (s) |
|---|---|---|---|---|---|---|---|---|---|
| | | | Avg. | Max | Avg. | Max | | | |
| 1 | 6 | 40 | 0.50 | 7.60 | 1.42 | 11.59 | 55.6 | 0.3 | 12.5 |
| 2 | 8 | 40 | 0.43 | 2.96 | 0.40 | 3.01 | 70.2 | 0.3 | 9.6 |
| 3 | 10 | 40 | 0.63 | 4.18 | 0.64 | 6.55 | 81.5 | 0.4 | 81.5 |
| 4 | 15 | 40 | 0.38 | 2.68 | 0.38 | 3.05 | 63.2 | 0.7 | 48.1 |
| 5 | 20 | 40 | 0.27 | 1.32 | 0.16 | 1.45 | 66.7 | 1.0 | 126.5 |
| 6 | 30 | 27 | 0.36 | 0.96 | 0.12 | 1.53 | 90.9 | 4.1 | 302.6 |
| 7 | 40 | 27 | 0.27 | 1.15 | 0.07 | 0.89 | 113.5 | 6.5 | 227.7 |
| 8 | 50 | 19 | 0.19 | 0.67 | 0.07 | 0.83 | 176.0 | 13.8 | 726.3 |
| All / Avg. | | 273 | 0.38 | 2.69 | 0.41 | 3.61 | 89.7 | 3.4 | 191.9 |




(a) Average improvements by FO and RO




(b) Maximum improvements by FO and RO




(c) CPU times of the three components

Fig. 4.6: Performance analysis of HM

improvement of 0.41% is achieved with the average maximum improvement being 3.61%, which further demonstrates that the route-based optimization component is effective. The computation times of the TI, FO and RO are 89.7s, 3.4s, and 191.9s, respectively. From Fig. 4.6 (a) and Fig. 4.6 (b), we can observe FO and RO can generally obtain greater improvements for instances with up to 20 customers. The improvements become less obvious as the number of customers increases. In terms of the computation times of the three components, Fig. 4.6 (c) show that for instances with up to 15 retailers, TI consumes the longest computation time among the three components due to the iterative procedure for solving model $\mathcal{P}_1$. While for instances with more than 15 retailers, RO takes longer computation time. This may be due to the large number of routes stored in $\mathcal{R}$ for instances with larger number of retailers, which makes models MBP($\mathcal{K}$) and MBP($\mathcal{K}_l \cup \mathcal{B}_l$) hard to solve. For all the instances, the FO takes very short computation time. Overall, the good performance of HM comes from the hybridization of the three components, i.e., TI, FO, and RO.

## 4.5 Conclusion

This chapter investigates a new multi-plant food production routing problem with perishability and packaging consideration. Firstly, the problem is formulated as a MILP which is later solved by B&C in two different ways. Then a hybrid matheuristic is developed to solve the problem. The heuristic combines a two-phase iterative method, a fix-and-optimize procedure and a route-based optimization process. In particular, the useful information obtained by the components TI and FO is exploited in RO. To performance of HM is evaluated by computational experiments on 320 randomly generated instances. Computational results for instances with up to 4 plants and 15 retailers show that HM can find solutions with average gap of -1.26% and 5.09% compared to the lower and upper bounds provided by CPLEX and needs only less than 5% of the computation time compared with CPLEX. Results for instances with up to 4 plants and 50 retailers indicate that CPLEX cannot solve these large-sized instances within 3600s. HM can propose solution with average gap of 7.55% and average computation time of 460.7s. Analysis on discount policies shows that the total profit is significantly impacted by different discount policies. Performance analysis of HM shows that its good performance is a result of the hybridization of the three components.

The corresponding work is going to be submitted to a journal.

# Chapter 5

# Bi-objective Food Production Routing Problem with Quality Consideration

## 5.1 Introduction

As reviewed in Chapter 2, food quality is one of the most important issues in FSC. Customers expect to receive high-quality food products. However, existing works on FSC optimization always aim to minimize the total cost or to maximize the total profit [39], [41], and quality has rarely been treated as an objective. Nowadays, customers have the highest expectation to receive high-quality food products than ever before. Motivated by the fact that the quality of food received by customers may impact the customer satisfaction which further affects a company's competitiveness. A bi-objective food production routing problem is studied in this chapter. The problem simultaneously optimizes two objectives: 1) to minimize the total production, inventory and routing costs; 2) to maximize the average quality of food received by final customers. For the studied problem, a bi-objective MILP is first formulated. Then an $\epsilon$-constraint-based two-phase iterative heuristic is developed to generate a set of Pareto solutions. In particular, useful information from the solution of the transformed single-objective problem in one iteration is used as input to the next iteration to reduce the computational efforts. Then a fuzzy-logic decision approach is applied to select a preferred solution among the generated Pareto solutions based on decision makers' preferences. Finally, the heuristic is evaluated by conducting numerical experiments on a case study and on 185 randomly generated instances with up to 100 retailers and 12 periods.

The remainder of this chapter is organized as follows. Section 5.2 gives the description and formulation of the considered problem. In Section 5.3, an $\varepsilon$-constraint-based two-phase iterative heuristic is developed to fast solve the problem. Computational

experiments on an illustrative case and randomly generated instances are conducted in Section 5.4. Finally, Section 5.5 summarizes this chapter.

## 5.2 Problem description and formulation

This study considers a BFPRP with quality consideration. The considered problem can be stated as follows.

Consider a complete digraph $G$=$\{N$,$A\}$ with a set of nodes $N$=$\{0, 1, \ldots, n\}$ and a set of arcs $A$=$\{(i, j) : i, j \in N, i \neq j\}$. A plant with limited production and storage capacities is located at node 0. It has a fleet of homogeneous vehicles $K$=$\{1, , 2, \ldots, |K|\}$, each having capacity $V$. A set of $n$ retailers $R$=$\{1, 2, \ldots, n\}$ with a limited storage capacity are geographically located at node $\{1, \ldots, n\}$. Consider a time horizon $T$=$\{1, 2, \ldots, |T|\}$, the customer demand at each retailer is assumed to be deterministic and time varying. A BFPRP consists of simultaneously determining production, storage, delivery, and routing planning to satisfy customer demand with food quality deterioration consideration. Note that food quality can be distinguished by a set of quality levels $Q$=$\{0, 1, \ldots, |Q|\}$, where 0 represents the freshest food. The two objectives of BFPRP are to simultaneously minimize the total production, inventory and routing cost and maximize the average quality of food provided to final customers. The decisions to be made for each period are: 1) how much to produce at the plant; 2) how much to replenish each retailer; 3) how to arrange the transportation routes for the planned deliveries; and 4) how to fulfill customer demand at each retailer.

The study is conducted under the following assumptions: 1) each vehicle's route starts and ends at the plant, and each vehicle can perform at most one trip within each period; 2) each retailer can be visited at most once within each period; i.e., split delivery is not allowed; 3) food quality degrades by one quality unit per period; and 4) once the quality goes beyond $|Q|$, food products can no longer be used to meet customer demand and should be directly turned into waste. The last two assumptions are realistic and are related to food characteristics. The first two assumptions are commonplace in IRP and PRP. They can be justified by the fact that it is always possible to reduce to length of the periods for the assumptions to hold. No split delivery assumption is further justified by the fact that deliveries disturb regular operations at the retailers who therefore prefer few deliveries. To ensure the consistence, we assume that the demand of any retailer in any period does not exceed the sum of the storage and vehicle capacities. To formulate the problem, the following parameters and variables are defined:

Parameters

$c_{ij}$ travel cost on arc $(i,j) \in A$;
$a_t$ unit production cost in period $t \in T$;
$b_t$ production setup cost in period $t \in T$;
$C$ production capacity;
$I_i^{q0}$ initial inventory at $i \in N$ with $q \in Q$;
$D_i^t$ customer demand at retailer $i \in R$ in period $t \in T$;
$U_i$ inventory capacity of $i \in N$;
$h_i^q$ unit inventory holding cost per period at $i \in N$ with $q \in Q$;
$V$ vehicle capacity;

Variables

$\xi_t$ production quantity in period $t$;
$w_t$ binary variable equal to 1 if $\xi_t > 0$; otherwise 0;
$I_i^{qt}$ inventory level with quality $q$ of retailer $i$ at the end of period $t$;
$y_i^{qkt}$ delivery quantity with quality $q$ to retailer $i$ by vehicle $k$ in period $t$;
$\vartheta_i^{qt}$ quantity of food used to fulfill customer demand with quality $q$ at retailer $i$ in period $t$;
$v_i^{kt}$ binary variable equal to 1 if retailer $i$ is visited by vehicle $k$ in period $t$; otherwise 0;
$x_{ij}^{kt}$ binary variable equal to 1 if arc $(i,j)$ is traversed by vehicle $k$ in period $t$; otherwise 0.

With the above description and notation, BFPRP can be formulated as follows:

$$\min f_1 = \sum_{t\in T}(a_t\xi_t + b_t w_t) + \sum_{i\in N}\sum_{q\in Q}\sum_{t\in T} h_i^q I_i^{qt} + \sum_{(i,j)\in A}\sum_{k\in K}\sum_{t\in T} c_{ij}x_{ij}^{kt} \quad (5.1)$$

$$\min f_2 = \sum_{i\in R}\sum_{q\in Q}\sum_{t\in T} \vartheta_i^{qt} q \,/ \sum_{i\in R}\sum_{t\in T} D_i^t \quad (5.2)$$

s.t.

$$I_0^{qt} = I_0^{q-1,t-1} - \sum_{i\in R}\sum_{k\in K} y_i^{qkt}, \forall q \in Q \setminus \{0\}, t \in T \quad (5.3)$$

$$I_0^{0t} = \xi_t - \sum_{i\in R}\sum_{k\in K} y_i^{0kt}, \forall t \in T \quad (5.4)$$

$$I_i^{qt} = I_i^{q-1,t-1} + \sum_{k\in K} y_i^{qkt} - d_i^{qt}, \forall i \in R, q \in Q \setminus \{0\}, t \in T \quad (5.5)$$

$$I_i^{0t} = \sum_{k\in K} y_i^{0kt} - d_i^{0t}, \forall i \in R, t \in T \quad (5.6)$$

$$\xi_t \le C w_t, \forall t \in T \quad (5.7)$$

$$\sum_{q\in Q} I_i^{qt} \le U_i, \forall i \in N, t \in T \quad (5.8)$$

$$\sum_{q\in Q} \vartheta_i^{qt} = D_i^t, \forall i \in R, t \in T \tag{5.9}$$

$$\sum_{i\in R}\sum_{q\in Q} y_i^{qkt} \leq V, \forall k \in K, t \in T \tag{5.10}$$

$$\sum_{q\in Q} y_i^{qkt} \leq V v_i^{kt}, \forall i \in R, k \in K, t \in T \tag{5.11}$$

$$\sum_{k\in K} v_i^{kt} \leq 1, \forall i \in R, t \in T \tag{5.12}$$

$$\sum_{j\in N\setminus\{i\}} x_{ij}^{kt} = \sum_{j\in N\setminus\{i\}} x_{ji}^{kt} = v_i^{kt}, \forall i \in R, k \in K, t \in T \tag{5.13}$$

$$\sum_{i\in R} x_{0i}^{kt} \leq 1, \forall k \in K, t \in T \tag{5.14}$$

$$\sum_{i\in S}\sum_{j\in S} x_{ij}^{kt} \leq |S| - 1, \forall S \subseteq R, |S| \geq 2, k \in K, t \in T \tag{5.15}$$

$$\xi_t \geq 0, \forall t \in T \tag{5.16}$$

$$w_t \in \{0, 1\}, \forall t \in T \tag{5.17}$$

$$I_i^{qt} \geq 0, \forall i \in N, q \in Q, t \in T \tag{5.18}$$

$$\vartheta_i^{qt} \geq 0, \forall i \in R, q \in Q, t \in T \tag{5.19}$$

$$y_i^{qkt} \geq 0, \forall i \in R, q \in Q, k \in K, t \in T \tag{5.20}$$

$$v_i^{kt} \in \{0, 1\}, \forall i \in R, k \in K, t \in T \tag{5.21}$$

$$x_{ij}^{kt} \in \{0, 1\}, \forall (i, j) \in A, k \in K, t \in T \tag{5.22}$$

Objective function (5.1) minimizes the total cost, in which the first summation denotes the fixed and variable production cost, the second and third ones are total inventory cost and total routing cost, respectively. Objective function (5.2) maximizes the average food quality level. Note that the smaller the objective value, the higher the average food quality. Constraints (5.3)-(5.6) indicate the food flow balance at the plant and retailers. Constraints (5.7) and (5.8) are the production and inventory capacity constraints. Constraints (5.9) indicate that customer demand at each retailer must be satisfied. The vehicle capacity constraint is imposed by (5.10). Constraints (5.11) allow positive delivery quantity to a retailer only if it is visited. Constraints (5.12) forbid split delivery. Constraints (5.13) correspond to the vehicle flow conservation. Constraints (5.14) denote that one vehicle can perform at most one route in each period. Constraints (5.15) eliminate all subtours. Constraints (5.16)-(5.22) are nonnegative and integer conditions on decision variables.

The considered BFPRP is NP-hard since it contains a VRP that is well known to be NP-hard [51]. Due to the complexity of the studied problem, a new approach that combines an $\epsilon$-constraint-based two-phase iterative heuristic is developed in next section.

