HEC MONTRÉAL
École affiliée à l'Université de Montréal

Quantifying the Impacts of Innovative Business Models in Last-Mile Delivery

par
Xin Wang

Thèse présentée en vue de l'obtention du grade de Ph. D. en administration
(spécialisation Sciences de la décision)

Décembre 2024

HEC MONTRÉAL

École affiliée à l'Université de Montréal

Cette thèse intitulée :

Quantifying the Impacts of Innovative Business Models in Last-Mile Delivery

Présentée par :

Xin Wang

a été évaluée par un jury composé des personnes suivantes :

Maryam Daryalal
HEC Montréal
Présidente-rapportrice

Erick Delage
HEC Montréal
Directeur de recherche

Okan Arslan
HEC Montréal
Codirecteur de recherche

Sanjay Dominik Jena
Université du Québec à Montréal
Membre du jury

Michael Hewitt
Loyola University Chicago
Examinateur externe

Raf Jans
HEC Montréal
Représentant du directeur de HEC Montréal

Résumé

La livraison du dernier kilomètre, l'étape finale critique dans la chaîne d'approvisionnement du commerce électronique, joue un rôle essentiel pour assurer la satisfaction des clients et le succès commercial dans le marché du commerce électronique, évalué à plusieurs milliers de milliards de dollars. Il est essentiel d'assurer des livraisons rapides, ponctuelles et efficaces avec une haute qualité de service, une grande fiabilité, une haute efficacité opérationnelle et des coûts réduits.

Pour atteindre cet objectif, nous étudions trois modèles de livraison innovants, notamment la livraison par partage d'espace (crowdkeeping), la livraison ultra-rapide et la livraison multi-magasins, tout en quantifiant leurs impacts sur les livraisons du dernier kilomètre. Les services de livraison par partage d'espace exploitent des espaces inutilisés pour le stockage temporaire et le transfert flexible de colis afin de réduire les coûts de livraison, éliminer les livraisons échouées et bénéficier à l'ensemble des participants du système. Les services de livraison ultra-rapide se concentrent sur des livraisons rapides et fiables malgré l'incertitudes dans les temps de trajet et dans la demande, en équilibrant la rapidité des livraisons et la rentabilité opérationnelle. Les services de livraison multi-magasins permettent aux clients de consolider leurs commandes provenant de plusieurs magasins et aux chauffeurs de transférer des articles aux nœuds de transbordement, tout en intègrant des stratégies d'attente pour répondre aux commandes arrivant de manière dynamique, facilitant ainsi des livraisons rapides et de courts temps de trajet.

Du point de vue de la modélisation, nous développons des programmes mathématiques pour relever ces défis et les reformulons en versions plus simples afin d'obtenir des

solutions de haute qualité efficacement. Pour le problème de la livraison par partage d'espace, nous modélisons un programme bi-niveaux, le reformulons en programmes équivalents à un seul niveau et dérivons des approximations précises pour obtenir des solutions de qualité élevée et efficaces. Pour le problème de livraison ultra-rapide, nous développons des programmes avec contraintes probabilistes robustes pour gérer les incertitudes, les reformulons comme des programmes linéaires semi-infinis équivalents et proposons leurs approximations internes et externes avec des contraintes linéaires finies, qui peuvent être résolues de manière optimale avec efficacité. Pour le problème de la livraison multi-magasins, nous formulons un programme linéaire en nombres entiers avec de nombreuses contraintes et variables liées à la planification des trajets et aux considérations temporelles, puis adoptons une approche d'optimisation par apprentissage qui intègre l'apprentissage automatique et l'optimisation pour atteindre efficacement des solutions de haute qualité.

Nous réalisons des expériences numériques avec des ensembles de données réelles afin de fournir des informations managériales pour améliorer l'efficacité, la rentabilité, la fiabilité et la satisfaction des clients dans la logistique du dernier kilomètre. Les services de livraison par partage d'espace génèrent des profits élevés en consolidant les livraisons et en éliminant les livraisons échouées. Cela s'explique par le fait que les "crowd keepers" ont une flexibilité accrue, une meilleure disponibilité et des coûts plus faibles que les options de stockage fixes telles que les casiers automatisés, conduisant à un système de livraison du dernier kilomètre plus efficace et plus rentable. Pour assurer des services de livraison ultra-rapides avec une grande fiabilité et rentabilité, un niveau de service quotidien, qui donne la priorité aux périodes avec une fréquence de commandes plus élevée et utilise une protection par couches multiples, s'avère efficace malgré les incertitudes liées aux arrivées de commandes ou aux conditions de trafic. Les services de livraison multi-magasins utilisant la consolidation, le transbordement et une stratégie d'attente de courte durée, permettent de réduire les temps de complétion des commandes et les temps de trajet des conducteurs grâce à une meilleure consolidation spatio-temporelle.

Mots-clés

Livraison du dernier kilomètre; Conception de réseau; Niveau de service; Tarification et routage; Ramassage et livraison; Consolidation et transbordement; Programmes bi-niveau; Programmes contraints par des enveloppes probabilistes; Optimisation robuste; Optimiser par apprentissage

Méthodes de recherche

Recherche quantitative; Programmation mathématique; Optimisation en incertitude; Apprentissage automatique

Abstract

Last-mile delivery, the critical final step in the e-commerce supply chain, plays a vital role in ensuring customer satisfaction and business success in the multi-trillion-dollar e-commerce market. It is essential to provide on-time, on-demand deliveries efficiently with high service quality, high reliability, high operational efficiency, and low costs.

To achieve this goal, we investigate three innovative delivery models, including crowdkeeping delivery, ultra-fast delivery, and multi-store order delivery, as well as quantify their impacts in last-mile deliveries. The crowdkeeping delivery services leverage the availability of the crowd for temporary parcel storage and flexible parcel transfers to reduce delivery costs, decrease delivery failures, and benefit all system participants. The ultra-fast delivery services focus on rapid and reliable deliveries under uncertainties in travel times and demand arrivals, balancing delivery speed and operational profitability. The multi-store order delivery services enable customers to consolidate orders from multiple stores, allow drivers to transfer items at transshipment nodes, and incorporate waiting strategies to serve dynamically arriving orders, facilitating fast deliveries and short travel times.

From the modeling perspective, we develop mathematical programs to tackle these challenges and reformulate them into more tractable formulations to obtain high-quality solutions efficiently. For the crowdkeeping delivery problem, we model a bilevel program to consider the preferences of different participants, reformulate it into equivalent single-level programs for exact solutions, and derive tight approximations for high-quality and efficient solutions. For the ultra-fast delivery problem, we develop robust probabilistic

envelope constrained programs to handle uncertainties, reformulate them as semi-infinite linear programs equivalently, and propose their inner and outer approximations with finite linear constraints, which can be solved to optimality efficiently. For the multi-store order delivery problem, we formulate a mixed-integer linear program and adopt a learning-to-optimize approach that integrates machine learning and optimization to achieve high-quality solutions efficiently.

We conduct numerical experiments using real-world datasets to derive managerial insights for improving efficiency, profitability, reliability, and customer satisfaction in last-mile logistics. The crowdkeeping delivery services generate high profits by consolidating deliveries and reducing failed deliveries. This is because crowd keepers have extra flexibility, more availability, and lower costs compared to fixed storage options such as automated lockers, leading to a more efficient and profitable last-mile delivery system. To provide ultra-fast delivery services with high reliability and profitability, a daily service level that prioritizes time periods with higher order frequencies and employs multi-layer partial protection proves effective despite uncertain order arrivals or traffic conditions. The multi-store order delivery services with consolidation, transshipment, and a short-duration waiting strategy yield short order completion times and total driver travel times through superior spatial and temporal consolidation.

Keywords

Last-mile delivery; Network design; Service level; Pricing and routing; Pickup and delivery; Consolidation and transshipment; Bilevel programs; Probabilistic envelope constrained programs; Robust optimization; Learning-to-optimize

Research Methods

Quantitative research; Mathematical programming; Optimization under uncertainty; Machine learning

Contents

List of Tables

List of Figures

List of Acronyms

ABBR Abréviation

BAA Baccalauréat en administration des affaires

DESS Diplôme d'études supérieures spécialisées

HEC Hautes études commerciales

MBA Maîtrise en administration des affaires

MSc Maîtrise

PhD Doctorat

Acknowledgements

Deep gratitude to my supervisors, my co-authors, my family, my friends, and everyone who has supported and guided me throughout this long journey of exploration. A special thanks to myself for persevering and overcoming every challenge along the way.

Preface

This thesis consists of three articles listed as follows:

1. Xin Wang, Okan Arslan, Erick Delage. (2024). Crowdkeeping in Last-mile Delivery. *Transportation Science*, 58(2), 474-498.