## 5.3 Solution method

As reviewed in Chapter 2, $\epsilon$-constraint method is one of the most effective methods in solving a bi-objective optimization problem. Thus in this section, we present an $\epsilon$-CTIH to solve the considered BFPRP. The basic idea is to optimize the principle objective and transform the other objective into constraints bounded by a parameter $\epsilon$. For our problem, we take the total cost as the principle objective and transform the quality objective into constraints. Thus yielding the mono-objective problem FPRP($\epsilon$). Let $\varphi(\boldsymbol{x})$ and $\omega(\boldsymbol{x})$ denote the two objectives $f_1$ and $f_2$, respectively, where $\boldsymbol{x}$ represents a vector of all decision variables. $\mathcal{X}$ is the feasible region of $\boldsymbol{x}$ defined by constraints (5.3)-(5.22). FPRP($\epsilon$) can be formulated as follows:

$$\min \varphi(\boldsymbol{x}) \tag{5.23}$$

s.t.

$$\omega(\boldsymbol{x}) \leq \epsilon \tag{5.24}$$

$$\boldsymbol{x} \in \mathcal{X} \tag{5.25}$$

In order to construct the set of Pareto-optimal solution, we need to know the set of all possible values of $\epsilon$, which is actually an interval. This interval can be determined by obtaining an Ideal point $(f_1^I, f_2^I)$ and a Nadir point $(f_1^N, f_2^N)$. They are obtained by exactly solving the following mono-objective problems:

$$f_1^I = \min\{\varphi(\boldsymbol{x}) | \boldsymbol{x} \in \mathcal{X}\} \tag{5.26}$$

$$f_2^I = \min\{\omega(\boldsymbol{x}) | \boldsymbol{x} \in \mathcal{X}\} \tag{5.27}$$

$$f_1^N = \min\{\varphi(\boldsymbol{x}) | \omega(\boldsymbol{x}) = f_2^I, \boldsymbol{x} \in \mathcal{X}\} \tag{5.28}$$

$$f_2^N = \min\{\omega(\boldsymbol{x}) | \varphi(\boldsymbol{x}) = f_1^I, \boldsymbol{x} \in \mathcal{X}\} \tag{5.29}$$

the value of $\epsilon$ can be bounded by interval $[f_2^I, f_2^N]$. Then the step size $\Delta$ should be fixed to explore values of $\epsilon$ and form a series of mono-objective problems that are solved iteratively to obtain the Pareto front or its approximation. The solution to each mono-objective FPRP($\epsilon_m$) for a given value of parameter $\epsilon$, if not dominated, is a Pareto-optimal solution to the original problem. The objective values of all Pareto solutions $(\varphi(\boldsymbol{x}), \omega(\boldsymbol{x}))$ form a Pareto front. Ideally, we expect to generate the exact Pareto front by exactly solving a series of FPRP($\epsilon$) with enumerating the possible value of $\epsilon$. This would require solving a large number of mono-objective problems due to the continuous nature of parameter $\epsilon$, which is impractical and unnecessary for decision makers. In addition, it is computationally challenging to optimally solve a complex mono-objective FPRP($\epsilon$). In practice, decision makers may expect some representative Pareto solutions within a reasonable amount of computation time. This

motivates us to design the $\epsilon$-CTIH to obtain an approximate Pareto front. Method $\epsilon$-CTIH consists of solving a set of $M = 1, 2, \ldots, |M|$ mono-objective FPRP($\epsilon$) for $|M|$ values of parameter $\epsilon$. Next, we present a two-phase iterative heuristic (TIH) to solve the $m^{th}$ mono-objective problem FPRP($\epsilon_m$) with $m \in M$.

### 5.3.1 Two-phase iterative heuristic for FPRP($\epsilon_m$)

The key to $\epsilon$-CTIH is to efficiently solve a series of mono-objective FPRP($\epsilon_m$) by decomposing it into a lot-sizing subproblem and some independent vehicle-routing problems (VRPs). This decomposition is necessary due to the complexity of FPRP($\epsilon_m$) with realistic size. More specifically, FPRP($\epsilon_m$) is first decomposed into a variant of lot-sizing problem FPDP($\epsilon_m, \boldsymbol{\delta}^m, \boldsymbol{\lambda}^m$) and VRP($\epsilon_m$), where $\boldsymbol{\delta}^m$ is a matrix with entries $\delta_{it}^m$ ($i \in R$ and $t \in T$) and $\boldsymbol{\lambda}^m$ is a vector with entries $\lambda_t^m$.

FPDP($\epsilon_m, \boldsymbol{\delta}^m, \boldsymbol{\lambda}^m$) determines the quantity of food of each quality level to deliver to each customer in each period without considering the detailed routes of individual vehicles. These delivery quantities are then disaggregated into the quantity to be transported to each retailer in each period by each individual vehicle by solving the vehicle routing problem VRP($\epsilon_m$). In order that the decomposition leads to a solution close to an optimal one and to ensure existence of a feasible disaggregation, the lot-sizing problem must take into account, in an aggregated way, the delivery cost and the transportation capacity. The delivery cost is calculated by estimating the cost of visiting each customer $i \in R$ in each period $t \in T$, namely $\delta_{it}^m$. The aggregate transportation capacity is calculated by taking into account the fill rate of the fleet in each period $t \in T$. This is done by introducing parameter $\lambda_t^m$. On the other hand, in order for the aggregation to be realistic, these two subproblems FPDP($\epsilon_m, \boldsymbol{\delta}^m, \boldsymbol{\lambda}^m$) and VRP($\epsilon_m$) are solved sequentially and iteratively until a near-optimal solution is found. The sets of parameters $\boldsymbol{\delta}^m$ and $\boldsymbol{\lambda}^m$ are updated at each iteration using the solution to VRP($\epsilon_m$) obtained in the previous iteration. The updating of $\boldsymbol{\delta}^m$ is described in detail later. Parameter $\lambda_t^m \in (0, 1]$ for each $t \in T$ is initially set to 1, and is decreased when the routing phase is infeasible. If there is no feasible solution to VRP($\epsilon_m$) for some period $t$, then $\lambda_t^m$ is decreased by $\tau \in (0, 1)$. Note that this idea has been used in [1], [4], [19].

To be specific, FPDP($\epsilon_m, \boldsymbol{\delta}^m, \boldsymbol{\lambda}^m$) is obtained from FPRP($\epsilon_m$) by eliminating route-related variables $x_{ij}^{kt}$'s and constraints (5.13)-(5.15) and (5.22). Vehicle-related index $k$ is dropped from vehicle-related variables $y_i^{qkt}$ and $v_i^{kt}$ by considering an aggregated vehicle with aggregated capacity. FPDP($\epsilon_m, \boldsymbol{\delta}^m, \boldsymbol{\lambda}^m$) can be formulated as follows:

$$\min f_1 = \sum_{t \in T} (a_t \xi_t + b_t w_t) + \sum_{i \in N} \sum_{q \in Q} \sum_{t \in T} h_i^q I_i^{qt} + \sum_{(i,j) \in A} \sum_{t \in T} \delta_{it}^m v_i^t \qquad (5.30)$$

s.t.

$$(5.7)-(5.9),(5.16)-(5.19), \text{and}$$

$$\sum_{i\in R}\sum_{q\in Q}\sum_{t\in T}\vartheta_i^{qt}q \,/\, \sum_{i\in R}\sum_{t\in T}D_i^t \le \epsilon_m \tag{5.31}$$

$$I_0^{qt} = I_0^{q-1,t-1} - \sum_{i\in R} y_i^{qt}, \forall q \in Q\setminus\{0\}, t\in T \tag{5.32}$$

$$I_0^{0t} = \xi_t - \sum_{i\in R} y_i^{0t}, \forall t \in T \tag{5.33}$$

$$I_i^{qt} = I_i^{q-1,t-1} + y_i^{qt} - \vartheta_i^{qt}, \forall i\in R, q\in Q\setminus\{0\}, t\in T \tag{5.34}$$

$$I_i^{0t} = y_i^{0t} - d_i^{0t}, \forall i\in R, t\in T \tag{5.35}$$

$$\sum_{i\in R}\sum_{q\in Q} y_i^{qt} \le \lambda_t^m |K|V, \forall t\in T \tag{5.36}$$

$$\sum_{q\in Q} y_i^{qt} \le V v_i^t, \forall i\in R, t\in T \tag{5.37}$$

$$y_i^{qt} \ge 0, \forall i\in R, q\in Q, t\in T \tag{5.38}$$

$$v_i^t \in \{0,1\}, \forall i\in R, t\in T \tag{5.39}$$

In this formulation, decision variable $v_i^t = 1$ if retailer $i$ is visited in period $t$, and 0 otherwise. Variable $y_i^{qt}$ is the delivery quantity with quality $q$ to retailer $i$ in period $t$. Other variables keep the same meaning as in BFPRP. Objective function (5.30) minimizes the sum of production, inventory and approximate visiting costs. Constraints (5.31) mean that the average quality is restricted by $\epsilon_m$. Constraints (5.32)-(5.35) are similar to (5.3)-(5.6). Constraints (5.36) ensure that the total delivery quantity to all retailers within one period cannot exceed the total vehicle capacity with the expected fill rate. Constraints (5.37) ensure the delivery quantity to be 0 if a retailer is not visited. Nonnegative and binary variables are defined by (5.38) and (5.39). For each iteration of TIH, once FPDP($\epsilon_m, \boldsymbol{\delta}^m, \boldsymbol{\lambda}^m$) is solved , we obtain values of all variables except those related to vehicle routes.

From the obtained solution to FPDP($\epsilon_m, \boldsymbol{\delta}^m, \boldsymbol{\lambda}^m$), we can form a set of retailers $\mathcal{V}^t$ to be visited in period $t\in T$; i.e., $\mathcal{V}^t = \{i\in R | v_i^t = 1\}$. For retailer $i\in\mathcal{V}^t$, let $\mathcal{D}_i^t$ denote the corresponding delivery quantity; i.e., $\mathcal{D}_i^t = \sum_{q\in Q} y_i^{qt}$. Then solving VRP($\epsilon_m$) is equivalent to solve a series of independent VRPs, one for each period $t\in T$ such that with $\mathcal{V}^t \neq \emptyset$. Such a problem is denoted VRP($t, \epsilon_m$). By definition, VRP($t, \epsilon_m$) can be formulated as follows:

$$\min \sum_{(i,j)\in A}\sum_{k\in K} c_{ij}x_{ij}^k \tag{5.40}$$

s.t.

$$\sum_{i\in R} \mathcal{D}_i^t v_i^k \le V, \forall k\in K \tag{5.41}$$

$$\sum_{k\in K} v_i^k = 1, \forall i \in \mathcal{V}^t \tag{5.42}$$

$$\sum_{j\in\mathcal{V}^t\cup\{0\}\setminus\{i\}} x_{ij}^k = \sum_{j\in\mathcal{V}^t\cup\{0\}\setminus\{i\}} x_{ji}^k = v_i^k, \forall i \in \mathcal{V}^t, k \in K \tag{5.43}$$

$$\sum_{i\in\mathcal{V}^t} x_{0i}^k \leq 1, \forall k \in K \tag{5.44}$$

$$\sum_{i\in S}\sum_{j\in S} x_{ij}^k \leq |S| - 1, \forall S \subseteq \mathcal{V}^t, |S| \geq 2, k \in K \tag{5.45}$$

$$v_i^k \in \{0,1\}, \forall i \in \mathcal{V}^t, k \in K \tag{5.46}$$

$$x_{ij}^k \in \{0,1\}, \forall i \in \mathcal{V}^t, j \in \mathcal{V}^t \setminus \{i\}, k \in K \tag{5.47}$$

Objective function (5.40) minimizes the total routing cost. Constraints (5.41)-(5.45) are the vehicle routing constraints. Constraints (5.46) and (5.47) define binary variables. FPDP($\epsilon_m, \boldsymbol{\delta}^m, \boldsymbol{\lambda}^m$) and VRP($\epsilon_m$) are solved iteratively by TIH. The solutions to these two problems form a feasible solution to FPRP($\epsilon_m$).

Then the visiting costs $\boldsymbol{\delta}^m$ are updated as follows: if retailer $i \in R$ is visited in period $t$, $\delta_{it}^m = c_{i^-i} + c_{i^+i} - c_{i^-i^+}$, where $i^-$ and $i^+$ denote the predecessor and successor nodes of $i$ in the solution to the vehicle routing problem VRP($t, \epsilon_m$) obtained in the previous iteration, respectively. Otherwise, $\delta_{it}^m$ is updated as the minimum insertion cost to one of the routes performed in period $t$. If the method falls into local optima, we introduce a diversification mechanism to restart the algorithm when the incumbent solution is not improved after $N^D$ iterations. The number of iterations of TIH is limited to $N^T$. Once the algorithm stops, we output the best incumbent solution, its objective value $f_1^m$, the corresponding value of $f_2^m$, and the corresponding value of the two sets of parameters $\boldsymbol{\delta}^{*m}$ and $\boldsymbol{\lambda}^{*m}$. Algorithm 5.1 describes the detailed steps of TIH.

In the main loop (line 3 to 19), FPDP($\epsilon_m, \boldsymbol{\delta}^m, \boldsymbol{\lambda}^m$) and VRP($\epsilon_m$) are solved sequentially. A diversification mechanism is introduced to re-start the algorithm (line 15 to 17). Other lines in algorithm 1 are self-explained.

### 5.3.2 $\epsilon$-constraint-based two-phase iterative heuristic

In this subsection, we depict the general framework of $\epsilon$-CTIH to solve BFPRP. Firstly, objective $f_2$ is transformed into a constraint with an upper bound $\epsilon$ to form FPRP($\epsilon$). To determine the range of $\epsilon$, we should first compute the interval $[f_2^I, f_2^N]$. Preliminary tests show that it is very time consuming to exactly solve (5.26)-(5.28). TIH is used to solve (5.26)-(5.28) to obtain near-optimal solutions and the corresponding objective values are denoted as $f_1^{AI}$ , $f_2^{AI}$ and $f_1^{AN}$, respectively. Because we use TIH which is a heuristic to solve FPRP($\epsilon$), we can only obtain an approximate value of $f_1^I$, denoted as $f_1^{AI}$. Therefore, the constraint $\varphi(\boldsymbol{x}) = f_1^I$ cannot be taken

**Algorithm 5.1** TIH for solving FPRP($\epsilon_m$)

1. Initialize $f_1^m \leftarrow +\infty, j \leftarrow 0$
2. Input $N^T$, $N^D$, $\epsilon_m, \boldsymbol{\lambda}^m$, $\tau$ and $\boldsymbol{\delta}^m$
3. **While** $(j < N^T)$ **do**
4. Solve FPDP($\epsilon_m, \boldsymbol{\delta}^m, \boldsymbol{\lambda}^m$) to obtain total production and inventory cost $PC$ , get $\mathcal{V}^t$ and $\mathcal{D}_i^t$
5. Solve VRP($t, \epsilon_m$) for all $t \in T$ such that $\mathcal{V}^t \neq \emptyset$ to get route $\mathcal{R}^{kt}$, for all $k \in K$, number of routes $\mathcal{N}^t$, and total routing cost $RC$
6. **If** there exists $\mathcal{N}^t > |K|$ **then**
7. Set $\lambda_t^m = \lambda_t^m - \tau$ and goto step 4
8. **Else if** $PC + RC < f_1^m$ **then**
9. Set $\boldsymbol{\delta}^{*m} \leftarrow \boldsymbol{\delta}^m$, $\boldsymbol{\lambda}^{*m} \leftarrow \boldsymbol{\lambda}^m$, $f_1^m = PC + RC$
10. Calculate $f_2^m$ with (5.2) and set $l = 0$
11. **Else**
12. set $l = l + 1$
13. **End if**
14. Update $\boldsymbol{\delta}^m$
15. **If** $l \geq N^D$ **then**
16. Diversify by setting $\delta_{it}^m \leftarrow \theta(c_{0i} + c_{i0})$ for $i \in R$ and $t \in T$, set $l \leftarrow 0$
17. **End if**
18. Set $j \leftarrow j + 1$
19. **End while**
20. Output $f_1^m, f_2^m, \boldsymbol{\delta}^{*m}$ and $\boldsymbol{\lambda}^{*m}$

into account to calculate $f_2^N$. Thus we use an alternative way to obtain the value $f_2^{AN}$ by calculating it directly with (5.2) based on the obtained solution to (5.26). The approximate Ideal point $(f_1^{AI}, f_2^{AI})$, approximate Nadir point $(f_1^{AN}, f_2^{AN})$, and interval $[f_2^{AI}, f_2^{AN}]$ of $\epsilon$ are formed.

By varying the value of $\epsilon$ in $[f_2^{AI}, f_2^{AN}]$, a set of $|M|$ mono-objective FPRP($\epsilon_m$) are solved by calling TIH. As mentioned above, for the $m^{th}$ iteration, TIH consists of sequentially and iteratively solving two subproblems FPDP($\epsilon_m, \boldsymbol{\delta}^m, \boldsymbol{\lambda}^m$) and VRP($\epsilon_m$) to obtain a near-optimal solution to FPRP($\epsilon_m$). The sets of parameters $\boldsymbol{\delta}^m$ and $\boldsymbol{\lambda}^m$ play a central role in TIH. Good initial values for these parameters may greatly improve the performance of TIH. In $\epsilon$-CTIH, the set of mono-objective FPRP($\epsilon_m$) differs from each other only in (5.31) by changing the value of $\epsilon$ with a step size $\Delta$. Therefore, the proper parameter settings of FPRP($\epsilon_m$) may also be used in solving FPRP($\epsilon_{m+1}$). For this purpose, in addition to outputting the best feasible solution to FPRP($\epsilon_m$), TIH outputs the corresponding values of $f_2^{*m}$, $\boldsymbol{\delta}^{*m}$ and $\boldsymbol{\lambda}^{*m}$. In the $(m+1)^{th}$ iteration of $\epsilon$-CTIH to solve FPRP($\epsilon_{m+1}$), the initial values are set as follows: $\epsilon_{m+1} = f_2^{*m} - \Delta$, $\boldsymbol{\delta}^{m+1} = \boldsymbol{\delta}^{*m}$ and $\boldsymbol{\lambda}^{m+1} = \boldsymbol{\lambda}^{*m}$. Note that these settings contain useful information of FPRP($\epsilon_m$) to avoid redundant iterations when solving FPRP($\epsilon_{m+1}$) and serve as a good starting point for FPRP($\epsilon_{m+1}$). Our heuristic links

FPRP($\epsilon_m$) to FPRP($\epsilon_{m+1}$) with $\epsilon, \boldsymbol{\delta}$, and $\boldsymbol{\lambda}$ while classic $\epsilon$-constraint methods only use $\epsilon$ as a link. The general framework of $\epsilon$-CTIH is outlined in Algorithm 5.2.

**Algorithm 5.2** $\epsilon$-CTIH for solving BFPRP

1. Initialize $m \leftarrow 0, \delta_{it}^0 \leftarrow c_{0i} + c_{i0}, \forall i \in R, \forall t \in T$, $\lambda_t^0 \leftarrow 1, \forall t \in T$
2. Call TIH to solve (5.26) to get $f_1^{AI}$, $\boldsymbol{\delta^{*0}}$ and $\boldsymbol{\lambda^{*0}}$, and calculate $f_2^{AN}$ with the solution to (5.26)
3. Call TIH to solve (5.27) and (5.28) to get $f_2^{AI}$ and $f_1^{AN}$
4. Set $\mathcal{A}_F \leftarrow \{(f_1^{AI}, f_2^{AN}), (f_1^{AN}, f_2^{AI})\}$, $\epsilon_m \leftarrow f_2^{AN}$
5. **While** $(\epsilon_m > f_2^{AI})$ **do**
6. Set $m \leftarrow m + 1$, $\epsilon_m \leftarrow \epsilon_{m-1} - \Delta, \boldsymbol{\delta}^m \leftarrow \boldsymbol{\delta}^{*m-1}$, $\boldsymbol{\lambda^m} \leftarrow \boldsymbol{\lambda}^{*m-1}$
7. Call TIH to solve FPRP($\epsilon_m$) with $\boldsymbol{\delta}^m$ and $\boldsymbol{\lambda^m}$, and output $(f_1^m, f_2^m), \boldsymbol{\delta}^{*m}$ and $\boldsymbol{\lambda}^{*m}$
8. Set $\mathcal{A}_F \leftarrow \mathcal{A}_F \cup (f_1^m, f_2^m)$
9. **End while**
10. Remove the dominated points from $\mathcal{A}_F$ and return $\mathcal{A}_F$

Having a set $S$ of Pareto solutions at hand, a fuzzy logic decision method can be used to help decision makers choose a best-compromised solution according to their preference. The fuzzy logic decision method has been introduced in Chapter 2. The preferred solution can be obtained by using equations (2.15) and (2.16) for each combination of objective weights.

## 5.4 Computational experiments

This section presents the computational experiments conducted on a case study to illustrate the performance of our algorithm and on 185 randomly generated instances to show the effectiveness and efficiency of the proposed approach. We compile the algorithms in C++ by using Microsoft Visual Studio 2010 linked with CPLEX version 12.6.0. All runs are performed on a personal computer with CORE CPU 2.5 GHz and 8 GB RAM.

### 5.4.1 Case study

A case is derived from a real-life fresh-meat product logistics network. A food company produces and distributes a single type of fresh-meat product to its 40 owned retail stores with 3 homogeneous vehicles. Detailed data of all parameters can be found in Tables 5.1 and 5.2. Decision makers have to make an integrated plan for the next week (7 days) indicating how much to produce on each day, how much to deliver to each retailer on each day, how to arrange proper routes for the planned deliveries, and at the same time, guide the retailers to sell meat products with different quality to their final customers.

Table 5.1: Data for the case study

| ID | $I_0$ | $D_1$ | $D_2$ | $D_3$ | $D_4$ | $D_5$ | $D_6$ | $D_7$ | ID | $I_0$ | $D_1$ | $D_2$ | $D_3$ | $D_4$ | $D_5$ | $D_6$ | $D_7$ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 177 | 80 | 72 | 64 | 79 | 74 | 79 | 69 | 21 | 171 | 73 | 86 | 79 | 75 | 74 | 80 | 81 |
| 2 | 188 | 61 | 67 | 73 | 84 | 74 | 80 | 82 | 22 | 144 | 82 | 61 | 79 | 82 | 84 | 82 | 66 |
| 3 | 163 | 80 | 70 | 72 | 81 | 79 | 65 | 67 | 23 | 174 | 86 | 81 | 86 | 77 | 77 | 78 | 73 |
| 4 | 151 | 67 | 62 | 80 | 62 | 80 | 70 | 73 | 24 | 169 | 76 | 66 | 90 | 74 | 79 | 78 | 81 |
| 5 | 174 | 76 | 64 | 78 | 71 | 84 | 80 | 82 | 25 | 154 | 89 | 72 | 72 | 69 | 81 | 74 | 85 |
| 6 | 185 | 64 | 67 | 64 | 74 | 70 | 67 | 71 | 26 | 161 | 85 | 85 | 90 | 72 | 73 | 76 | 72 |
| 7 | 154 | 77 | 79 | 73 | 71 | 71 | 77 | 67 | 27 | 173 | 69 | 87 | 65 | 82 | 86 | 67 | 86 |
| 8 | 183 | 87 | 82 | 70 | 81 | 73 | 72 | 80 | 28 | 158 | 67 | 85 | 77 | 76 | 75 | 77 | 77 |
| 9 | 187 | 88 | 78 | 75 | 71 | 76 | 64 | 89 | 29 | 168 | 78 | 69 | 74 | 71 | 83 | 65 | 65 |
| 10 | 170 | 60 | 88 | 68 | 70 | 82 | 63 | 62 | 30 | 157 | 76 | 70 | 76 | 75 | 89 | 70 | 80 |
| 11 | 154 | 85 | 73 | 76 | 77 | 71 | 63 | 82 | 31 | 148 | 73 | 70 | 73 | 80 | 73 | 75 | 71 |
| 12 | 178 | 81 | 81 | 77 | 77 | 74 | 76 | 78 | 32 | 178 | 65 | 65 | 77 | 68 | 81 | 64 | 83 |
| 13 | 184 | 77 | 82 | 76 | 84 | 70 | 71 | 87 | 33 | 181 | 88 | 67 | 82 | 70 | 78 | 73 | 62 |
| 14 | 147 | 62 | 70 | 73 | 81 | 75 | 78 | 71 | 34 | 168 | 68 | 83 | 78 | 77 | 72 | 75 | 73 |
| 15 | 146 | 84 | 83 | 90 | 87 | 78 | 80 | 74 | 35 | 185 | 69 | 77 | 82 | 71 | 72 | 70 | 65 |
| 16 | 147 | 79 | 79 | 77 | 68 | 74 | 75 | 90 | 36 | 180 | 71 | 67 | 73 | 77 | 81 | 64 | 71 |
| 17 | 177 | 78 | 84 | 75 | 76 | 83 | 87 | 61 | 37 | 176 | 63 | 73 | 77 | 72 | 80 | 78 | 81 |
| 18 | 155 | 71 | 79 | 63 | 83 | 76 | 64 | 73 | 38 | 177 | 83 | 71 | 62 | 67 | 82 | 66 | 66 |
| 19 | 143 | 65 | 64 | 75 | 77 | 88 | 74 | 66 | 39 | 167 | 64 | 87 | 76 | 66 | 77 | 70 | 76 |
| 20 | 178 | 82 | 76 | 74 | 81 | 71 | 74 | 68 | 40 | 188 | 67 | 80 | 64 | 83 | 76 | 66 | 75 |

Table 5.2: Data for the case study

| | |
|---|---|
| Production Capacity: | 60000 Kg; |
| Inventory capacity at Production site: | 40000 Kg; |
| Initial inventory at production site: | 500 Kg; |
| Vehicle capacity: | 6000 Kg; |
| Fixed production setup cost: | 2000 RMB; |
| Variable production cost: | 20 RMB/Kg; |
| Unit inventory cost at production site: | 0.08 RMB/Kg/day; |
| Unit inventory cost at all retailers: | 0.12 RMB/Kg/day; |
| Unit transportation cost: | 30 RMB/Km; |
| Quality level range: | [0, 3]. |

Currently, the company applies a three-phase heuristic to generate the production and distribution plan for the following week. The daily production quantity is first determined based on the demand from each retailer to minimize the production cost by assuming products are stored only at the production site. The distribution plan is then calculated to minimize the total inventory cost. Vehicle routes are generated last based on the planned deliveries. The three-phase heuristic generates a plan with a total cost of 309,260 RMB and an average quality level of 2.00. Note that the quality level is an average value of all product quality sold to final customers. In the case study, the quality level can be interpreted as the average number of days a product has been stored before it is sold.

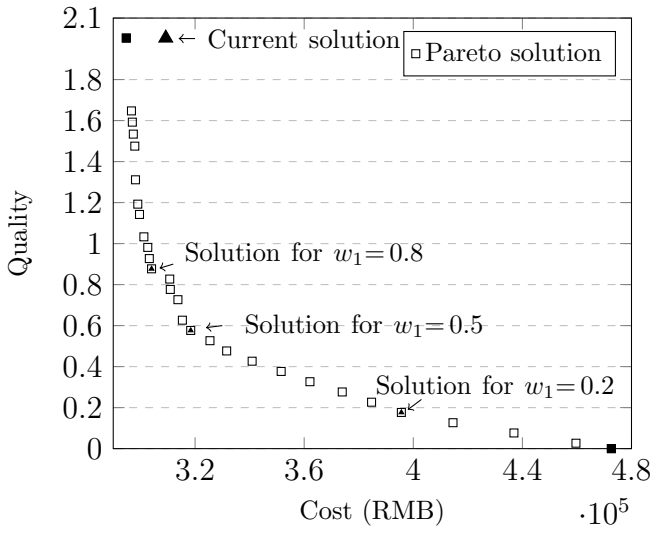



Fig. 5.1: Approximate Preto front for the case

#### 5.4.1.1 Model solution

CPLEX is unable to solve the studied case with guaranteed optimality due to its large size. Therefore, we use the proposed model and approach to obtain a set of Pareto solutions for decision makers. The obtained Pareto solutions (dominated solutions have been removed) are shown in Fig. 5.1, where the horizontal and vertical axes denote the cost and quality, respectively. Having a set of Pareto solutions at hand, we can directly provide all these solutions to decision makers. Besides, we can also help them select a best-compromised solution. As can be seen from Fig. 5.1 that we can obtain 29 Pareto solutions for the studied case. The computation time is 1741.7s. The obtained Pareto front shows a clear conflict between these two objectives. The two points formed by the approximate lower and upper bounds of the two objectives are shown in solid squares. The solution generated by the currently-used three-phase heuristic is shown as a solid triangle. It is obvious that the solution provided by the currently used three-phase heuristic is dominated by some solutions obtained by our heuristic. The solution ($f_1^{AI}$, $f_2^{AN}$) obtained by our $\epsilon$-CTIH, as compared to the solution obtained by the three-phase heuristic, can reduce the total cost by 10.77%.

#### 5.4.1.2 Results and discussion

To help decision makers select their preferred solutions, we assume the decision makers have 3 different preferences based on different situations. In the first situation, they focus more on cost and the weight of the first objective is set to $w_1 = 0.8$. In the second situation, they treat two objectives equally, i.e., $w_1 = 0.5$. While in the last situation, they focus more on quality, i.e., $w_1 = 0.2$. The three selected solutions are shown in Fig. 5.1, from which we can see that different weights may lead to different Pareto

solutions. Detailed results of the selected solutions are shown in Table 5.3, where the first four columns are self-explained. Column $\Delta f_1$ gives the cost increase compared with $f_1^{AI}$; i.e., $\Delta f_1 = (F_1 - f_1^{AI})/f_1^{AI}$ and $\Delta f_2$ denotes the quality improvement compared with $f_2^{AN}$, i.e., $\Delta f_2 = (f_2^{AN} - F_2)/f_2^{AN}$. The next three columns correspond to the production, inventory and transportation costs, respectively. The last column indicates the number of production setups.