2. Xin Wang, Okan Arslan, Jean-François Cordeau, Erick Delage. Optimizing Ultra-fast Delivery Networks and Service Guarantees under Uncertainty. Under review.

3. Xin Wang, Okan Arslan, Jean-François Cordeau, Erick Delage. Learning-to-optimize for Consolidation and Transshipment in Multi-store Order Delivery. Under review.

General Introduction

E-commerce has experienced unprecedented growth during and after the COVID-19 pandemic, encompassing a wide range of sectors such as essentials, furniture, fashion, and food. Online retailers such as Amazon and Ikea, along with food delivery services such as UberEats and DoorDash, have become indispensable in daily life. By 2024, the e-commerce market reached a revenue of USD 4.12 trillion, with projections suggesting hat it will grow to USD 6.48 trillion by 2029, driven by an annual growth rate of 9.49% (Statista 2024). This rapid expansion has significantly reshaped consumer expectations, fueling a demand for fast, reliable, and cost-efficient last-mile deliveries. Last-mile delivery, the final step in the e-commerce supply chain where products are transported from depots to customers, is critical for ensuring customer satisfaction and driving business success. It accounts for a substantial portion of delivery costs, with estimates suggesting it can comprise up to 41% of the total logistics cost (Jacobs et al. 2019).

From a research perspective, Savelsbergh and Van Woensel (2016) review and discuss the challenges and opportunities in last-mile logistics, emphasizing its critical role in modern supply chains. In the realm of on-time and on-demand last-mile delivery, much of the research focuses on improving delivery times to enhance service quality or reducing travel costs to meet demand and improve efficiency. This is achieved through empirical studies demonstrating the importance of on-demand delivery (Mao et al. 2022; Li and Wang 2024), optimizing driver-to-customer assignments or driver routing for efficient dispatch (Liu, He, and Max Shen 2021; Carlsson et al. 2024), introducing innovative business models (Cao and Qi 2023; Raghavan and Zhang 2024) to improve the overall

system efficiency, and applying advanced learning techniques (Hildebrandt and Ulmer 2022; Auad, Erera, and Savelsbergh 2024) for efficient practical implementation .

In line with this body of research, the focus of this thesis is to develop models and algorithms that foster on-time and on-demand deliveries while improving delivery efficiency, reducing costs, enhancing reliability, and achieving high-quality solutions efficiently. To this end, we investigate three distinct delivery models, each addressing specific challenges and opportunities in last-mile logistics.

In Chapter 1, we explore crowdkeeping delivery services, a novel form of crowdsourcing that leverages unused crowd space for temporary parcel storage. This approach reduces delivery costs, decreases delivery failures, and enhances service quality. The "crowd keepers", driven by incentives to provide services, temporarily store parcels on behalf of delivery companies and transfer them to customers at their earliest convenience. Companies such as Pickme (2024) in France and CaiNiao (2023) in China are implementing these crowdkeeping delivery services. This business model motivates the crowd to provide the service by offering compensation and encourages customers to use it through a lower pickup fee. It respects the independent decision-making of both customers and keepers while balancing cost minimization for customers with profit maximization for keepers. We present a bilevel program for the problem that jointly determines the assignment, routing, and pricing decisions while considering customer preferences, keeper behaviors, and platform operations. We then develop equivalent single-level programs that can be solved to optimality using a row generation algorithm, as well as high-quality approximations with estimated optimal travel times to solve the problems approximately but more efficiently. The numerical study using a real-world dataset from Amazon shows that the crowdkeeping delivery system has the potential to generate higher profits due to its ability to consolidate deliveries and eliminate failed deliveries.

In Chapter 2, we examine ultra-fast delivery services, which focus on transporting food, groceries, and essentials from micro-depots to customers within tight timeframes, often as short as 15 minutes. For instance, Getir (2022) in Turkey, Gorillas (2022) in Europe, and Goodfood (2022) in Canada have aimed to provide fast delivery services

within 15 to 30 minutes. Sharing the same principles of proximity, sustainability, and accessibility as the 15-minute city (Moreno et al. 2021), this model reduces reliance on cars, cuts fuel consumption and pollution, and improves customer satisfaction. By strategically locating micro-depots and allocating customers while accounting for uncertainties in travel times and demand arrivals, the model aims to maximize profits while ensuring reliable and timely service. We develop robust probabilistic envelope constrained (PEC) programs to handle uncertainties and optimize the protection level to avoid both excessive risk and conservatism. To enhance the tractability of PEC models, we derive their equivalent semi-infinite linear programs and propose inner and outer approximations with finite linear constraints. The numerical study using a real-world dataset from Amazon and the Google API shows that a daily service level that prioritizes time periods with higher order frequencies and applies multi-layer partial protection yields high profitability with mild violations of service level guarantees. This strategy proves to be effective for profitable and reliable ultra-fast delivery without over-committing or under-performing, regardless of ordering times or traffic conditions. Additionally, empirical evidence indicates that providing ultra-fast delivery in rural areas poses unique challenges compared to urban settings.

In Chapter 3, we investigate multi-store consolidated-order delivery services, which enable customers to place orders from multiple stores in a single transaction, with the orders fulfilled through combined deliveries. For instance, DoubleDash (2023) and Instacart (2022) in the US and Epipresto (2023) in Canada allow customers to shop from multiple stores in a single transaction without an additional delivery charge, ensuring that all items are delivered together by the same driver. We further consider that orders can be handled by different drivers for pickup, allowing partial order transfers among drivers at transshipment points. Consolidation saves on delivery fees for customers and enhances operational efficiency for companies, while transshipment can further improve delivery times by avoiding unnecessary detours to multiple far-apart stores. The process involves optimizing driver assignments to pick up and deliver customer orders, planning efficient delivery routes under time and capacity constraints, and facilitating partial order trans-

fers at selected transshipment points. We develop a mixed-integer linear program for the multi-store order problem with consolidation and transshipment and adopt a learning-to-optimize approach that integrates machine learning and optimization to provide high-quality solutions efficiently through offloading a portion of the computational workload to the offline phase. The numerical study using a real-world dataset and implemented in a dynamic environment shows that the consolidated-order delivery with transshipment, coupled with a short-duration waiting strategy, consistently delivers superior performance through spatial and temporal consolidation. The optimal waiting strategy varies depending on customer arrival rates and driver availability relative to customer demand.

In summary, the business models considered in this thesis focus on efficient delivery operations that address the diverse and dynamic needs of modern consumers for fast, flexible, and reliable services. These models contribute to the overall success of e-Commerce by ensuring seamless integration between order fulfillment, last-mile logistics, and customer experience. Additionally, they emphasize the balance between operational efficiency and service quality, both of which are essential in modern online retail. For each business idea, we formulate mathematical programs that capture real-world challenges in last-mile logistics and develop efficient solution methods to obtain high-quality solutions. Through conducting numerical experiments using real-world datasets, we provide managerial insights into how emerging delivery strategies can transform last-mile logistics into a more cost-effective, reliable, and customer-centric process, ultimately enhancing the efficiency and competitiveness of e-Commerce operations.

Chapter 1

Crowdkeeping in Last-Mile Delivery

Abstract

In order to improve the efficiency of the last-mile delivery system when customers are possibly absent for deliveries, we propose the idea of employing the crowd to work as keepers and to provide storage services for their neighbors. Crowd keepers have extra flexibility, more availability, and lower costs than fixed storage such as automated lockers, and this leads to a more efficient and a more profitable system for last-mile deliveries. We present a bilevel program that jointly determines the assignment, routing, and pricing decisions while considering customer preferences, keeper behaviors, and platform operations. We develop an equivalent single-level program, a mixed-integer linear program with subtour elimination constraints, that can be solved to optimality using a row generation algorithm. To improve the efficiency of the solution procedure, we further derive exact best response sets for both customers and keepers, and approximate optimal travel times using linear regression. We present a numerical study using a real-world dataset from Amazon. The fixed-storage and the no-storage systems are used as benchmarks to assess the performance of crowdkeeping system. The results show that the crowdkeeping delivery system has the potential to generate higher profits due to its ability to consolidate deliveries and to eliminate failed deliveries.

1.1 Introduction

E-commerce is thriving. The number of sales has almost tripled from 2014 to 2019 (Deloison et al. 2020). This boom has led to an unprecedented volume of goods being shipped every day. Customers are more demanding than ever in terms of the quality of delivery services: they are expecting to receive orders at any time they want (M. Ulmer and Savelsbergh 2020), and their expectations for speed forces e-tailers to offer same-day delivery with small time windows (Savelsbergh and Van Woensel 2016; Koch and Klein 2020). Such services lead to costly last-mile deliveries, which constitute the final stage in the delivery process when a product is transported and delivered to a customer. Indeed, last-mile services can comprise up to 41% of the total cost to move goods (Jacobs et al. 2019).

Several novel technologies and business models, including crowdshipping, drones, autonomous robots, and parcel lockers, have emerged. Common goals in such innovations are cost reduction or improved service quality. Sharing the same goals, we propose an innovative business model in last-mile delivery, referred to as *crowdkeeping*. Crowdkeeping, a new form of crowdsourcing in last-mile delivery, aims to utilize the unused space within the crowd for storage and has the potential to reduce delivery costs, eliminate delivery failures, and improve service quality. Broadly defined, it involves employing the crowd to keep parcels locally until customers pick them up. In other words, the *'crowd keepers'*, who volunteer to act as keepers and provide keeping services, initially attend the delivery on behalf of customers and transfer parcels to customers on behalf of the delivery company.

Compared to pickup points and automated lockers that are used in the real world, the availability and capacity of crowd keepers are higher, the cost of using crowd keepers is lower, and crowd keepers are more flexible to adapt to different customer groups in different time periods. Moreover, it can be implemented without substantial additional infrastructure and with modest operational cost. We consider an online service platform that coordinates customers and keepers to reduce the delivery costs and to improve the overall profitability of the delivery company.

We contribute to the current research on delivery logistics from three aspects:

- We propose the idea of crowdkeeping for last-mile delivery systems, and present its concept, viability, benefits, and operational framework.

- To model the behaviors of all participants in the delivery system, including customers, keepers, and the platform, we present a bilevel program that jointly considers the assignment, routing, and pricing decisions. We make use of the duality theory to obtain an equivalent mixed-integer linear programming formulation, and develop a row generation algorithm to find the exact optimal solutions. To further improve the efficiency of the solution procedure without sacrificing the effectiveness, we derive explicit expressions for the best responses of customers and keepers, and propose an approximation model by approximating optimal travel times using linear regression.

- We carry out extensive experiments on a real-world dataset to investigate the effectiveness of the delivery system, the efficiency of the solution procedure, and how these are influenced by factors such as the number of customers, customer absence ratio, keeper service range, and various related costs. We find that crowdkeeping has the potential to improve customer service levels, increase platform profits, and enhance the overall cost-efficiency and environmental friendliness of the delivery system.

The paper is organized as follows. Section 1.2 reviews the related literature in last-mile delivery. We define the problem setting in Section 1.3, present a bilevel program in Section 1.4, and develop the solution methodology in Section 1.5. We finally carry out an extensive numerical study in Section 1.6 and present our conclusions in Section 1.7. We also refer the reader to Appendix for a list of alternative delivery systems and for all proofs.

1.2 Literature Review

In this section, we first review the three most common types of last-mile delivery and their associated challenges. Subsequently, we delve into the most-recent innovations addressing these challenges, including lockers in Section 1.2.2, crowdsourcing in Section 1.2.3, and demand management in Section 1.2.4. Undoubtedly, the problem considered here is closely related to the traveling salesman problem (TSP) and its variants, constituting a vast body of knowledge. We lastly review the related literature in Section 1.2.5.

1.2.1 Last-Mile Delivery Types, Challenges, and Innovations

There is a wide range of products being shipped and delivered every day. According to the necessity of the customer presence and the coordination of delivery time windows, deliveries are categorized into three types. In 'unattended home delivery' (UHD), customer presence is not needed since a parcel is left at the doorstep with no attendance requirement. In 'attended home delivery' (AHD), the customer is required to be present at the time of delivery, for instance, an important document that requires a signature, a high-tech computer, or groceries shipped from a local store. In AHD, the company and the customer can either agree or not on a delivery time window, a.k.a. coordinated AHD (c-AHD) and uncoordinated AHD (u-AHD), respectively.

Each delivery type poses different challenges. In UHD, coordination and customer absence is not a concern. However, theft and weather conditions pose important risks. There are also risks associated with denial-of-receipt or burglary at the house (McKinnon and Tallam 2003). In u-AHD, not finding the customer at home causes inefficiencies since it requires a second trip to the same customer. In c-AHD, timing is important. Companies offer limited number of delivery time slots to customers and each time slot comes potentially with a different delivery price (M. W. Ulmer 2020; Koch and Klein 2020). Time windows can increase the delivery costs significantly, because consolidation may not be possible for parcels destined to the same region. When customers cannot find a suitable time slot that fits their needs, the demand (and therefore the revenue) is lost.

Last-mile delivery is a growing field to deal with these challenges, that is, to make the deliveries on-time and low-risk, to eliminate failed deliveries, and to reduce the delivery costs. The goals are achieved either by improving the operational procedures with self-service lockers and crowdsourcing, or by managing the demand.

1.2.2 Self-Service Locker Systems

Self-service locker systems are proposed to alleviate the risk of theft, to protect from unfavorable weather conditions, and to provide consolidation of parcels. Motivated by an example of Singaporean companies experimenting with a set of shared parcel lockers, Lin et al. (2020) propose a quantitative approach to determine the optimal locker locations with the objective to maximize the overall quality of services. Schwerdfeger and Boysen (2020) further consider the dynamic relocation of parcel lockers during the day. Rohmer and Gendron (2020) extensively investigate different delivery concepts that exploit parcel locker stations and their associated decision problems.

There are unfortunately major disadvantages of employing lockers. First, setting up a network of lockers requires large initial investment costs, which can eventually lead to small returns. Currently, there is a lack of a dense locker network. The ownership of lockers is also a major problem. Hasija, Shen, and Teo (2020) argue that, due to the proprietary nature of such systems, the utilization of lockers tends to be low. The lockers can also be shared among multiple firms, in which case the assignment of capacities becomes a concern. Shared or not, the use of lockers may result in rental costs, which could eventually be imposed on customers. Joerss et al. (2016) also report that *"Somewhat surprisingly, unattended delivery to parcel lockers does not really appeal to consumers despite the possibility of picking up their parcel 24/7"*. The authors report that customers put large value on home delivery instead of going to the lockers and conclude that their wide utilization is unlikely.

In our understanding, the locker system is a viable option for delivery, especially when the confidentiality of items is a concern. Therefore, lockers represent an important

option of the last-mile delivery problem and add value to the overall system. However, the benefits of using automated lockers are limited due to setup and maintenance costs, as well as the restricted number of available locations. The crowdkeeping system, on the other hand, provides a comparable service without requiring substantial additional infrastructure, other than setting up an online platform. Even though compensation is necessary to offer keepers proper incentives, it can be adapted as capacity changes. In the new system, the cost for unused capacities is avoided.

1.2.3 Crowdsourcing

Carbone, Rouquet, and Roussat (2017) conceptualize the applications of crowdsourcing in logistics by reviewing the websites of 57 initiatives. The authors argue that most of these initiatives mainly offer two types of logistics services: crowdshipping and crowd storage. Crowdshipping is the transportation of parcels by the crowd in return for a compensation and is offered as an option in the last-mile delivery. There is a high level of interest for crowdshipping both in practical applications and in the scientific literature. A. M. Arslan et al. (2019) report that several companies use crowdshipping partially or completely in their delivery operations. The authors investigate the benefits of crowdshipping by considering a platform that matches parcel delivery tasks and ad hoc drivers in real time. All requests are essentially served. A related problem is the online vehicle routing problem with occasional drivers (Archetti, Guerriero, and Macrina 2021), in which a penalty is incurred for not serving a customer or for violating the time window constraints. Dayarian and Savelsbergh (2020) consider crowdshipping by employing in-store customers to deliver online orders. M. Ulmer and Savelsbergh (2020) study the problem of keeping a scheduled delivery workforce along with crowdsourcing to hedge against the uncertainty in crowdsourced delivery capacity. Qi et al. (2018) study shared mobility in last-mile delivery by optimally sizing the service zones. They argue that crowdshipping is not a scalable alternative of the conventional truck-only system in terms of operating costs, but that a combined operational mode can provide flexibilities and benefits. For a recent re-

view on multiple dimensions of crowdshipping, we refer to Le et al. (2019) and Alnaggar, Gzara, and Bookbinder (2021). Crowd storage, on the other hand, is considered as a logistics operation in rental of storage areas such as cellars, spare rooms, garages, or yards. It is considered as a local service that is particularly suitable in urban areas for those who need to store furniture or similar items for long terms. To the best of our knowledge, crowd storage is still not considered for last-mile deliveries.