Table 5.3: Detailed information for the selected solutions

| $w_1$ | $\eta^{max}$ | $F_1$ | $F_2$ | $\Delta f_1(\%)$ | $\Delta f_2(\%)$ | $Pcost$ | $Icost$ | $Tcost$ | $S$ |
|---|---|---|---|---|---|---|---|---|---|
| 0.8 | 0.87 | 304050 | 0.88 | 3.14 | 56.21 | 281942 | 2162 | 19946 | 3 |
| 0.5 | 0.79 | 318404 | 0.58 | 8.01 | 71.21 | 289511 | 1516 | 27377 | 5 |
| 0.2 | 0.82 | 395631 | 0.18 | 34.21 | 91.19 | 363580 | 2614 | 29437 | 6 |

From Table 5.3, we can see that the membership values range from 0.79 to 0.87, which are relatively high. If decision makers focus more on cost, the selected solution shows 3.14% cost increase and 56.21% quality improvement. If they focus more on quality, the selected solution shows 34.21% cost increase and 91.19% quality improvement. We can observe from the last four columns that as they focus more on quality, the production cost increases because the production should be set up more frequently; the transportation cost increases since the products should be delivered to retailers more frequently to ensure quality; the inventory cost decreases since less inventory can be held because of the high quality requirement. Note that the inventory cost increases from 1516 RMB to 2614 RMB when the quality improves from 0.58 to 0.18. This is because the initial inventory can no longer be all used to fulfill the demand when the quality requirement is high, and some of the initial inventory is held till the end of its shelf life. It is also shown in the last column that when the quality level improves from 0.88 to 0.18, the production setup number doubles.

### 5.4.2 Randomly generated instances

To thoroughly evaluate the proposed model and solution method, we randomly generate 37 sets of instances with 5 for each, totaling 185 instances. These instances are divided into two groups characterized by the number of retailers, vehicles and periods. The first group of instances, small-sized instances, are generated as follows: the number of retailers $n$ is 10, 15, or 20; the length of the planning horizon $|T|$ is 3, 4, or 5 for $n$= 10, 15, or 20 with 1 vehicle; and it is 3, 6, 9, or 12 for $n$= 20 with 2 vehicles. $|Q|$ is set to All parameters concerning inventory routing are generated according to the rules of [39]. The initial inventory $I_i^{0q}$ for all $i \in N$ and $q \in Q$ is set to 0. Remaining parameters concerning production are generated based on [24],

as stated below: the production capacity $C$ is set to be $\frac{\sum_{i\in R}\sum_{t\in T} D_i^t}{|T|}\beta$, where $\beta$ is randomly generated from interval [2, 4]; the inventory capacity of the plant $U_0$ is set to be $\gamma C$, where $\gamma$ is randomly generated from interval [1.5, 2]; the unit production cost $a_t$ is randomly generated from interval [4, 7]; the fixed setup cost $b_t$ is set to be $\mu C$, where $\mu$ is randomly generated from interval [0.3, 0.5].

In TIH, FPDP($\epsilon_m, \boldsymbol{\delta^m}, \boldsymbol{\lambda^m}$) is solved by CPLEX with default settings and a time limit of 10 seconds. VRP($t, \epsilon_m$) is solved by VRPH package [53],[54]. Parameters $N^T$ and $N^D$ are set to 20 and 5, respectively. The value of $\theta$ is randomly generated from [0.5, 1.5] for each $i \in R, t \in T$. The aggregate vehicle capacity parameters $\lambda_t^m$ and $\tau$ are set to 1 and 0.05, respectively. According to our preliminary test, the step size parameter $\Delta$ is set to 0.05.

To evaluate the performance of $\epsilon$-CTIH, we compare the approximate Pareto front $\mathcal{A}_F$ obtained by $\epsilon$-CTIH with a reference set $\mathcal{R}_F$ obtained by another $\epsilon$-constraint method ($\epsilon$-CC) that solves the BFPRP by applying the $\epsilon$-constraint method framework and adopts CPLEX to solve each of the mono-objective problem FPRP($\epsilon_m$) instead of TIH. The general framework of $\epsilon$-CC is similar to that in Algorithm 2. However, CPLEX is called to solve FPRP($\epsilon_m$) with a time limit of 14400s, i.e., 4 hours and only $\epsilon$ is used to link FPRP($\epsilon_m$) and FPRP($\epsilon_{m+1}$). Since the number of subtour elimination constraints (5.15) increases exponentially when the number of nodes increases, they are treated as lazy constraints and are added only when they are violated. In detail, we first solve the full model FPRP($\epsilon_m$) with (5.15) being relaxed. At the end of each iteration, if there exist one or more subtours in the solution, then we add the subtour elimination constraints concerning these subtours. Then the model is re-solved. This process is repeated until no subtour can be found. Then we get the optimal solution. If the time limit is reached and there are still subtours in the solution, then CPLEX fails to provide any feasible solution. To compare the performance of $\epsilon$-CTIH and $\epsilon$-CC, we use the four widely used performance indicators for bi-objective optimization methods [80], namely the cardinality $|\mathcal{A}_F|$ and $|\mathcal{R}_F|$, hypervolume ratio $\mathcal{H}$, average e-dominance $\mathcal{D}$, and computation time $\mathcal{T}$. Note that $\mathcal{A}_F$ and $\mathcal{R}_F$ give the Pareto solution sets obtained by the evaluated method (i.e., $\epsilon$-CTIH) and the reference method (i.e., $\epsilon$-CC), respectively.

#### 5.4.2.1 Results for small-sized instances

Firstly, we present the computational results on 65 (13 sets) small-sized instances in Table 5.4, where the first four columns denote the numbers of instance sets, retailers, vehicles and periods, respectively. Cardinality $|\mathcal{R}_F|$ and $|\mathcal{A}_F|$ are shown in columns 5 and 6. Indicators $\mathcal{H}$ and $\mathcal{D}$ are represented in columns 7 and 8, respectively. Columns

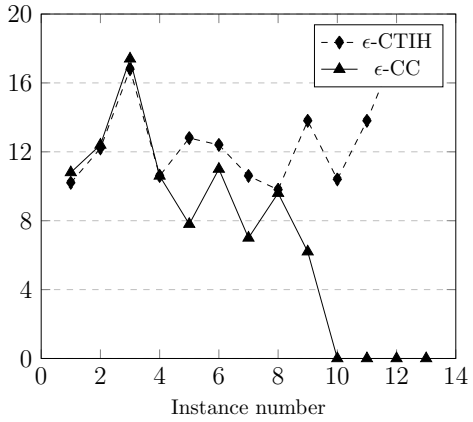
$\mathcal{T}_{\mathcal{R}}$ and $\mathcal{T}_{\mathcal{A}}$ correspond to the average computation times (seconds) of $\epsilon$-CC and $\epsilon$-CTIH, respectively. Each value in column 5 to 10 is an average value over 5 instances with the same size. Finally, the last two columns show the longest computation time for each group of 5 instances by $\epsilon$-CC and $\epsilon$-CTIH, respectively. The results are further shown in Fig. 5.2 that gives the comparisons of the four indicators, i.e., the Cardinality, Hypervolume ration, e-dominance and computation time.

Table 5.4: Comparison Results for small-sized Instances

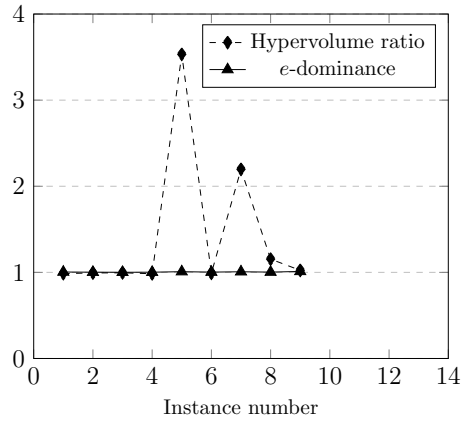
| NO. | $n$ | $\|K\|$ | $\|T\|$ | $\|\mathcal{R}_F\|$ | $\|\mathcal{A}_F\|$ | $\mathcal{H}$ | $\mathcal{D}$ | $\mathcal{T}_{\mathcal{R}}$(s) | $\mathcal{T}_{\mathcal{A}}$(s) | $\#\mathcal{T}_{\mathcal{R}}$(s) | $\#\mathcal{T}_{\mathcal{A}}$(s) |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 10 | 1 | 3 | 10.8 | 10.2 | 0.986 | 1.004 | 260 | 25 | 667 | 28 |
| 2 | | | 4 | 12.4 | 12.2 | 0.993 | 1.002 | 880 | 38 | 2899 | 57 |
| 3 | | | 5 | 17.4 | 16.8 | 0.995 | 1.001 | 2325 | 72 | 7265 | 103 |
| 4 | 15 | 1 | 3 | 10.6 | 10.6 | 0.984 | 1.002 | 1954 | 31 | 4958 | 39 |
| 5 | | | 4 | 7.8 | 12.8 | 3.532 | 1.008 | 10081 | 48 | 14400 | 54 |
| 6 | | | 5 | 11 | 12.4 | 0.990 | 1.003 | 6179 | 65 | 14400 | 101 |
| 7 | 20 | 1 | 3 | 7 | 10.6 | 2.197 | 1.007 | 2963 | 31 | 14400 | 37 |
| 8 | | | 4 | 9.6 | 9.8 | 1.154 | 1.003 | 5779 | 48 | 14400 | 50 |
| 9 | | | 5 | 6.2 | 13.8 | 1.024 | 1.011 | 9647 | 65 | 14400 | 95 |
| 10 | 20 | 2 | 3 | 0 | 10.4 | - | - | 14400 | 36 | 14400 | 60 |
| 11 | | | 6 | 0 | 13.8 | - | - | 14400 | 164 | 14400 | 309 |
| 12 | | | 9 | 0 | 18 | - | - | 14400 | 553 | 14400 | 809 |
| 13 | | | 12 | 0 | 16.4 | - | - | 14400 | 780 | 14400 | 918 |
| Average | | | | 7.1 | 12.9 | 1.428 | 1.005 | 4452 | 150 | 9754 | 204 |

We can observe from cardinality columns (columns 4 and 5) of Table 5.4 and Fig. 5.2(a) that $\epsilon$-CTIH can generate almost as many Pareto solutions as $\epsilon$-CC does for instances with 10 retailers and 1 vehicle, which indicates the performances of $\epsilon$-CTIH are comparable with $\epsilon$-CC. Algorithm $\epsilon$-CTIH performs better than $\epsilon$-CC by providing more Pareto solutions for other instances (instances with 15 retailers or more). It is worth noting that $\epsilon$-CC cannot even obtain a Pareto feasible solution within 14400s for instances with 20 retailers and 2 vehicles. In terms of hypervolume ratio, Fig. 5.2(b) indicates that all values are either close to or greater than 1, which means that $\epsilon$-CTIH performs equally or better than $\epsilon$-CC. Moving to the e-dominance indicator, all values of the e-dominance are very close to 1, which implies that the obtained Pareto solutions by the two methods are quite close to each other. As for the computation time, Table 5.4 shows that the average computation times for $\epsilon$-CTIH and $\epsilon$-CC are 150s and 7513 s, respectively. The former needs only 2% computation time of the latter. The same trend can also be observed from the last two columns which show the longest computation time of the two methods. The last results indicate that $\epsilon$-CTIH is relatively stable since the average of the longest computation time over all instances is only 54s longer than that of the average computation time. It shows, however, a great difference between the average computation time and the longest computation

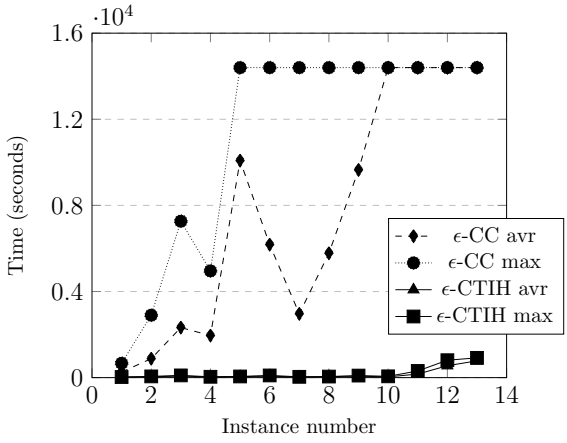
time of $\epsilon$-CC. In addition, the computation time of $\epsilon$-CC increases exponentially while that of $\epsilon$-CTIH increases relatively slowly, as shown in Fig. 5.2(c).




(a) comparison results of cardinality




(b) Hypervolume ration and e-dominance




(c) comparison results of computation times

Fig. 5.2: Comparison results between $\epsilon$-CTIH and $\epsilon$-CC

#### 5.4.2.2 Results for medium- and large-sized instances

To further evaluate the performance of $\epsilon$-CTIH, we next conduct experiments on 120 instances that are considered as medium- and large-sized instances of the considered problem. For these instances, $\epsilon$-CC cannot provide a feasible solution within 14400s. Therefore, we only report computational results obtained by $\epsilon$-CTIH and evaluate its performance in terms of cardinality and computation time. Detailed results are given in Table 5.5.