We define crowdkeeping by introducing the idea of crowd storage into the last-mile delivery. That is, keepers can provide storage services for their neighbors, or neighbors can temporarily store parcels for absent customers. In fact, delivering a parcel to a neighbor is not an entirely new idea. Jacobs et al. (2019) report that *"55% of consumers can accept the service of delivering products to neighbors in their vicinity"*. There are also empirical evidences that neighbors can cooperatively undertake delivery tasks with little or no compensation, and that 70% of customers in a survey reported that they can make deliveries for less than $5 (Devari, Nikolaev, and He 2017). Nevertheless, there is no formal way of delivering to a neighbor. In the current operations, the courier needs to search for an available neighbor to deposit the parcel when the customer is absent from the delivery (McKinnon and Tallam 2003). In our study of crowdkeeping, neighbors are incentivized by a monetary compensation to participate in the delivery process as crowd keepers. Then keepers are selected by the platform to serve multiple customers before deliveries. In this case, deliveries are consolidated, and the additional task of searching for an available neighbor is eliminated. Customers then pick up their parcels possibly by walking. At this point, it is worth mentioning that walking is also reported as a mode of transportation in crowd logistics (Carbone, Rouquet, and Roussat 2017): *"Transport resources can be vans, cars, scooters, bicycles, public transport, or even walking"*. We identified a single study in the literature considering walking as a form of transportation in crowdshipping. Martinez-Sykora et al. (2020) consider drivers making deliveries in dense urban areas by walking at the end of their vehicle trip in crowdshipping to avoid heavy traffic.

1.2.4 Demand Management

There is extensive research in the area of demand management for last-mile deliveries. Our work is most related to the service time slot, the service price, and customer incentives. E-tailers can offer different delivery time windows and associated prices to manage the demand. Several static and dynamic demand management strategies have been investigated including differentiated time slot allocation and differentiated time slot pricing.

In the time slot allocation dimension, Agatz et al. (2011) study assigning time slots to zip codes in a service region to minimize the delivery costs. Spliet and Gabor (2015) introduce the time window assignment vehicle routing problem, in which time windows have to be assigned before demand is realized. Bruck, Cordeau, and Iori (2018) study the problem of creating time slot tables and routing technicians in a cost-effective way. A decision support system for an Italian company is also developed for a related problem (Bruck, Castegini, et al. 2020).

In the time slot pricing dimension, Yang and Strauss (2017) present a delivery cost approximation scheme by decomposing the delivery problem into a collection of smaller problems. The customers' delivery time slot choices are estimated using a multinomial logit model. Klein, Neugebauer, et al. (2019) study differentiating the time-slot pricing by considering the routing phase. The customers' choice behavior is modeled as a general nonparametric rank-based choice model. These authors study two policies for incorporating the routing costs, by explicitly incorporating the routing constraints to their model or by using a model-based approximation and find that the latter can be used in real-world applications. In a similar line of research, Klein, Mackert, et al. (2018) present a cost approximation approach for dynamic time slot pricing decisions by forecasting the potential future customers. Koch and Klein (2020) additionally combine dynamic pricing with dynamic vehicle routing.

Another interesting idea in demand management is incentivizing customers. One of the first papers in customer incentives is Campbell and Savelsbergh (2006), who investigate the use of incentives for demand management to reduce the delivery costs. M. W.

Ulmer (2020) considers anticipatory pricing and routing policy method for the same-day delivery, in which customers are incentivized to select delivery deadline options efficiently to align with routing considerations. Yildiz and Savelsbergh (2020) consider offering a discount to customers on their delivery fee in return for flexibility to adjust a previously agreed upon delivery window. The authors report that the cost savings of offering discounts can exceed 30%.

These studies on demand management reveal the importance of the time slot management, the pricing for services and incentives, and the integration of pricing and routing. However, some customers may be unavailable in any of the offered time slots, and this implies lost revenues. Our approach can provide on-time deliveries but does not have to enforce the time slot management, and it can jointly consider the service pricing, the incentive pricing, and the routing decisions.

1.2.5 Traveling Salesman Problem and Variants

The Traveling salesman problem is one of the most classical problems in logistics and its extensions attract significant attention due to their economic importance, theoretical challenge, and applicability in many real-world contexts (Vidal, Laporte, and Matl 2020). It consists of finding one route from a depot such that all customers are visited and the total cost is minimized. We refer the reader to Applegate et al. (2007) and to Toth and Vigo (2014) for related problems, methods, and applications. When customers are not necessarily visited by a vehicle and it is sufficient to visit another close-by node in the network, the problem is then called covering-tour problem (CTP) (Gendreau, Laporte, and Semet 1997). The problem we consider in this paper is closely related to CTP, because keepers in the same vicinity of customers cover the customer nodes. Exact solutions of CTP and its variants are notoriously difficult to obtain. A branch-and-price algorithm is introduced by Jozefowiez (2014) for solving CTP. The pricing subproblem is a ring-star problem, which is solved using a branch-and-cut algorithm. Kartal, Hasgul, and Ernst (2017) introduce the single allocation p-hub median location and routing problem

with simultaneous pick-up and delivery. Other closely related problems are location-or-routing problem (LoRP) by O. Arslan (2021), and location-and-routing problem (LRP) by Kartal, Hasgul, and Ernst (2017). In LoRP, the location decision is related to depots, the vehicles are dispatched from the selected depots, the depots are not connected to each other, and a customer is served either by being covered by a located depot or by being visited by a vehicle routing. In LRP, the depots are connected, the vehicles are dispatched from the selected depots, and all customers are visited by a vehicle routing. In our crowdkeeping delivery problem (CDP), the location decision is related to keepers and customers, a vehicle is assumed to visit a subset of nodes (i.e., the active nodes), and the inactive nodes are covered by the active nodes. The CDP bears similarities with two-echelon vehicle routing problems (Perboli, Tadei, and Vigo 2011), but it is enriched by the inclusion of pricing mechanisms and coordination among the different participants.

1.3 Problem Description

This section introduces the crowdkeeping framework by comparing it with the standard operational framework for deliveries. It describes the behaviors and decisions of keepers, customers, and the platform and lists the potential benefits and real-world applications of crowdkeeping.

1.3.1 Operational Framework

In the standard operational framework for delivery systems (Figure 3.2(a)), the delivery phase is a combination of order preparation and parcel transportation. In the order preparation phase, companies receive order requests, pick up items, and pack them as parcels for deliveries. In the transportation phase, parcels are in transit from their origin depots to their destination depots, and delivered to customers in last mile delivery.

In our new business model (Figure 3.2(b)), we decompose the last-mile delivery process into two steps. First, parcels are delivered to keepers who store them for customers.

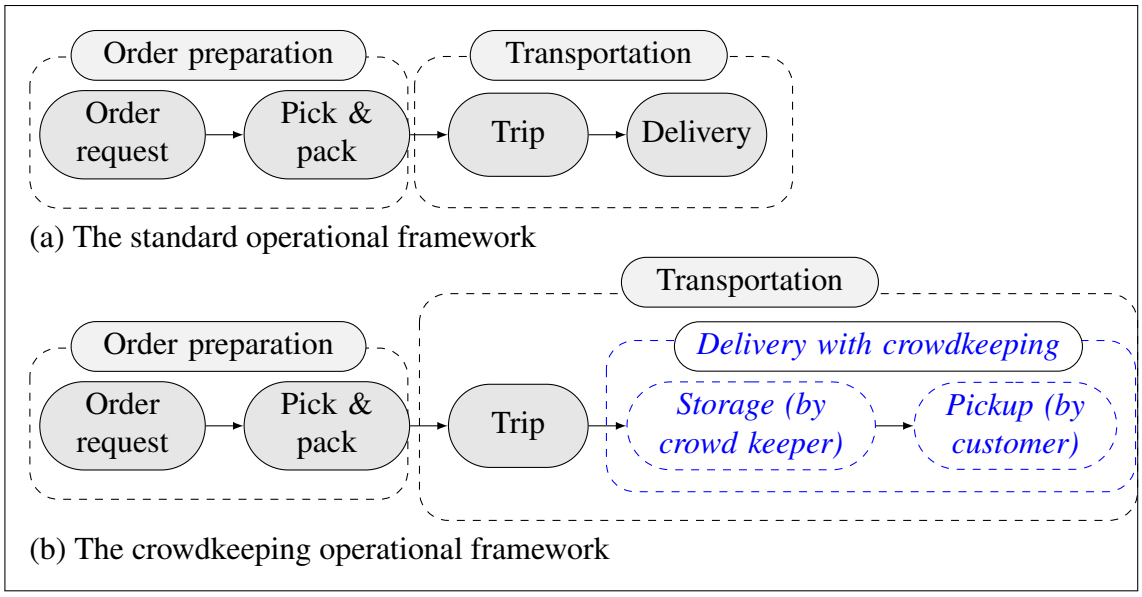


Figure 1.1 – Comparison of the standard and the new operational frameworks for delivery systems

Second, customers pick up their parcels from their selected keepers to finish the delivery process. Decomposition of tasks allows deliveries to be coordinated with absent customers and also with crowd keepers who have more flexibility to consolidate orders in their neighborhood.

1.3.2 Implementation

We now explain the implementation details of crowdkeeping in reality by taking Pickme (2024) as an example, which is a platform that provides parcel reception service, allows keepers to earn compensation by storing parcels for their neighbors, and aims to avoid failed deliveries. Detailed steps of implementation are shown in Table 1.1 and Figure 1.2. In the first stage, the platform determines the pricing for the service and pre-assigns potential keepers to customers. In the second stage, customers and keepers make decisions based on their preferences. After observing the behavior of keepers and customers in this stage, the platform makes the final assignment decisions in the third stage and plans the deliveries. Finally, upon receiving the parcels, keepers notify customers of the pickup

location and time slots, and hand over the parcels to complete the deliveries.

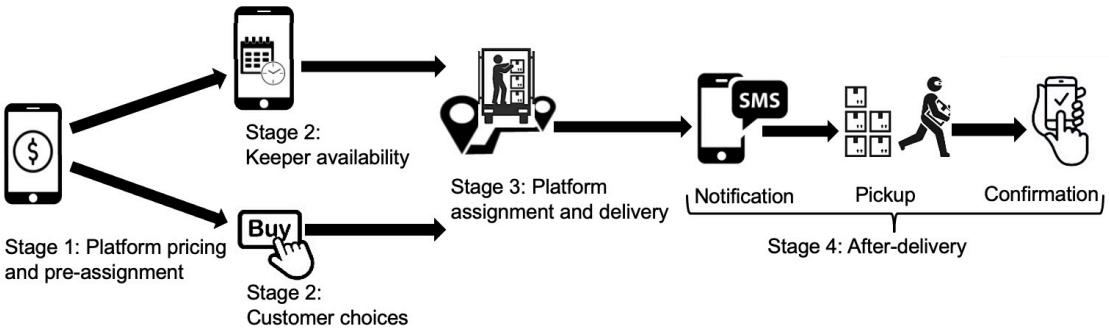


Figure 1.2 – Implementation of the crowdkeeping operational framework.

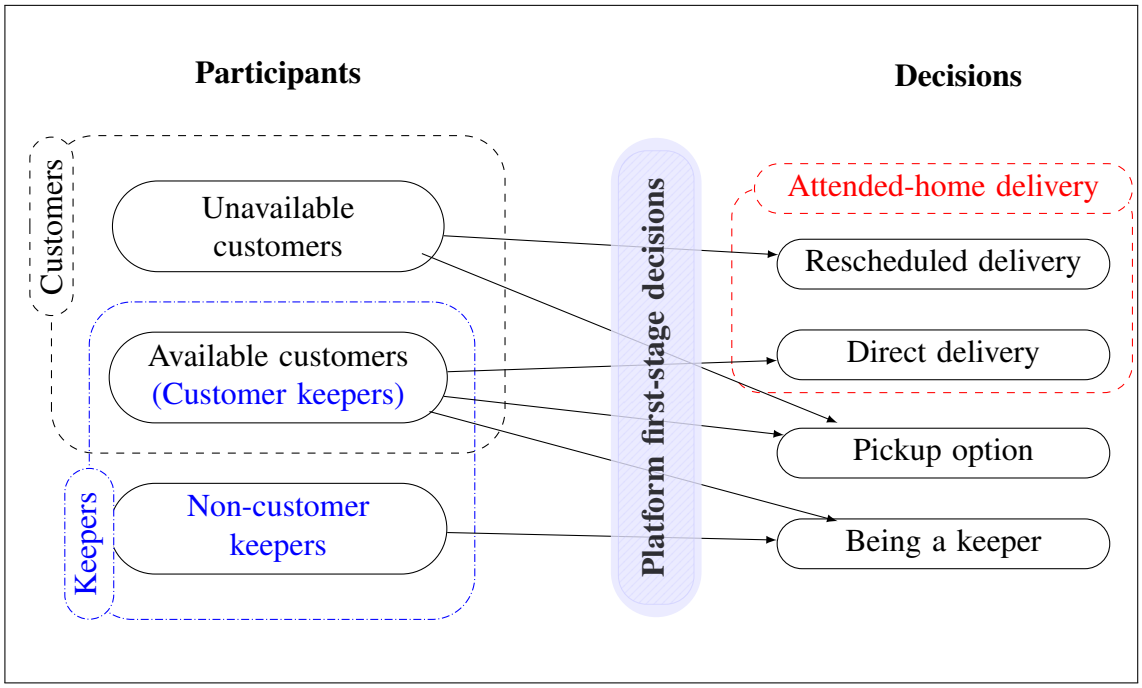
1.3.3 Participants and Their Behaviors

In the crowdkeeping delivery system, there are three groups of players: customers, keepers and the platform. The ***customers*** are people who purchase products online and expect their parcels to be delivered. The ***keepers*** are individuals, such as homemakers, stay-at-home parents, people working from home, and unemployed persons, that can receive parcels and temporarily store them. This term makes a clear reference to the duty that such an individual performs and emphasizes the functional difference from the "*courier*" generally used in crowdshipping. Similar to other supply sides on crowdsourcing platforms, keepers work in reputation-based systems and normally receive a compensation for every customer they serve. Different from other crowdsourcing platforms, the entry to the crowdkeeping market is simpler, because it only requires a smart phone and no investment or special equipment is necessary. Coupled with the mobile application, a smart phone is capable of updating the tracking information, specifying the pickup location and duration, and collecting the customer signature to ensure the safety and the convenience of the delivery process. The third participant, the ***platform***, coordinates the deliveries between customers and keepers. The delivery company delivers the product to the keeper and, in doing so, has more flexibility to consolidate orders in the same neighborhood.

Table 1.1 – Implementation details

Stage	Description
Stage 1:	*Platform pricing and pre-assignment*: The platform commits to compensating the crowd for their crowdkeeping services, encouraging them to register as potential keepers by providing their information, including address and available time slots. It also displays delivery and pickup fees to customers and encourages them, including those who may be absent, to use the parcel storage service. Additionally, the platform presents the proximity of potential keepers to nearby customers, helping them estimate the walking time for the pickup option and provides a minimum guaranteed compensation to keepers, helping them decide whether to provide the service or not.
Stage 2:	*Keeper availability*: Potential keepers who are pleased with the proposed compensation can choose to make themselves available and provide the requested capacity. Otherwise, they make themselves unavailable. *Customer choice*: Customers choose between two options: *pickup* and *attended-home delivery*. If customers agree to be assigned by the platform subsequently to pick up their parcels from any of the listed pre-assigned keepers, they will be charged a lower pickup fee than the delivery fee. Attended-home delivery includes two choices: *direct delivery*, which takes place as soon as possible, and *rescheduled delivery*, which is delayed to another day. If customers prefer attended-home delivery but declare themselves as absent, then their delivery will be rescheduled for another day. Otherwise, the default option is direct delivery. Additionally, customers who choose direct delivery can also work as keepers.
Stage 3:	*Platform assignment and delivery*: The platform assigns inactive customers, who prefer the pickup option, to available keepers while respecting the preferences of keepers and customers. Carriers deliver the parcels to active keepers who serve customers and to active customers who choose the direct delivery. Keepers and customers are allowed to modify their availability before the delivery. If the selected keeper refuses to receive parcels due to any reason, the parcels will be redirected to another available keeper. In the case that all keepers are unavailable, rescheduling the delivery becomes the backup option.
Stage 4:	*Notification*: Keepers are notified of the arrival of the delivery by the platform. After receiving parcels, keepers use their smartphone to scan these parcels, which updates the tracking information and contacts customers to arrange an appointment for picking up. *Pickup*: Customers who choose the pickup option receive an identification code beforehand by email or text message, and must show this code to the keeper to secure the exchange when collecting their parcels. *Confirmation*: Keepers enter the customer's code and scan the package to confirm the transaction, and they are instantly rewarded once the delivery of the parcel to its recipient is confirmed.

From the platform's perspective, customers represent the *demand*, and keepers represent the *supply*, and the objective is to match the demand and supply in this market.



Notes. The participants are on the left side and each arrow represents a potential option for a participant. They make their own decisions, which are guided (or filtered) by the platform's first-stage decisions.

Figure 1.3 – Problem setting of the crowdkeeping delivery system.

Observe that customers may be absent during deliveries. Therefore, we categorize customers as *available customers*, who are present for attended home deliveries, and *unavailable customers*, who are absent during deliveries. The available customers are also referred to as *customer keepers* because they can additionally provide storage services in their neighborhood. In our problem setting, we also consider keepers who may not necessarily be a customer receiving a parcel but declare their availability to provide storage services. This group is called *non-customer keepers*. Both customer keepers and non-customer keepers are included in the group of keepers (i.e., crowd keepers). These participants and their options are displayed in the left side of Figure 1.3.