As can be seen from Table 5.5, $\epsilon$-CTIH is able to solve all instances and obtain a reasonable number of approximate Pareto solutions within acceptable computation time. The average number of Pareto solutions obtained is 13.8 and the average and longest computation times are 1119s and 1660s, respectively. It is shown that the computation time of $\epsilon$-CTIH increases quickly with the number of periods $|T|$,

Table 5.5: Computational Results for large-sized Instances

| $n$ | $\|K\|$ | $\|T\|$ | $\|\mathcal{A}_F\|$ | $\mathcal{T}_{\mathcal{A}}$(s) | $\#\mathcal{T}_{\mathcal{A}}$(s) | $n$ | $\|K\|$ | $\|T\|$ | $\|\mathcal{A}_F\|$ | $\mathcal{T}_{\mathcal{A}}$(s) | $\#\mathcal{T}_{\mathcal{A}}$(s) |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 30 | 3 | 3 | 8.8 | 33 | 45 | 60 | 6 | 3 | 10.8 | 70 | 132 |
| | | 6 | 14.4 | 247 | 421 | | | 6 | 11.6 | 446 | 552 |
| | | 9 | 16.4 | 741 | 1274 | | | 9 | 13.4 | 1353 | 2215 |
| | | 12 | 16.2 | 1203 | 1442 | | | 12 | 21.2 | 3441 | 5060 |
| 40 | 4 | 3 | 11.6 | 50 | 94 | 80 | 8 | 3 | 9.6 | 81 | 160 |
| | | 6 | 15.8 | 356 | 624 | | | 6 | 10.8 | 558 | 659 |
| | | 9 | 17.4 | 1230 | 1519 | | | 9 | 19.8 | 2228 | 3890 |
| | | 12 | 16 | 1424 | 2038 | | | 12 | 16.4 | 3125 | 4078 |
| 50 | 5 | 3 | 7.2 | 42 | 62 | 100 | 10 | 3 | 7.8 | 82 | 206 |
| | | 6 | 12.8 | 384 | 419 | | | 6 | 11 | 626 | 933 |
| | | 9 | 14.8 | 1238 | 1508 | | | 9 | 13.8 | 2288 | 4050 |
| | | 12 | 15.8 | 1896 | 2590 | | | 12 | 18.6 | 3726 | 5881 |
| Average | | | | | | | | | 13.8 | 1119 | 1660 |

and it increases slightly with the number of retailers $n$. Take instances with 100 retailers as an example. As the number of periods increases from 3 to 12, the average computation time increases from 82s to 3726s. If we fix the number of periods as 3, the average computation time increases from 33s to 82s as the number of retailers increases from 30 to 100. In terms of the longest computation time, some of the instances show a relatively large difference from the average, but the computation time is still acceptable. The longest computation time is 1660s on average over the 12 sets of instances. The computation time varies even for instances with the same size because the number of Pareto solutions obtained is different, which may result in solving different number of mono-objective problems.

## 5.5 Conclusion

This chapter investigates a new bi-objective food production routing problem in which two objectives are simultaneously optimized, i.e., to minimize the total production, inventory and routing cost, and to maximize the average quality of food products received by final customers. Firstly, a novel bi-objective MILP model is proposed for the problem. Then a heuristic that combines an $\epsilon$-constraint framework and a two-phase iterative heuristic is developed to generate near-optimal Pareto solution set. And a fuzzy logic decision method is applied to help decision makers select a preferred solution. Computational results on a case study indicate that the proposed model and solution method are able to solve a real world case. In particular, the proposed approach can reduce 10.77% total cost compared with an three-phase constructive heuristic. The developed approach also shows that a decision maker can improve the average quality provided to final customers with a slight cost increase. Computational

results on randomly generated instances show that the proposed approach can provide better Pareto solution set with shorter computation time compared with CPLEX.

The corresponding work has been published in the following paper.

Y. Li, F. Chu, C. Feng, C. Chu, and M. Zhou. Integrated Production Inventory Routing Planning for Intelligent Food Logistics Systems. *IEEE Transactions on Intelligent Transportation Systems*, pages 1-12, (99)2018. DOI:10.1109/TITS.2018.2835145.

# Chapter 6

# Food Production Routing Problem with Time Windows

## 6.1 Introduction

In supply chains, a good delivery plan should often respect a time interval (time window) imposed by retailers. Time windows are one of the most common realistic constraints in a distribution network [6]. In food supply chains, delivery time windows are more relevant to perishable food products distribution since their quality deteriorates quickly.

In this chapter, we study a food production routing problem with time windows (FPRPTW) that extends the classic PRP by considering food perishability and service time windows. Firstly, a MILP is formulated for the considered problem. Then several valid inequalities are proposed to strengthen the formulation. Finally, randomly generated instances are used to validate the proposed model with a MILP solver.

The remainder of this chapter is organized as follows. Section 6.2 presents the description and formulation of the considered problem. In Section 6.3, a series of valid inequalities are proposed to strengthen the model. Computational experiments on randomly generated instances are conducted in Section 6.4. Finally, Section 6.5 summarizes this chapter.

## 6.2 Problem description and formulation

The studied FPRPTW can be stated as follows.

Consider a complete digraph $G$=$\{N$,$A\}$ with a set of nodes $N$=$\{0, 1, \ldots, n\}$ and a set of arcs $A$=$\{(i, j) : i, j \in N, i \neq j\}$. A plant with limited production and storage capacities is located at vertex 0. It has a fleet of homogeneous vehicles $K$=$\{1, , 2, \ldots, |K|\}$, each having capacity $V$. A set of $n$ retailers $R$=$\{1, 2, \ldots, n\}$ with

a limited storage capacity are geographically located at nodes $\{1, \ldots, n\}$. Consider a time horizon $T$=$\{1, 2, \ldots, |T|\}$, the customer demand at each retailer is assumed to be deterministic and time varying. We assume that the newly produced food can be immediately used to replenish the retailer, and the delivered food can be directly used by the retailer to fulfill the demand of its customers. Moreover, in each time period, each retailer has a specified visit time window, which remains the same for all time periods. Note that food quality can be distinguished by a set of quality levels $Q$=$\{0, 1, \ldots, |Q|\}$, where 0 represents the freshest food. The objective is to maximize the total profit that is equal to the total selling revenue minus the production, inventory and routing costs. The decisions to be made for each period are: 1) how much to produce at the plant; 2) how much to replenish each retailer; 3) how to arrange the transportation routes for the planned deliveries so that the time window constraints are respected; and 4) how to fulfill customer demand at each retailer.

The study is conducted under the following assumptions: 1) each vehicle's route starts and ends at the plant, and each vehicle can perform at most one trip within each period; 2) each retailer can be visited at most once within each period; i.e., split delivery is not allowed; 3) food quality degrades by one quality unit per period; and 4) once the quality goes beyond $|Q|$, food products can no longer be used to meet customer demand and should be directly turned into waste. To formulate the problem, the following parameters and variables are defined:

Parameters

$c_{ij}$: travel cost on arc $(i, j) \in A$;
$\tau_{ij}$: travel time on arc $(i, j) \in A$;
$a_t$: unit production cost in period $t \in T$;
$b_t$: production setup cost in period $t \in T$;
$C$: production capacity;
$I_i^{q0}$: initial inventory at $i \in N$ with $q \in Q$;
$D_i^t$: customer demand at retailer $i \in R$ in period $t \in T$;
$U_i$: inventory capacity of $i \in N$;
$h_i^q$: unit inventory holding cost per period at $i \in N$ with $q \in Q$;
$g_i^q$: selling price for a unit of a product with quality $q \in Q$ at retailer $i \in R$;
$e_i$: starting time of time window of retailer $i \in R$;
$e_0$: earliest departure time from the plant;
$l_i$: finishing time of time window of retailer $i \in R$;
$l_0$: latest arrival time at the plant;
$s_i$: service time at retailer $i \in R$;
$V$: vehicle capacity;
$M_{ij}$: a big number equal to $l_i + s_i + \tau_{ij}$;

$Y_t$: a big number equal to $Y_t = \min(C, \sum_{i \in R} \sum_{t'=t}^{|T|} D_{it'})$;
$W_{it}$: a big number equal to $W_{it} = \min(U_i + D_{it}, \sum_{t'=t}^{|T|} D_{it})$.

Variables

$\xi_t$: production quantity in period $t$;
$w_t$: binary variable equal to 1 if $\xi_t > 0$; otherwise 0;
$I_i^{qt}$: inventory level with quality $q$ of retailer $i$ at the end of period $t$;
$y_i^{qkt}$: delivery quantity with quality $q$ to retailer $i$ by vehicle $k$ in period $t$;
$\vartheta_i^{qt}$: quantity of food used to fulfill customer demand with quality $q$ at retailer $i$ in period $t$;
$z_i^{kt}$: time to serve retailer $i$ by vehicle $k$ in period $t$;
$v_i^{kt}$: binary variable equal to 1 if retailer $i$ is visited by vehicle $k$ in period $t$; otherwise 0;
$\varepsilon^{kt}$: departure time of vehicle $k$ in period $t$ at the plant;
$\lambda^{kt}$: arrival time of vehicle $k$ in period $t$ at the plant;
$x_{ij}^{kt}$: binary variable equal to 1 if arc $(i, j)$ is traversed by vehicle $k$ in period $t$; otherwise 0.

With the above description and notation, BFPRP can be formulated as follows:

$$\max \sum_{i \in R} \sum_{q \in Q} \sum_{t \in T} g_i^q \vartheta_i^{qt} - \sum_{t \in T} (a_t \xi_t + b_t w_t) - \sum_{i \in N} \sum_{q \in Q} \sum_{t \in T} h_i^q I_i^{qt} - \sum_{(i,j) \in A} \sum_{k \in K} \sum_{t \in T} c_{ij} x_{ij}^{kt} \tag{6.1}$$

s.t.

$$I_0^{qt} = I_0^{q-1,t-1} - \sum_{i \in R} \sum_{k \in K} y_i^{qkt}, \forall q \in Q \setminus \{0\}, t \in T \tag{6.2}$$

$$I_0^{0t} = \xi_t - \sum_{i \in R} \sum_{k \in K} y_i^{0kt}, \forall t \in T \tag{6.3}$$

$$I_i^{qt} = I_i^{q-1,t-1} + \sum_{k \in K} y_i^{qkt} - \vartheta_i^{qt}, \forall i \in R, q \in Q \setminus \{0\}, t \in T \tag{6.4}$$

$$I_i^{0t} = \sum_{k \in K} y_i^{0kt} - \vartheta_i^{0t}, \forall i \in R, t \in T \tag{6.5}$$

$$\xi_t \leq Y_t w_t, \forall t \in T \tag{6.6}$$

$$\sum_{q \in Q} I_i^{qt} \leq U_i, \forall i \in N, t \in T \tag{6.7}$$

$$\sum_{q \in Q} \vartheta_i^{qt} = D_i^t, \forall i \in R, t \in T \tag{6.8}$$

$$\sum_{i \in R} \sum_{q \in Q} y_i^{qkt} \leq V, \forall k \in K, t \in T \tag{6.9}$$

$$\sum_{q \in Q} y_i^{qkt} \leq W_{it} v_i^{kt}, \forall i \in R, k \in K, t \in T \tag{6.10}$$

$$\sum_{k\in K} v_i^{kt} \leq 1, \forall i \in R, t \in T \tag{6.11}$$

$$\sum_{j\in N\setminus\{i\}} x_{ij}^{kt} = \sum_{j\in N\setminus\{i\}} x_{ji}^{kt} = v_i^{kt}, \forall i \in R, k \in K, t \in T \tag{6.12}$$

$$\sum_{i\in R} x_{0i}^{kt} \leq 1, \forall k \in K, t \in T \tag{6.13}$$

$$z_i^{kt} + s_i + \tau_{ij} - M_{ij}(1 - x_{ij}^{kt}) \leq z_j^{kt}, \forall i \in R, j \in R \setminus \{j\}, k \in K, t \in T \tag{6.14}$$

$$\varepsilon^{kt} + \tau_{0i} - M_{0i}(1 - x_{0i}^{kt}) \leq z_i^{kt}, \forall i \in R, k \in K, t \in T \tag{6.15}$$

$$z_i^{kt} + \tau_{i0} - M_{i0}(1 - x_{i0}^{kt}) \leq \lambda^{kt}, \forall i \in R, k \in K, t \in T \tag{6.16}$$

$$e_i \leq z_i^{kt} \leq l_i, \forall i \in R, k \in K, t \in T \tag{6.17}$$

$$e_0 \leq \varepsilon^{kt}, \forall k \in K, t \in T \tag{6.18}$$

$$\lambda^{kt} \leq l_0, \forall k \in K, t \in T \tag{6.19}$$

$$\xi_t \geq 0, \forall t \in T \tag{6.20}$$

$$w_t \in \{0, 1\}, \forall t \in T \tag{6.21}$$

$$I_i^{qt} \geq 0, \forall i \in N, q \in Q, t \in T \tag{6.22}$$

$$\vartheta_i^{qt} \geq 0, \forall i \in R, q \in Q, t \in T \tag{6.23}$$

$$y_i^{qkt} \geq 0, \forall i \in R, q \in Q, k \in K, t \in T \tag{6.24}$$

$$v_i^{kt} \in \{0, 1\}, \forall i \in R, k \in K, t \in T \tag{6.25}$$

$$x_{ij}^{kt} \in \{0, 1\}, \forall (i, j) \in A, k \in K, t \in T \tag{6.26}$$

$$z_i^{kt} \geq 0, \forall i \in R, k \in K, t \in T \tag{6.27}$$

$$\varepsilon^{kt} \geq 0, \forall k \in K, t \in T \tag{6.28}$$

$$\lambda^{kt} \geq 0, \forall k \in K, t \in T \tag{6.29}$$

Objective function (6.1) maximizes the total profit, which is equal to the total selling revenue minus the production, inventory, and routing costs. Constraints (6.2)-(6.5) indicate the food flow balance at the plant and retailers. Constraints (6.6) restrict the production quantity with the production capacity and the total remaining demand. In particular, the production quantity should be 0 if the plant is not set up for production. Constraints (6.7) are the inventory capacity constraints. Constraints (6.8) indicate that customer demand at each retailer must be satisfied. The vehicle capacity is imposed by constraints (6.9). Constraints (6.10) mean that if a retailer is visited, the delivery quantity must not exceed its remaining demand; otherwise, the delivery quantity should be 0. Constraints (6.11) forbid split delivery. Constraints (6.12) correspond to the vehicle flow conservation. Constraints (6.13) denote that a vehicle can perform at most one route in each period. Constraints (6.14)-(6.19) guarantee the schedule feasibility of time windows. Precisely, constraints (6.14) indicate

the time consistency among retailers. Constraints (6.15) and (6.16) correspond to the time consistency of a vehicle departure from and arrival to the plant, respectively. Constraints (6.17)-(6.19) ensure the time window constraints of each retailer to be respected, and limit the departure and arrival times at the plant. Constraints (6.20)-(6.29) are nonnegative and integer conditions on decision variables. We denote the objective (6.1) and constraints (6.2)-(6.29) as the original model, namely model $\mathcal{P}_1$. This FPRPTW is NP-hard, since it contains a classical vehicle routing problem with time windows (VRPTW) which has been well known as a NP-hard problem [91].

## 6.3 Valid inequalities

In this section, we present valid inequalities that strengthen the model $\mathcal{P}_1$. The following valid inequalities are developed based on that introduced by [14] and [38].

$$\sum_{j\in R} x_{0j}^{kt} \geq \sum_{j\in R} x_{0j}^{k+1,t},\ \forall k \in K \setminus \{|K|\}, t \in T \tag{6.30}$$

$$v_i^{kt} \leq \sum_{j=1}^{i} v_j^{k-1,t},\ \forall i \in R, k \in K \setminus \{1\}, t \in T \tag{6.31}$$

$$\sum_{q\in Q} I_i^{qt} \geq (1 - \sum_{k\in K}\sum_{\tau=t+1}^{\tau=t'} v_i^{k\tau}) \sum_{\tau=t+1}^{\tau=t'} D_{i\tau},\ \forall\, i \in R, t \in T, t < t' \leq |T| \tag{6.32}$$

Constraints (6.30) and (6.31) break the symmetry in selecting the identical vehicles and assign these vehicles to routes. Particularly, constraints (6.30) make sure that the vehicles with smaller number are always used first. Constraints (6.31) mean that if the $k$th vehicle is used to serve retailer $i$, then $(k-1)$th vehicle must have been used to serve retailers with smaller sequence number. Constraints (6.32) are used to strengthen the inventory routing part. These constraints refer that if a retailer is not visited during the time interval $[t+1, \tau]$, then the inventory held in period $t$ must be sufficient to meet the summation of the total demand in this interval.

Similarly, we derive the following constraints to strengthen the formulation by considering the food perishability. These constraints denote that there must be at least one visit to each retailer within the $|Q|+1$ time periods. This must happen since the food can not be stored longer than $|Q|+1$ periods, e.g. if one retailer is visited in time period $t$, then the next visit is no later than time period $t+|Q|+1$. Constraints (6.33) can be very strong when the food has a very short shelf-life.

$$\sum_{k\in K}\sum_{\tau=t}^{t+|Q|} v_i^{kt} \geq 1,\ \forall i \in R, 1 \leq t \leq |T| - |Q| \tag{6.33}$$

The objective function (6.1) and the constraints (6.2)-(6.33) form the strengthened model, namely model $\mathcal{P}_2$.

## 6.4 Computational experiments

In this section, in order to verify the effectiveness of the proposed models, computational experiments on 45 randomly generated instances are carried out. The instances are solved by CPLEX version 12.6.0. All runs are performed on a personal computer with CORE CPU 2.5 GHz and 8 GB RAM.

Table 6.1: Parameters for FPRPTW model

| Parameters | Generation description |
|---|---|
| $c_{ij}$ | $= \sqrt{(X_i - X_j)^2 + (Y_i - Y_j)^2}$ |
| $\tau_{ij}$ | $c_{ij}/10$ |
| $D_{it}$ | randomly generated from U[30, 210] |
| $C$ | $= \beta(\sum_{i\in R}\sum_{t\in T} D_{it})/\lvert T\rvert$, where $\beta$ is randomly generated from U[2, 4] |
| $U_0$ | $= \beta C$,where $\beta$ is randomly generated from U[1.5, 2] |
| $U_i$ | $= \beta_i \max_{t\in T}\{D_{it}\}$, where $\beta_i$ is randomly generated from U[2, 3], $i \in R$ |
| $h_i^q$ | $= (\beta_1 + q\beta_2)/(1 + q)/100$, where $\beta_1$ and $\beta_2$ are randomly generated from U[0, 100] and [0, 70], respectively |
| $a_t$ | randomly generated from U[4, 7] |
| $b_t$ | $= \beta C$, and $\beta$ is randomly generated from U[0.3, 0.5] |
| $g_i^q$ | $= \beta_1 - (\beta_1 - \beta_2)q/\lvert Q\rvert$, where $\beta_1$ and $\beta_2$ are randomly generated from U[10, 20] and [4, 7], respectively |
| $V$ | $= 1.25(\sum_{i\in R}\sum_{t\in T} D_{it})/\lvert K\rvert/\lvert T\rvert$ |
| $s_i$ | U[60, 120] |
| $e_i$ | $\max(\beta_1 - \beta_2/2, e_0 + \tau_{0i})$, where $\beta_1$ is generated from U$[e_0 + \tau_{0i}, l_0 - \tau_{i0} - s_i]$, and $\beta_2$ is set to $\beta_3(l_0 - e_0)$, in which $\beta_3$ is randomly generated from U[0.125, 0.25] |
| $l_i$ | $\min(\beta_1 + \beta_2/2, l_0 - \tau_{i0} - s_i)$, where $\beta_1$ is generated from U$[e_0 + \tau_{0i}, l_0 - \tau_{i0} - s_i]$, and $\beta_2$ is set to $\beta_3(l_0 - e_0)$, in which $\beta_3$ is randomly generated from U[0.125, 0.25] |

### 6.4.1 Instance generation

To validate the proposed models, we randomly generate 9 sets of instances with 5 for each, totaling 45 instances. These instances are generated as follows: the number of retailers $n$ is set to $\{10, 20, 30, 40\}$; the length of the planning horizon $|T|$ is $\{3, 6, 10\}$, $\{3, 6\}$, and $\{3\}$ for instances with up to 20, 30, and 40 retailers, respectively. The number of vehicles $|K|$ is set to 1 for instances with 10 retailers, and 2 for instances with 20, 30, and 40 retailers. The number of quality levels $|Q|$ is set as: 2 for $|T|$= 3; 3

for $|T|= 6$; and 5 for $|T|= 10$. The coordinate of a node $i$ in graph $G$ that corresponds to a plant or a retailer location $(X_i, Y_i)$ is randomly generated from U[0,1000]. At the beginning of the planning horizon, the initial inventory $I_i^{0q}$ for all $i \in N$ and $q \in Q$ is set to 0. Concerning the parameters for the time windows, we use the principle proposed by [91]. The earliest departure time $e_0$ from the depot and the latest arrival time $l_0$ to the depot are set to be 0 and 1200 respectively. The service time in retailer is randomly generated from interval [60, 120]. The detailed generation of parameters for MFPRP is given in Table 6.1.

### 6.4.2 Computational results

In this section, the computational results are reported in Table 6.2, in which the first four columns denote the number of retailers, periods, vehicles, and quality levels, respectively. Note that these results are average values of five instances with the same size. The best solution value obtained is denoted by "LB". The "UB1" and "UB2" denote the best upper bound found with the original $\mathcal{P}_1$ and strengthened $\mathcal{P}_2$ respectively within the time limit 7200 seconds. "Gap" denotes the average gap between the obtained lower bound and the upper bound by $\mathcal{P}_1$, i.e. (UB1-LB)/LB×100%. The second column indicates the CPU time. The last column "Opt" reports the number of instances optimally solved by CPLEX. The results in Table 6.2 show that CPLEX is capable to optimally solve small instances with 10 retailers up to 10 periods, and 20 retailers up to 3 time periods. The average computational time is less than 102.97s. In addition, 2 out of 5 instances with 20 retailers and 6 periods are solved to optimality, but with longer computational time. The computational time increases dramatically with the number of retailers and the length of planning horizon. Looking at the instance with 30 retailers and 3 periods, the average gap reaches 22.27%, which shows the complexity of the proposed problem. Note that the average computational time is less than the time limit, because most of the tests, if not solved optimally, run out of memory before reaching the time limit. Concerning the computational results with all valid inequalities, results in column UB2 obviously show that the all upper bounds have decreased on average comparing with that in column UB1. This indicates that the proposed valid inequalities can help CPLEX generate better upper bounds.

## 6.5 Conclusion

This chapter investigates a food production routing problem with time windows. The problem is firstly formulated as a MILP. Then valid inequalities are introduced to strengthen the formulation. Experimental results on 45 randomly generated instances

Table 6.2: Comparison Results for Small-size Instances

| $n$ | $\|T\|$ | $\|K\|$ | $\|Q\|$ | LB | UB1 | UB2 | Gap(%) | Tim(s) | Opt |
|---|---|---|---|---|---|---|---|---|---|
| 10 | 3 | 1 | 2 | 13449 | 13449 | 13449 | 0 | 0.28 | 5 |
| 20 | 3 | 2 | 2 | 33656 | 33656 | 33656 | 0 | 102.97 | 5 |
| 30 | 3 | 2 | 2 | 39606 | 48079 | 47793 | 22.27 | 1714.58 | 0 |
| 40 | 3 | 2 | 2 | 63964 | 73016 | 72927 | 14.87 | 4931.71 | 0 |
| 10 | 6 | 1 | 3 | 34279 | 34279 | 34279 | 0 | 3.44 | 5 |
| 20 | 6 | 2 | 3 | 76397 | 77770 | 77699 | 1.85 | 1642.84 | 2 |
| 30 | 6 | 2 | 3 | 111336 | 122412 | 122211 | 10.03 | 4667.89 | 0 |
| 10 | 10 | 1 | 5 | 71921 | 71921 | 71921 | 0 | 32.38 | 5 |
| 20 | 10 | 2 | 5 | 151998 | 154367 | 154040 | 1.6 | 4742.49 | 0 |

show that the proposed model is capable of providing integrated plans for the decision makers. As the problem is quite complex, only small instances can be solved optimally by CPLEX. This is still an ongoing work, efficient algorithms should be developed to solve large-sized instances for the problem.

The corresponding work has been published in the following paper.
Y. Li, F. Chu, C. Chu, W. Zhou, and Z. Zhu. Integrated production inventory routing planning with time windows for perishable food. In *19th IEEE International conference on Intelligent Transportation Systems*, Rio de Janeiro, Brazil, November 1-4, 2016 , pages 2651-2656, 2016.

# Chapter 7

# Conclusions and perspectives

This thesis investigates a class of integrated supply chain planning problems, i.e., the PRPs, which simultaneously optimize the production, inventory, and routing decisions. Firstly, we study a multi-product PRP with outsourcing (MPRPOS), and develop an efficient three-level heuristic that can solve both the MPRPOS and the classic PRP. Considering the specific characteristics of food supply chain, we then investigate a multi-plant FPRP integrating packaging decision (MFPRP), and develop a hybrid matheuristic to solve it. Thirdly, to investigate the potential tradeoff between cost and quality, a bi-objective FPRP with quality consideration (BFPRP) is studied and an $\epsilon$-constraint-based two-phase iterative heuristic and a fuzzy logic decision method are developed. Finally, a FPRP with delivery time windows (FPRPTW) is addressed.

In Chapter 2, we first present a systematic review of recent developments on the PRP. Next we review the integrated planning problems considering food quality and perishability. Then we report the existing solution algorithms for solving the PRP. We find that although the PRP has been receiving increasing attention, the MPRP has not been sufficiently studied yet. In addition, the FPRP has received little attention despite its importance.

Chapter 3 investigates the MPRPOS that is a generalization of the classic PRP. Firstly, a MILP is proposed for the problem. Then a three-level heuristic (TLH) is developed to solve both MPRPOS and the classic PRP. In TLH, the original MPRPOS is decomposed into two subproblems that are solved iteratively to generate an initial solution (probably infeasible); then a restricted production direct-distribution problem is solved to repair the infeasible solution and a route consolidation procedure that solves many TSPs further improves the incumbent solution; finally, a fix-and-optimize procedure is used to improve the incumbent solution iteratively. To evaluate the performance of TLH, 225 newly generated MPRPOS instances with up to 200 customers, 20 vehicles, 6 periods, and 12 products are first tested, followed by the tests on 1530 widely used PRP benchmark instances with up to 200 customers, 13

vehicles, and 20 periods. Computational results on MPRPOS instances show that TLH can find feasible solutions with average gaps (times) of 1.99% (98.6s) and 9.90% (106.8s) for instances with up to 14 customers and 200 customers, respectively. In particular, TLH finds 283 new best solutions for the tested benchmark instances.

In Chapter 4, we address the MFPRP. Firstly, a MILP model and a hybrid matheuristic (HM) are developed for it. The heuristic combines a two-phase iterative method, a fix-and-optimize procedure, and a route-based optimization. In particular, the route-based optimization exploits the useful route information provided by the two-phase iterative and the fix-and-optimize components. The performance of HM is evaluated on 320 randomly generated instances with up to 4 plants, 50 retailers, 6 periods and 3 packages. Numerical results for instances with up to 4 plants and 15 retailers show that HM can find solutions with an average gap of -1.26% and 5.09% compared to the lower and upper bounds provided by CPLEX and needs only less than 5% of the computation time compared with CPLEX. Results for instances with up to 4 plants and 50 retailers indicate that HM can still propose solution with an average gap of 7.55% and average computation time of 460.7s. Analysis on discount policies show that the total profit is significantly impacted by different discount policies. As the discount becomes quicker, the total profit decreases. Performance analysis of HM shows that its good performance is a result of the hybridization of the three components.

Chapter 5 investigates the BFPRP in which two objectives are simultaneously optimized, i.e., to minimize the total cost, and to maximize the average quality of food products received by customers. Especially, the quality of food received by customers, which significantly impacts the customer satisfaction, is for the first time treated as an objective. Firstly, a novel bi-objective MILP model is proposed for the problem. Then an $\epsilon$-constraint two-phase iterative heuristic ($\epsilon$-CTIH) and a fuzzy logic decision method are developed to generate a near-optimal Pareto solution set and to help decision makers select a preferred solution. $\epsilon$-CTIH transforms the original bi-objective problem into a series of single-objective problems that are solved iteratively. Computational results on a case study indicate that the proposed model and solution method are able to solve a real world case. The proposed fuzzy logic decision method can help decision makers select a preferred solution based on their preference. We observe from the results for the case study that a decision maker can significantly improve the average quality provided to customers with a slight cost increase. Computational results on 185 randomly generated instances indicate that $\epsilon$-CTIH is efficient in providing an approximate Pareto solution set.

In Chapter 6, we study the FPRPTW where the delivery time window is considered. As it is still an ongoing work, we present a MILP and some preliminary

results. The proposed model is directly solved by CPLEX. Experimental results on 45 randomly generated instances show that the problem is quite complex, and only small-sized instances can be solved optimally by CPLEX. The results motivate us to develop new algorithms to solve this complex problem.