We now discuss the choices of each participant. The platform prices the pickup fee,

which is lower than the standard delivery fee to encourage customers to use the crowdkeeping service. The platform also prices the compensation, an incentive to attract the crowd to work as keepers and provide the crowdkeeping service. Customers then choose between the attended-home delivery (subject to a delivery fee) and the pickup option (subject to a pickup fee and a short walking distance). For the available customers who prefer the attended-home delivery, the direct delivery is the default option. The deliveries of the unavailable customers that prefer the attended-home delivery yet cannot attend the direct delivery are rescheduled to another day. Both the non-customer keepers and available customers that choose the direct delivery have the option to be potential keepers and provide storage services. Finally, the platform assigns the customers that prefer the pickup option to the available keepers, and the carrier visits the *active customers* (who prefer the direct delivery) and the *active keepers* (who serve the other customers).

1.3.4 Benefits and Challenges

The concept of using pickup locations to reduce failed deliveries and improve last-mile delivery efficiency is well-established. Companies such as Amazon and IKEA operate dedicated pickup points in cities, allowing customers to retrieve their orders. These locations are typically staffed by full-time employees, remain fixed, and are designed for long-term use, making them less adaptable to changing demands. Similarly, automated lockers provide self-service pickup options, providing convenience and flexibility. We define such pickup points and lockers as *fixed storage* since they both operate from permanent locations. For further details on delivery systems using fixed storage, see Appendix 1.8.

Compared to fixed storage, crowd keepers have the advantages of offering extra flexibility, more availability, and lower costs. There are concrete benefits of crowdkeeping for distribution companies, customers, and keepers. For distribution companies, crowdkeeping improves operational efficiency by consolidating deliveries in both time and space. The flexibility and availability of the crowd helps increase delivery capacity while re-

ducing failed deliveries. For customers, crowdkeeping enhances convenience as they can pickup parcels in person whenever they are available. It also increases parcel security through direct supervision by the keepers. Additionally, customers gain an alternative to standard home delivery by opting for a lower-cost pickup service. For keepers, individuals acting as crowd keepers can earn compensation or rewards with minimal investment or setup costs. Unlike fixed storage, which incurs setup costs regardless of usage, crowd keepers are only compensated when they are assigned tasks, making the system more adaptable and cost-efficient.

Despite its benefits, crowdkeeping introduces concerns about legal responsibility and parcel security, as non-professional keepers temporarily store customer packages. While legal and regulatory issues are beyond the scope of this study, lessons from existing crowdshipping platforms can be applied to crowdkeeping. Furthermore, in local neighborhood settings, established relationships between participants reduce legal risks. Real-world applications such as Pickme (2024) and CaiNiao (2023) demonstrate the viability of crowdkeeping, reinforcing its potential as a practical alternative to fixed storage.

1.4 Bilevel Program for Crowdkeeping Delivery Problem

We define the Crowdkeeping Delivery Problem (CDP) and present models for the customer, the keeper, and the platform. Subsequently, we formulate a bilevel program for the CDP, with the platform as the leader and customers and keepers making decisions simultaneously as followers.

Definition 1 *The **Crowdkeeping Delivery Problem** is defined as pricing the compensation and the pickup fee that maximize the platform profit by respecting independent decision making mechanisms of customers and keepers, which involve minimization of the delivery service cost for each customer and maximization of the profit for providing storage services for each keeper.*

Table 1.2 – Notations

Sets	Description
$\mathscr{N}$	the set of all customers, including the unavailable customers and customer keepers
$\mathscr{M}$	the set of non-customer keepers
Variables for platform decisions made in the first stage	
f^p	the pickup fee offered to customers
$\mathfrak{c}$	the compensation offered to keepers
m_i	the maximum number of customers proposed to be served by keeper i, (i.e., the requested capacity by the platform)
v_{ij}	1 if platform shows potential keeper j to customer i, 0 otherwise
Variables for customer or keeper decisions made in the second stage	
u_i	1 if customer i chooses the direct delivery, 0 otherwise
w_i	1 if customer or keeper i makes themselves available to serve others, 0 otherwise
z_i	1 if customer i chooses to reschedule the delivery, 0 otherwise
Variables for platform decision made in the third stage	
$\hat{v}_{ij}$	1 if platform assigns keeper j to serve customer i, 0 otherwise
x_{ij}	1 if arc (i, j) appears on tour, 0 otherwise
y_j	1 if participant j is active and needs to be visited, 0 otherwise
$\hat{z}_i$	1 if the delivery of customer i is rescheduled by the platform due to the lack of available nearby keepers, 0 otherwise
Parameters	
f^d	the standard delivery fee offered to customers
t_{ij}	the travel time between i and j
a_i	1 if customer i is absent for deliveries, 0 otherwise
b_j	the capacity of keeper j
e_i	1 if node i is a customer, 0 if node i is a non-customer keeper
r_{ij}	1 if customer i and keeper j are located in the service zone, 0 otherwise (with $r_{ii} = 0$ to model the infeasibility of serving oneself)
c^p	the inconvenience cost per minute of walk time for picking up
c^d	the truck delivery cost per minute of travel time
c^r	the inconvenience cost of rescheduling a delivery incurred by customers themselves
$\hat{c}^r$	the penalty for rescheduling a delivery as a result of the platform's decision
c^k	the fixed inconvenience cost of being available as a keeper
c^s	the marginal inconvenience cost for serving each parcel as a keeper

Descriptions of the notation are given in Table 1.2. According to the implementation details shown in Section 1.3.2 and the decisions of each participant involved in the CDP as presented in Section 1.3.3, Figure 1.4 illustrates the timeline of decisions made by each of them.

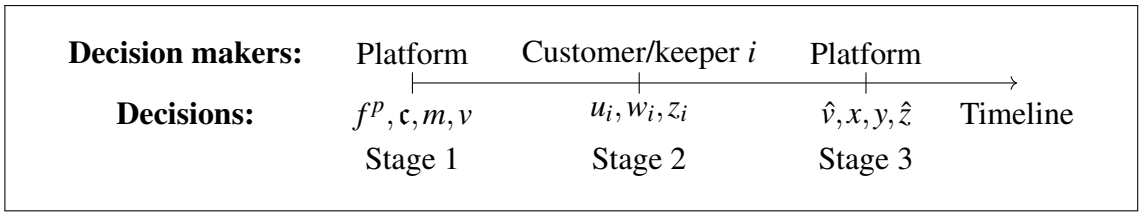


Figure 1.4 – Timeline of decisions made by the platform, customers, and keepers

We assume that the delivery operations are carried out under the condition of full information. In the first stage, the platform prices the pickup fee (f^p), shows the potential keepers to each customer (v), prices compensation ($\mathfrak{c}$), and determines the maximum number of customers to be served by each keeper (m). In the second stage, given the first-stage decisions, customers choose between the delivery and the pickup options. Customers who prefer the attended-home delivery choose the direct delivery ($u_i = 1$) if available, or reschedule the delivery ($z_i = 1$) if absent. Keepers, including customer keepers who choose direct delivery, show their availability and become potential active keepers ($w_i = 1$) if they are satisfied with the to-be-earned compensation by serving a certain number of customers. In the third stage, the platform assigns those customers who prefer the pickup option to potential keepers ($\hat{v}$) by respecting their preferences, or reschedules the delivery ($\hat{z}$) if those customers cannot be served by any nearby available keeper. The platform then plans the visit to all active nodes (x, y) to complete the deliveries. In summary, the service fee, the compensation and the pre-assignment are revealed before customers and keepers make their decisions, and customer demands and keeper availabilities are revealed before the assignment and routing planning takes place. These assumptions lead to a form of three-stage Stackelberg game, in conformity to the first three stages in Table 1.1 and Figure 1.2.

1.4.1 Customer and Keeper Models

Available customers have three choices. The first one is to pay the standard delivery fee and to have their parcels delivered to their doorstep (*Direct delivery* in Figure 1.3). The second one is to pay the pickup fee and to pick up their parcels from one of those potential keepers (*Pickup option*). They also have the third option of working as a crowd keeper in addition to receiving their own parcels (*Being a keeper*). Absent customers, on the other hand, have two choices: they can pick up their parcels or reschedule their delivery (*Rescheduled delivery*). The model for customer keeper i is as follows:

$$
\begin{aligned}
H_i(f^p, \mathfrak{c}, m_i, v_{i:}) \triangleq \min_{u_i, w_i, z_i} \quad & f^d(u_i + z_i) + \left[c^k + (c^s - \mathfrak{c})m_i\right] w_i \\
& + (f^p + c^p \max_{j \in \mathscr{M} \cup \mathscr{N}} t_{ij} v_{ij})(e_i - u_i - z_i) && \text{(1.1a)} \\
\text{s.t.} \quad & u_i \leq e_i(1 - a_i) && \text{(1.1b)} \\
& z_i \leq e_i a_i && \text{(1.1c)} \\
& e_i w_i \leq u_i && \text{(1.1d)} \\
& m_i w_i \leq b_i && \text{(1.1e)} \\
& u_i, w_i, z_i \in \{0, 1\}. && \text{(1.1f)}
\end{aligned}
$$

Given the platform's decision on the pickup fee f^p, the compensation $\mathfrak{c}$, the maximum number of to-be-served customers m_i, and the potential keepers from where customer i could pick up their parcels $v_{i:}$ (denoting a row vector), $H_i(f^p, \mathfrak{c}, m_i, v_{i:})$ is the optimization model of customer i. Using this model, customers decide whether they prefer a direct delivery (u_i), acting as a keeper (w_i) for a total compensation of $\mathfrak{c}m_i$, rescheduling the delivery (z_i), or picking up from any of the pre-assigned keepers that will be designated by the platform later. The objective function (1.1a) states that each customer minimizes the total amount they pay to receive their parcels. If customers choose the delivery option, whether it is the direct delivery or rescheduled delivery, they need to pay the delivery fee f^d. When working as crowd keepers, they earn a compensation $\mathfrak{c}$ for keeping each parcel and incur both a fixed inconvenience cost c^k for making themselves available and at a marginal cost c^s for serving each parcel. If customers choose the pickup option, they

need to pay the pickup fee f^p and the inconvenience cost c^p for each minute of pickup walk time. Customers anticipate the walk time according to the travel duration from their homes to the farthest pre-assigned keeper. This means that if customers prefer pickup, they are open to being assigned to any of the pre-assigned keepers. Note that a fixed inconvenience cost for walking can also be accounted as part of the pickup fee f^p. The parameter e_i indicates whether participant i is a customer or non-customer keeper. This implies that $e_i = 1$ for all customer $i \in \mathcal{N}$, and that $e_i - u_i - z_i = 1$ when the customer chooses the pickup option. Constraints (1.1b) and (1.1c) state that customers preferring the attended-home delivery have to reschedule the delivery if absent ($a_i = 1$), and that they have no reason to delay the delivery if available ($a_i = 0$). Constraints (1.1d) and (1.1e) state that customers can work as crowd keepers only when they are available for the direct delivery and when they have enough capacity to serve m_i customers. Constraints (1.1f) are domain restrictions. We finally note that one could easily modify the model to account for the inconvenience of rescheduling the delivery, compared to receiving it on the same day, by adding some $c_i^r z_i$ to the objective function. We omit this detail for simplicity of presentation and because customers never formally choose between $u_i = 1$ or $z_i = 1$ in our model.

When $e_i = 0$, the formulation (1.1) models a non-customer keeper $i \in \mathcal{M}$ and it reduces to:

$$H_i(\mathfrak{c}, m_i) \triangleq \min_{w_i} \quad \left[c^k + (c^s - \mathfrak{c})m_i\right] w_i \tag{1.2a}$$

$$\text{s.t.} \quad m_i w_i \leq b_i \tag{1.2b}$$

$$w_i \in \{0,1\}. \tag{1.2c}$$

Hence, when $e_i = 0$, i is a non-customer keeper and is willing to provide crowdkeeping service only if the total to-be-earned compensation $\mathfrak{c}m_i$ is higher than the inconvenience cost $c^k + c^s m_i$ and if the number of to-be-served customers m_i is lower than their capacity b_i.

Finally, it is worth observing that in the case that $m_i \leq b_i$, both models (1.1) and (1.2) can be reformulated as linear programs, as described in the following proposition.

Proposition 1 *Models* (1.1) *and* (1.2) *with relaxed integrality requirements always have an optimal solution in which all variables assume binary values when* $m_i \leq b_i$*, for all* $i \in \mathscr{N} \cup \mathscr{M}$.

In other words, the constraint matrices of models (1.1) and (1.2) are totally unimodular. The proof is presented in Appendix B, Section B.1.

1.4.2 Platform Model

The platform attracts the crowd on the supply side to provide storage services by offering them compensation, and it encourages customers on the demand side to use storage services by offering them convenience and a lower fee. The platform matches supply and demand in the first stage by pricing the compensation $\mathfrak{c}$ and the pickup fee f^p, and by showing the maximum number of to-be-served customers m to keepers and those potential keeper locations v to customers; and in the third stage by assigning inactive customers choosing the pickup option to available keepers using $\hat{v}$ variables, and by determining a vehicle route using x variables for visiting all active customers and keepers identified by y. The platform model is:

$$H_P \triangleq \max_{f^p,\mathfrak{c},m,v} \quad \sum_{i\in\mathscr{N}} \Big[f^d(u_i+z_i) + f^p(e_i-u_i-z_i) \Big] - \mathfrak{c} \sum_{j\in\mathscr{M}\cup\mathscr{N}} m_j - h(m,u,v,w,z) \tag{1.3a}$$

$$\text{s.t.} \quad v_{ij} \leq r_{ij}, \forall i \in \mathscr{N}, \forall j \in \mathscr{M}\cup\mathscr{N} \tag{1.3b}$$

$$f^p \in [0,\bar{f}^p], \mathfrak{c} \in [0,\bar{\mathfrak{c}}], m_j \in [0,\bar{m}], v_{ij} \in \{0,1\}, \quad \forall i \in \mathscr{N}, \forall j \in \mathscr{M}\cup\mathscr{N}, \tag{1.3c}$$

with the third-stage model

$$h(m,u,v,w,z) \triangleq \min_{\hat{v},x,y,\hat{z}} \quad c^d \sum_{i\in\mathscr{M}\cup\mathscr{N}} \sum_{j\in\mathscr{M}\cup\mathscr{N}} t_{ij}x_{ij} + \sum_{i\in\mathscr{N}} (c^r z_i + \hat{c}^r \hat{z}_i) \tag{1.4a}$$

$$\text{s.t.} \quad \sum_{i\in\mathscr{N}} \hat{v}_{ij} \leq m_j, \forall j \in \mathscr{M}\cup\mathscr{N} \tag{1.4b}$$

$$\hat{v}_{ij} \leq w_j, \forall i \in \mathscr{N}, \forall j \in \mathscr{M}\cup\mathscr{N} \tag{1.4c}$$

$$\hat{v}_{ij} \leq v_{ij}, \forall i \in \mathscr{N}, \forall j \in \mathscr{M} \cup \mathscr{N} \tag{1.4d}$$

$$\sum_{i \in \mathscr{N}} \hat{v}_{ij} \leq \bar{m} y_j, \forall j \in \mathscr{M} \cup \mathscr{N} \tag{1.4e}$$

$$y_i \geq u_i, \forall i \in \mathscr{N} \tag{1.4f}$$

$$\sum_{j \in \mathscr{M} \cup \mathscr{N}} \hat{v}_{ij} + u_i + z_i + \hat{z}_i = 1, \forall i \in \mathscr{N} \tag{1.4g}$$

$$\sum_{i \in \mathscr{M} \cup \mathscr{N}} x_{ij} = y_j, \forall j \in \mathscr{M} \cup \mathscr{N} \tag{1.4h}$$

$$\sum_{i \in \mathscr{M} \cup \mathscr{N}} x_{ji} = y_j, \forall j \in \mathscr{M} \cup \mathscr{N} \tag{1.4i}$$

$$\sum_{i,j \in \mathscr{S}} x_{ij} \leq |\mathscr{S}| - 1, \quad \forall \mathscr{S} \subset \mathscr{M} \cup \mathscr{N}, 2 \leq |\mathscr{S}| \leq |\mathscr{M} \cup \mathscr{N}| - 2 \tag{1.4j}$$

$$\hat{v}_{ij}, x_{i'j}, y_j, \hat{z}_i \in \{0,1\}, \quad \forall i \in \mathscr{N}, \forall i' \in \mathscr{M} \cup \mathscr{N}, \forall j \in \mathscr{M} \cup \mathscr{N}. \tag{1.4k}$$

Model H_P is the optimization model of the platform. The objective function (1.3a) maximizes the platform's profit. The first term is the revenue generated from the delivery and pickup fees paid by customers, the second term is the compensation paid to keepers, and the third term is the cost of making deliveries and rescheduling deliveries. Constraints (1.3b) ensure that the platform only pre-assigns customers to those keepers located in the same zone. Constraints (1.3c) are domain restrictions for the first-stage decisions, where $\bar{f}^p, \bar{\mathfrak{c}}, \bar{m}$ are the upper bounds of $f^p, \mathfrak{c}, m$, respectively.