The PRP is an optimization problem integrating the production, inventory and routing decisions. There are many future research directions to be investigated. They are summarized as follows:

1) Developing efficient solution algorithms for the PRP. Combining matheuristics with metaheuristics is promising. Matheuristics show some strengths in providing good near optimal solution, and metaheuristics have the potential to further improve the solution by exploiting part of the solution space. For the studied FPRPTW in Chapter 6, our preliminary results show that only small-sized instances can be solved by a commercial solver within the time limit, and the solution quality is far from optimal. In future work, efficient algorithms that combine the strengths of matheuristics and metaheuristics could be developed to solve it.

2) Integrating more realistic issues in the PRP. In the production planning aspect, the production setup times and changeovers among several production lines can be included. In terms of the routing aspect, rich routing problems, e.g., split delivery that a customer can be visited several times in a period, open routes that a vehicle does not have to return to the depot, and transshipments among customers can be considered to model some real-world situations. In particular, the PRP with multiple trips, which is common in modern e-commerce logistics and daily food distributions, could be studied.

3) Considering environmental aspect. In order to cope with global climate change, the amount of greenhouse gas emission is adopted as a key performance indicator by many companies. The PRP involves the production, inventory and routing activities during which green house gas is emitted. Therefore, emission can be considered in a PRP in the lot-sizing aspect as [99] or in the vehicle routing aspect as [43]. In addition, as the amount of daily food produced and distributed is huge, the recycle packages and the re-distribution of food are also worth studying. These activities will help reduce green house gas emission and food waste.

4) Dealing with demand uncertainty. In reality, customer demand is frequently uncertain and only part of the demand information is known at the beginning of the planning horizon. In this case, a deterministic model is no longer suitable and the demand uncertainty should be explicitly considered. Correspondingly, efficient algorithms should also be developed to propose robust solutions.

5) Quality dependent demand. In reality, the quality of a perishable food product affects the customer demand. Customers may always prefer the freshest product, and

any product whose quality does not meet their expectations will turn into lost sales. Thus explicitly modeling the relationship between the food quality and demand is promising in a FPRP. Accordingly, efficient solution methods for these complex problems should be developed to provide good-quality solutions to the decision makers.

6) Temperature control throughout the integrated planning. The production, storage and transportation of food products should respect certain conditions and regulations. In particular, food products are often stored and transported with a recommended range of temperature. Thus in a long food supply chain, it is meaningful to adopt different transportation modes with different capacity, temperature, and speed. A proper planning of such multi-modal distribution will reduce food waste and improve food quality and safety.

# Bibliography


[1] Nabil Absi, Claudia Archetti, Stéphane Dauzere-Péres, and Dominique Feillet. A two-phase iterative heuristic approach for the production routing problem. *Transportation Science*, 49(4):784–795, 2014.

[2] Nabil Absi, Claudia Archetti, Stéphane Dauzère-Pérès, Dominique Feillet, and M Grazia Speranza. Comparing sequential and integrated approaches for the production routing problem. *European Journal of Operational Research*, 269(2):633–646, 2018.

[3] Yossiri Adulyasak, Jean-François Cordeau, and Raf Jans. Formulations and branch-and-cut algorithms for multivehicle production and inventory routing problems. *INFORMS Journal on Computing*, 26(1):103–120, 2014.

[4] Yossiri Adulyasak, Jean-François Cordeau, and Raf Jans. Optimization-based adaptive large neighborhood search for the production routing problem. *Transportation Science*, 48(1):20–45, 2014.

[5] Yossiri Adulyasak, Jean-François Cordeau, and Raf Jans. Benders decomposition for production routing under demand uncertainty. *Operations Research*, 63(4):851–867, 2015.

[6] Yossiri Adulyasak, Jean-François Cordeau, and Raf Jans. The production routing problem: A review of formulations and solution algorithms. *Computers & Operations Research*, 55:141–152, 2015.

[7] Agostinho Agra, Cristina Requejo, and Filipe Rodrigues. An adjustable sample average approximation algorithm for the stochastic production-inventory-routing problem. *Networks*, 72(1):5–24, 2018.

[8] Agostinho Agra, Cristina Requejo, and Filipe Rodrigues. A hybrid heuristic for a stochastic production-inventory-routing problem. *Electronic Notes in Discrete Mathematics*, 64:345–354, 2018.

[9] Ravindra K Ahuja, Wei Huang, H Edwin Romeijn, and Dolores Romero Morales. A heuristic approach to the multi-period single-sourcing problem with production and inventory capacities and perishability constraints. *INFORMS Journal on Computing*, 19(1):14–26, 2007.

[10] Pedro Amorim and Bernardo Almada-Lobo. The impact of food perishability issues in the vehicle routing problem. *Computers & Industrial Engineering*, 67:223–233, 2014.

[11] Pedro Amorim, MAF Belo-Filho, FMB Toledo, C Almeder, and Bernardo Almada-Lobo. Lot sizing versus batching in the production and distribution planning of perishable goods. *International Journal of Production Economics*, 146(1):208–218, 2013.

[12] Pedro Amorim, H-O Günther, and Bernardo Almada-Lobo. Multi-objective integrated production and distribution planning of perishable products. *International Journal of Production Economic*, 138(1):89–101, 2012.

[13] Enrico Angelelli, Renata Mansini, and M Grazia Speranza. Kernel search: A general heuristic for the multi-dimensional knapsack problem. *Computers & Operations Research*, 37(11):2017–2026, 2010.

[14] Claudia Archetti, Luca Bertazzi, Gilbert Laporte, and Maria Grazia Speranza. A branch-and-cut algorithm for a vendor-managed inventory-routing problem. *Transportation Science*, 41(3):382–391, 2007.

[15] Claudia Archetti, Luca Bertazzi, Giuseppe Paletta, and M Grazia Speranza. Analysis of the maximum level policy in a production-distribution system. *Computers & Operations Research*, 38(12):1731–1746, 2011.

[16] Claudia Archetti, Natashia Boland, and M Grazia Speranza. A matheuristic for the multivehicle inventory routing problem. *INFORMS Journal on Computing*, 29(3):377–387, 2017.

[17] Vinícius Amaral Armentano, André Luís Shiguemoto, and Arne Løkketangen. Tabu search with path relinking for an integrated production–distribution problem. *Computers & Operations Research*, 38(8):1199–1209, 2011.

[18] Jonathan F Bard and Narameth Nananukul. Heuristics for a multiperiod inventory routing problem with production decisions. *Computers & Industrial Engineering*, 57(3):713–723, 2009.

[19] Jonathan F Bard and Narameth Nananukul. The integrated production–inventory–distribution–routing problem. *Journal of Scheduling*, 12(3):257–280, 2009.

[20] Jonathan F Bard and Narameth Nananukul. A branch-and-price algorithm for an integrated production and inventory routing problem. *Computers & Operations Research*, 37(12):2202–2217, 2010.

[21] MAF Belo-Filho, Pedro Amorim, and Bernardo Almada-Lobo. An adaptive large neighbourhood search for the operational integrated production and distribution problem of perishable products. *International Journal of Production Research*, 53(20):6040–6058, 2015.

[22] Luca Bertazzi, Demetrio Laganà, Leandro Callegari Coelho, and Annarita De Maio. *The Multi-depot Inventory Routing Problem: An Application of Vendor-management Inventory in City Logistic*. CIRRELT, Interuniversity Research Centre on Enterprise Networks, Logistics and Transportation, 2017.

[23] Jean-François Bérubé, Michel Gendreau, and Jean-Yves Potvin. An exact $\varepsilon$–constraint method for bi–objective combinatorial optimization problems: Application to the traveling salesman problem with profits. *European Journal of Operational Research*, 194(1):39–50, 2009.

[24] Mourad Boudia, Mohamed Aly Ould Louly, and Christian Prins. Combined optimization of production and distribution. In *CD-ROM proceedings of the international conference on industrial engineering and systems management, IESM*, volume 5, 2005.

[25] Mourad Boudia, Mohamed Aly Ould Louly, and Christian Prins. A reactive grasp and path relinking for a combined production–distribution problem. *Computers & Operations Research*, 34(11):3402–3419, 2007.

[26] Mourad Boudia, Mohamed Aly Ould Louly, and Christian Prins. Fast heuristics for a combined production planning and vehicle routing problem. *Production Planning and Control*, 19(2):85–96, 2008.

[27] Mourad Boudia and Christian Prins. A memetic algorithm with dynamic population management for an integrated production–distribution problem. *European Journal of Operational Research*, 195(3):703–715, 2009.

[28] Nadjib Brahimi and Tarik Aouam. Multi-item production routing problem with backordering: a milp approach. *International Journal of Production Research*, 54(4):1076–1093, 2016.

[29] Gerald Brown, Joseph Keegan, Brian Vigus, and Kevin Wood. The kellogg company optimizes production, inventory, and distribution. *Interfaces*, 31(6):1–15, 2001.

[30] Ann Melissa Campbell and Martin WP Savelsbergh. A decomposition approach for the inventory-routing problem. *Transportation Science*, 38(4):488–502, 2004.

[31] Pankaj Chandra. A dynamic distribution model with warehouse and customer replenishment requirements. *Journal of the Operational Research Society*, 44(7):681–692, 1993.

[32] Pankaj Chandra and Marshall L Fisher. Coordination of production and distribution planning. *European Journal of Operational Research*, 72(3):503–517, 1994.

[33] Huey-Kuo Chen, Che-Fu Hsueh, and Mei-Shiang Chang. Production scheduling and vehicle routing with time windows for perishable food products. *Computers & Operations Research*, 36(7):2311–2319, 2009.

[34] Masoud Chitsaz, Jean-François Cordeau, and Raf Jans. A unified decomposition matheuristic for assembly, production and inventory routing. *Cahiers du GERAD G-2017-03*, 2017.

[35] Chengbin Chu, Feng Chu, Jinhong Zhong, and Shanlin Yang. A polynomial algorithm for a lot-sizing problem with backlogging, outsourcing and limited inventory. *Computers & Industrial Engineering*, 64(1):200–210, 2013.

[36] Feng Chu and Chengbin Chu. Polynomial algorithms for single-item lot-sizing models with bounded inventory and backlogging or outsourcing. *IEEE transactions on Automation Science and Engineering*, 4(2):233–251, 2007.

[37] Feng Chu, Chengbin Chu, and Xiao Liu. Lot sizing models with backlog or out-sourcing. In *2004 IEEE International Conference onSystems, Man and Cybernetics*, volume 5, pages 4342–4347. IEEE, 2004.

[38] Leandro C Coelho and Gilbert Laporte. Improved solutions for inventory-routing problems through valid inequalities and input ordering. *International Journal of Production Economics*, 155:391–397, 2014.

[39] Leandro C Coelho and Gilbert Laporte. Optimal joint replenishment, delivery and inventory management policies for perishable products. *Computers & Operations Research*, 47:42–52, 2014.

[40] Carlos A Coello Coello, Gary B Lamont, David A Van Veldhuizen, et al. *Evolutionary algorithms for solving multi-objective problems*, volume 5. Springer, 2007.

[41] Yves Crama, Mahmood Rezaei, Martin Savelsbergh, and Tom Van Woensel. Stochastic inventory routing for perishable products. *Transportation Science*, 52(3):526–546, 2018.

[42] Kalyanmoy Deb. Multi–objective optimization. In *Search methodologies*, pages 403–449. Springer, 2014.

[43] Emrah Demir, Tolga Bektaş, and Gilbert Laporte. The bi–objective pollution–routing problem. *European Journal of Operational Research*, 232(3):464–478, 2014.

[44] Priyantha Devapriya, William Ferrell, and Neil Geismar. Integrated production and distribution scheduling with a perishable product. *European Journal of Operational Research*, 259(3):906–916, 2017.

[45] Chung-Yuan Dye and Tsu-Pang Hsieh. An optimal replenishment policy for deteriorating items with effective investment in preservation technology. *European Journal of Operational Research*, 218(1):106–112, 2012.

[46] Masoud Esmaili, Heidar Ali Shayanfar, and Nima Amjady. Multi–objective congestion management incorporating voltage and transient stabilities. *Energy*, 34(9):1401–1412, 2009.

[47] Awi Federgruen and Michal Tzur. Time-partitioning heuristics: Application to one warehouse, multiitem, multiretailer lot-sizing problems. *Naval Research Logistics (NRL)*, 46(5):463–486, 1999.

[48] Marshall L Fisher. The lagrangian relaxation method for solving integer programming problems. *Management science*, 27(1):1–18, 1981.

[49] Francesca Fumero and Carlo Vercellis. Synchronized development of production, inventory, and distribution schedules. *Transportation Science*, 33(3):330–340, 1999.

[50] H Neil Geismar, Gilbert Laporte, Lei Lei, and Chelliah Sriskandarajah. The integrated production and transportation scheduling problem for a product with a short lifespan. *INFORMS Journal on Computing*, 20(1):21–33, 2008.

[51] Michel Gendreau, Alain Hertz, and Gilbert Laporte. A tabu search heuristic for the vehicle routing problem. *Management Science*, 40(10):1276–1290, 1994.

[52] K Matthew Gilley and Abdul Rasheed. Making more by doing less: an analysis of outsourcing and its effects on firm performance. *Journal of management*, 26(4):763–790, 2000.

[53] Chris Groër. `https://www.coin-or.org/download/source/VRPH/`, 2010.

[54] Chris Groër, Bruce Golden, and Edward Wasil. A library of local search heuristics for the vehicle routing problem. *Mathematical Programming Computation*, 2(2):79–101, 2010.

[55] Gianfranco Guastaroba and Maria Grazia Speranza. A heuristic for bilp problems: the single source capacitated facility location problem. *European Journal of Operational Research*, 238(2):438–450, 2014.

[56] Mohammed Haoues, Mohammed Dahane, and Nadia Kenza Mouss. Outsourcing optimization in two-echelon supply chain network under integrated production-maintenance constraints. *Journal of Intelligent Manufacturing*, pages 1–25, 2016.

[57] PH Hsu, HM Wee, and HM Teng. Preservation technology investment for deteriorating inventory. *International Journal of Production Economics*, 124(2):388–394, 2010.

[58] IBM. Ibm ilog cplex optimization studio 12.7 information center. `https://www.ibm.com/support/knowledgecenter/SSSA5P_12.7.1/ilog.odms.studio.help/pdf/usrcplex.pdf`, 2017.

[59] Qi Kang, ShuWei Feng, MengChu Zhou, Ahmed Chiheb Ammari, and Khaled Sedraoui. Optimal load scheduling of plug-in hybrid electric vehicles via weight-aggregation multi-objective evolutionary algorithms. *IEEE Transactions on Intelligent Transportation Systems*, 18(9):2557–2568, 2017.

[60] Philippe Lacomme, Aziz Moukrim, Alain Quilliot, and Marina Vinot. Supply chain optimisation with both production and transportation integration: multiple vehicles for a single perishable product. *International Journal of Production Research*, pages 1–24, 2018.

[61] Tung Le, Ali Diabat, Jean-Philippe Richard, and Yuehwern Yih. A column generation-based heuristic algorithm for an inventory routing problem with perishable goods. *Optimization Letters*, 7(7):1481–1502, 2013.

[62] Shine-Der Lee and Shu-Chuan Lan. Production lot sizing with a secondary outsourcing facility. *International Journal of Production Economic*, 141(1):414–424, 2013.

[63] Young Hae Lee, Chan Seok Jeong, and Chiung Moon. Advanced planning and scheduling with outsourcing in manufacturing supply chain. *Computers & Industrial Engineering*, 43(1-2):351–374, 2002.

[64] Lei Lei, Shuguang Liu, Andrzej Ruszczynski, and Sunju Park. On the integrated production, inventory, and distribution routing problem. *IIE Transactions*, 38(11):955–970, 2006.

[65] Feiyue Li, Bruce Golden, and Edward Wasil. Very large-scale vehicle routing: new test problems, algorithms, and results. *Computers & Operations Research*, 32(5):1165–1179, 2005.

[66] Guiping Li, Xiuli He, Jing Zhou, and Hao Wu. Pricing, replenishment and preservation technology investment decisions for non-instantaneous deteriorating items. *Omega*, 2018.

[67] Qing Li, Peiwen Yu, and Xiaoli Wu. Shelf life extending packaging, inventory control and grocery retailing. *Production and Operations Management*, 2017.

[68] Yantong Li and Feng Chu. Bi-objective optimization of an integrated production inventory routing planning for perishable food. 2017.

[69] Yantong Li, Feng Chu, and Kejia Chen. Coordinated production inventory routing planning for perishable food. *IFAC-PapersOnLine*, 50(1):4246–4251, 2017.

[70] Yantong Li, Feng Chu, Chengbin Chu, Wei Zhou, and Zhanguo Zhu. Integrated production inventory routing planning with time windows for perishable food. In *2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC)*, pages 2651–2656. IEEE, 2016.

[71] Yantong Li, Feng Chu, Chengbin Chu, and Zhanguo Zhu. An efficient three-level heuristic for the large-scaled multi-product production routing problem with outsourcing. *European Journal of Operational Research*, 272(3):914–927, 2019.

[72] Yantong Li, Feng Chu, Chenpeng Feng, Chengbin Chu, and MengChu Zhou. Integrated production inventory routing planning for intelligent food logistics

systems. *IEEE Transactions on Intelligent Transportation Systems*, (99):1–12, 2018.

[73] Yantong Li, Feng Chu, Yang Zhen, and Roberto Calvo Wolfler. A production inventory routing planning for perishable food with quality consideration. *IFAC-PapersOnLine*, 49(3):407–412, 2016.

[74] Shen Lin and Brian W Kernighan. An effective heuristic algorithm for the traveling-salesman problem. *Operations Research*, 21(2):498–516, 1973.

[75] Guowei Liu, Jianxiong Zhang, and Wansheng Tang. Joint dynamic pricing and investment strategy for perishable foods with price-quality dependent demand. *Annals of Operations Research*, 226(1):397–416, 2015.

[76] George Mavrotas. Effective implementation of the $\varepsilon$–constraint method in multi–objective mathematical programming problems. *Applied Mathematics and Computation*, 213(2):455–465, 2009.

[77] Pedro L Miranda, Jean-François Cordeau, Deisemera Ferreira, Raf Jans, and Reinaldo Morabito. A decomposition heuristic for a rich production routing problem. *Computers & Operations Research*, 2018.

[78] Pedro L Miranda, Reinaldo Morabito, and Deisemara Ferreira. Optimization model for a production, inventory, distribution and routing problem in small furniture companies. *TOP*, 26(1):30–67, 2018.

[79] Samira Mirzaei and Abbas Seifi. Considering lost sale in inventory routing problems for perishable goods. *Computers & Industrial Engineering*, 87:213–227, 2015.

[80] Tatsuya Okabe, Yaochu Jin, and Bernhard Sendhoff. A critical survey of performance indices for multi–objective optimisation. In *Evolutionary Computation, 2003. CEC'03. The 2003 Congress on*, volume 2, pages 878–885. IEEE, 2003.

[81] Yuzhuo Qiu, Ming Ni, Liang Wang, Qinqin Li, Xuanjing Fang, and Panos M Pardalos. Production routing problems with reverse logistics and remanufacturing. *Transportation Research Part E: Logistics and Transportation Review*, 111:87–100, 2018.

[82] Yuzhuo Qiu, Jun Qiao, and Panos M Pardalos. A branch-and-price algorithm for production routing problems with carbon cap-and-trade. *Omega*, 68:49–61, 2017.

[83] Yuzhuo Qiu, Jun Qiao, and Panos M Pardalos. Optimal production, replenishment, delivery, routing and inventory management policies for products with perishable inventory. *Omega*, 2018.

[84] Yuzhuo Qiu, Liang Wang, Xiaoling Xu, Xuanjing Fang, and Panos M Pardalos. Formulations and branch-and-cut algorithms for multi-product multi-vehicle production routing problems with startup cost. *Expert Systems with Applications*, 2018.

[85] Yuzhuo Qiu, Liang Wang, Xiaoling Xu, Xuanjing Fang, and Panos M Pardalos. A variable neighborhood search heuristic algorithm for production routing problems. *Applied Soft Computing*, 66:311–318, 2018.

[86] Aiying Rong, Renzo Akkerman, and Martin Grunow. An optimization approach for managing fresh food quality throughout the supply chain. *International Journal of Production Economic*, 131(1):421–429, 2011.

[87] Mirko Ruokokoski, OGUZ Solyali, Jean-François Cordeau, Raf Jans, and Haldun Süral. Efficient formulations and a branch-and-cut algorithm for a production-routing problem. *GERAD Technical Report G-2010-66*, 2010.

[88] Robert A Russell. Mathematical programming heuristics for the production routing problem. *International Journal of Production Economic*, 193:40–49, 2017.

[89] Florian Sahling, Lisbeth Buschkühl, Horst Tempelmeier, and Stefan Helber. Solving a multi-level capacitated lot sizing problem with multi-period setup carry-over via a fix-and-optimize heuristic. *Computers & Operations Research*, 36(9):2546–2553, 2009.

[90] Homayoun Shaabani and Isa Nakhai Kamalabadi. An efficient population-based simulated annealing algorithm for the multi-product multi-retailer perishable inventory routing problem. *Computers & Industrial Engineering*, 99:189–201, 2016.

[91] Marius M Solomon. Algorithms for the vehicle routing and scheduling problems with time window constraints. *Operations research*, 35(2):254–265, 1987.

[92] Oğuz Solyalı and Haldun Süral. A multi-phase heuristic for the production routing problem. *Computers & Operations Research*, 2017.

[93] Mehmet Soysal, Jacqueline M Bloemhof-Ruwaard, Rene Haijema, and Jack GAJ van der Vorst. Modeling an inventory routing problem for perishable products with environmental considerations and demand uncertainty. *International Journal of Production Economic*, 164:118–133, 2015.

[94] Douglas J Thomas and Paul M Griffin. Coordinated supply chain management. *European Journal of Operational Research*, 94(1):1–15, 1996.

[95] Guangdong Tian, Yaping Ren, and MengChu Zhou. Dual-objective scheduling of rescue vehicles to distinguish forest fires via differential evolution and particle swarm optimization combined algorithm. *IEEE Transactions on Intelligent Transportation Systems*, 17(11):3009–3021, 2016.

[96] Guangdong Tian, MengChu Zhou, Peigen Li, Chaoyong Zhang, and Hongfei Jia. Multiobjective optimization models for locating vehicle inspection stations subject to stochastic demand, varying velocity and regional constraints. *IEEE Transactions on Intelligent Transportation Systems*, 17(7):1978–1987, 2016.

[97] Trung Hieu Tran, Gábor Nagy, Thu Ba T Nguyen, and Niaz A Wassan. An efficient heuristic algorithm for the alternative-fuel station location problem. *European Journal of Operational Research*, 269(1):159–170, 2018.

[98] Peng Wu, Ada Che, Feng Chu, and Yunfei Fang. Exact and heuristic algorithms for rapid and station arrival-time guaranteed bus transportation via lane reservation. *IEEE Transactions on Intelligent Transportation Systems*, 18(8):2028–2043, 2017.

[99] Tao Wu, Fan Xiao, Canrong Zhang, Yan He, and Zhe Liang. The green capacitated multi-item lot sizing problem with parallel machines. *Computers & Operations Research*, 2018.

[100] Chih-Te Yang, Chung-Yuan Dye, and Ji-Feng Ding. Optimal dynamic trade credit and preservation technology allocation for a deteriorating inventory model. *Computers & Industrial Engineering*, 87:356–369, 2015.

[101] Jianxiong Zhang, Zhenyu Bai, and Wansheng Tang. Optimal pricing policy for deteriorating items with preservation technology investment. *Journal of Industrial & Management Optimization*, 10(4):1261–1277, 2014.

[102] Qi Zhang, Arul Sundaramoorthy, Ignacio E Grossmann, and Jose M Pinto. Multiscale production routing in multicommodity supply chains with complex production facilities. *Computers & Operations Research*, 79:207–222, 2017.

[103] Zhen Zhou, Feng Chu, Ada Che, and MengChu Zhou. $\varepsilon$-constraint and fuzzy logic-based optimization of hazardous material transportation via lane reservation. *IEEE Transactions on Intelligent Transportation Systems*, 14(2):847–857, 2013.

université PARIS-SACLAY
ÉCOLE DOCTORALE
Sciences et technologies de l'information et de la communication (STIC)

**Titre :** Modèles et algorithmes pour une classe de problèmes combinés de production et de tournées de véhicules

**Mots clés :** Production et tournées de véhicules ; Agroalimentaire ; Conditionnement ; Programmation linéaire ; Matheuristique ; Multi-critère.

**Résumé :** Le problème combiné de production et de tournées de véhicules (PRP) consiste à proposer une planification intégrée de la production et de la distribution. Il vise à optimiser le coût global de la chaîne logistique et à améliorer le niveau de service aux clients. Bien que le PRP et son application en agroalimentaire (FPRP) présentent des enjeux importants à la fois scientifiques et industriels, ils n'ont pas été suffisamment étudiés dans la littérature. L'objectif de cette thèse est de développer de nouveaux modèles et algorithmes pour le PRP et le FPRP.
Dans cette thèse, nous avons d'abord étudié un PRP multi-produit avec sous-traitance (MPRPOS) qui est une extension naturelle du PRP classique. Pour ce problème, un nouveau programme linéaire en nombres mixte (MILP) a été proposé et une heuristique à trois niveaux a été développée. Les expériences numériques sur 225 instances générées aléatoirement pour le MPRPOS et 1530 instances de benchmark du PRP montrent de très bonnes performances de l'heuristique proposée. En particulier, nous avons obtenu de nouvelles meilleures solutions pour 283 instances de benchmark.
A partir de l'étude du MPRPOS et en prenant en compte les spécificités des produits agroalimentaires (périssabilité et qualité), trois nouveaux FPRPs ont été ensuite étudié : 1) un FPRP multi-site avec conditionnement (MFPRP) ; 2) un FPRP multi-critère (BFPRP) : minimisation du coût total de la chaîne logistique et maximisation de la qualité ; et 3) un FPRP avec des contraintes de fenêtres horaires (FPRPTW). Pour chacun des problèmes, un modèle MILP a été établi. En outre, une Matheuristique hybride combinant une méthode itérative, une procédure de fixe-et-optimisation et un processus d'optimisation basé sur les routes déterminées pendant les deux premières étapes a été développée pour le MFPRP. Pour le BFPRP, une heuristique du type $\epsilon$-contrainte et une méthode par logique floue sont proposées. Et le FPRPTW est directement résolu par le solveur CPLEX. Une étude des cas montre que le modèle et l'algorithme proposés pour le BFPRP peuvent significativement améliorer la performance de l'entreprise. Les résultats numériques sur des instances générées aléatoirement montrent que les méthodes développées sont plus performantes que le CPLEX.

**Title :** Models and Algorithms for a Class of Production Routing Problems

**Keywords :** Production routing problem ; Food ; Packaging ; Linear programming ; Matheuristics ; multi-objective.

**Abstract :** The production routing problem (PRP) consists of determining an integrated production and distribution planning that aims to optimize overall cost and improve service level. Although the PRP has been attracting academic and practical interests, it has not been well studied in the literature. Food production routing problem (FPRP) that is more complex than the classic PRP due to food perishability, has rarely been studied. This thesis focuses on developing new models and algorithms for the PRP and FPRP.
Firstly, a multi-product PRP with outsourcing (MPRPOS) that is a generalization of the classic PRP is addressed. For the problem, a mixed integer linear programming (MILP) model is proposed and a three-level heuristic is designed. Computational experiments on 225 newly generated MPRPOS instances and 1530 PRP benchmark instances demonstrate the effectiveness and efficiency of the proposed heuristic. Especially, 283 new best solutions for PRP benchmark instances are found by the heuristic.
Considering food quality and perishability, and based on the study for the PRP, three new FPRPs are then investigated, i.e., 1) a multi-plant FPRP with packaging consideration (MFPRP) ; 2) a bi-objective FPRP (BFPRP) that minimizes the total supply chain cost and maximizes food quality simultaneously ; and 3) a FPRP with delivery time window constraints (FPRPTW). For each of the studied problems, a MILP model is proposed. Moreover, a hybrid matheuristic that combines a two-phase iterative method, a fix-and-optimize procedure, and a route-based optimization is developed for the MFPRP. For the BFPRP, an $\epsilon$-constraint-based heuristic and a fuzzy logic decision method are proposed to generate near-optimal Pareto solutions and to help decision makers select a preferred solution. And the FPRPTW is directly solved by the state-of-the-art solver CPLEX. A case study shows the proposed model and algorithm for BFPRP can improve food supply chain performance. Computational results on randomly generated instances demonstrate the proposed hybrid matheuristic and $\epsilon$-constraint-based heuristic outperform CPLEX.

**Université Paris-Saclay**
Espace Technologique / Immeuble Discovery
Route de l'Orme aux Merisiers RD 128 / 91190 Saint-Aubin, France