The third-stage model $h(m,u,v,w,z)$ assigns active keepers to serve inactive customers, builds a vehicle tour for visiting all active nodes, and reschedules other deliveries. Objective function (1.4a) is the minimization of the total cost for visiting all active nodes and for rescheduling deliveries, including the active ones rescheduled by customers themselves ($z=1$) and the passive ones rescheduled by the platform due to the lack of available keepers ($\hat{z}=1$). Constraints (1.4b)$-$ (1.4d) state that the platform can only assign customers to one of those available keepers that are accepted by customers and have enough capacity. Specifically, the platform ensures that active keeper j does not receive more deliveries than the promised maximum number m_j, and allocates customers to pick up from one

of their pre-assigned keepers. Constraints (1.4e) and (1.4f) require that active nodes are visited. Constraints (1.4g) ensure that if customers prefer the pickup option but cannot be served by any keeper, their delivery along with those deliveries of absent customers will be rescheduled. Constraints (1.4h) and (1.4i) are degree constraints, (1.4j) are sub-tour elimination constraints (SECs), and (1.4k) are domain restrictions for the third-stage decisions.

1.4.3 Bilevel Program with Multiple Followers

Considering the platform as the leader and customers and keepers as followers, the bilevel program (BP) for the Crowdkeeping Delivery Problem is presented as follows:

$$
\begin{aligned}
(\mathrm{BP}) \max_{\substack{f^p,\mathfrak{c},m,v,\\ u,w,z,\hat{v},x,y,\hat{z}}} \quad & \sum_{i\in\mathscr{N}} \Big[f^d(u_i+z_i) + f^p(e_i-u_i-z_i) \Big] - \mathfrak{c} \sum_{j\in\mathscr{M}\cup\mathscr{N}} m_j \\
& -c^d \sum_{i\in\mathscr{M}\cup\mathscr{N}} \sum_{j\in\mathscr{M}\cup\mathscr{N}} t_{ij}x_{ij} - \sum_{i\in\mathscr{N}} (c^r z_i + \hat{c}^r \hat{z}_i) \\
\text{s.t.} \quad & (1.3\mathrm{b}) - (1.3\mathrm{c}), (1.4\mathrm{b}) - (1.4\mathrm{k}) \\
& \langle u_i, w_i, z_i \rangle \in \arg\min H_i(f^p, \mathfrak{c}, m_i, v_{i:}), \forall i \in \mathscr{M}\cup\mathscr{N}, \qquad (1.5)
\end{aligned}
$$

where constraints (1.5) indicate that $\langle u_i, w_i, z_i \rangle$ are the best responses of customer or keeper i.

The CDP is a generalization of the Traveling Salesman Problem (TSP) and therefore is NP-hard (Dantzig, Fulkerson, and Johnson 1954). The CDP is closely related to the Covering Tour problem (CTP), since customers who choose delivery and keepers who serve customers are nodes to be visited in the optimal tour, and customers who choose to pick up are nodes to be covered. Compared to CTP, the main difference in CDP is that the platform, keepers and customers optimize their respective objective functions, and that the decisions are decentralized to three stages to be in accordance with the implementation. In addition to the covering and routing decisions, we also consider the pricing decisions. Note that, in this study, customers are assumed to pick up their parcels from the keeper, which is in line with the current practice. Nevertheless, the keeper could also

do the deliveries to customers. A model that accommodates such a feature is presented in Appendix 1.8.

1.5 Solution Procedure

In order to solve the BP in Section 1.4.3, we first reformulate it into an equivalent single-level model using the strong duality theorem and then solve the model exactly using a row generation algorithm. We then derive the exact best response sets of followers and develop an approximation of the optimal travel time to improve the efficiency of the solution procedure.

1.5.1 Reformulation as a Single-level Program

Due to Proposition 1, the integrality requirement of the variables u_i, w_i, z_i can be relaxed into $\{u_i, w_i, z_i \geq 0\}$. The upper bounds $\{u_i, w_i, z_i \leq 1\}$ can be omitted since they are implied by constraints (1.1b) $-$ (1.1d). Due to the fact that models (1.1) and (1.2) are feasible, we can use the strong duality to represent the followers' optimal decisions in the first level of BP. Additionally, $m_i \leq b_i$ can be added to the BP without affecting the optimal solution (see proof in Appendix 3.7), which in turn makes constraint (1.1e) redundant. Let ν, ϕ, and ψ be the dual variables corresponding to constraints (1.1b)$-$(1.1d), respectively, and let λ be the dual variables corresponding to constraints $w \leq 1$, we then reformulate the BP into an equivalent single-level program (SP_1):

$$
\begin{aligned}
(\text{SP}_1) \max_{\substack{f^p, \mathfrak{c}, m, v, u, w, z, \\ \hat{v}, x, y, \hat{z}, \nu, \phi, \psi, \lambda}} \quad & \sum_{i \in \mathscr{N}} \Big[f^d(u_i + z_i) + f^p(e_i - u_i - z_i) \Big] - \mathfrak{c} \sum_{j \in \mathscr{M} \cup \mathscr{N}} m_j \\
& - c^d \sum_{i \in \mathscr{M} \cup \mathscr{N}} \sum_{j \in \mathscr{M} \cup \mathscr{N}} t_{ij} x_{ij} - \sum_{i \in \mathscr{N}} (c^r z_i + \hat{c}^r \hat{z}_i) && \text{(1.6a)} \\
\text{s.t.} \quad & [(1.1\text{b}) - (1.1\text{d}), \forall i \in \mathscr{M} \cup \mathscr{N}], (1.3\text{b}) - (1.3\text{c}), (1.4\text{b}) - (1.4\text{k}) \\
& (f^d - f^p)(u_i + z_i) + [c^k + (c^s - \mathfrak{c}) m_i] w_i - c^p \left(\max_{j \in \mathscr{M} \cup \mathscr{N}} t_{ij} v_{ij} \right) (u_i + z_i) \\
& \qquad \leq e_i (a_i - 1) \nu_i - e_i a_i \phi_i + (e_i - 1) \lambda_i, \forall i \in \mathscr{M} \cup \mathscr{N} && \text{(1.6b)} \\
& - e_i \psi_i + (e_i - 1) \lambda_i \leq c^k + (c^s - \mathfrak{c}) m_i, \forall i \in \mathscr{M} \cup \mathscr{N} && \text{(1.6c)}
\end{aligned}
$$

$$-\nu_i + \psi_i \leq f^d - f^p - c^p \max_{j \in \mathscr{M} \cup \mathscr{N}} t_{ij} v_{ij}, \forall i \in \mathscr{N} \tag{1.6d}$$

$$-\phi_i \leq f^d - f^p - c^p \max_{j \in \mathscr{M} \cup \mathscr{N}} t_{ij} v_{ij}, \forall i \in \mathscr{N} \tag{1.6e}$$

$$u_i, w_j, z_i, \nu_i, \phi_i, \psi_i, \lambda_{j'} \geq 0, m_j \in [0, b_j],$$
$$\forall i \in \mathscr{N}, \forall j \in \mathscr{M} \cup \mathscr{N}, \forall j' \in \mathscr{M}, \tag{1.6f}$$

where constraints (1.3b)−(1.3c) and (1.4b)−(1.4k) are constraints of the platform model, constraints (1.1b)−(1.1d) ensure the primal feasibility and (1.6c)−(1.6e) ensure the dual feasibility of the customer model, constraints (1.6b) guarantee the optimality of the customer model, and (1.6f) are domain restrictions.

Due to the bilinear and nonlinear terms $f^p u_i$, $f^p z_i$, $\mathfrak{c} m_j$, $(c^s - \mathfrak{c}) m_i w_i$, and $\max_{j \in \mathscr{M} \cup \mathscr{N}} t_{ij} v_{ij}$, the SP_1 is a mixed-integer nonlinear program. Therefore, we first define the auxiliary variable γ_i to model $\max_{j \in \mathscr{M} \cup \mathscr{N}} t_{ij} v_{ij}$,(i.e., $\gamma_i \geq t_{ij} v_{ij}$, for all $j \in \mathscr{M} \cup \mathscr{N}$), and then linearize the bilinear terms by letting $\tau_j = m_j w_j$, $\theta_j = \mathfrak{c} m_j$, $\rho_j = \theta_j w_j$, $\rho_{1i} = f^p u_i$, $\rho_{2i} = f^p z_i$, $\rho_{3i} = \gamma_i u_i$, $\rho_{4i} = \gamma_i z_i$, and by adding auxiliary linear constraints (1.7f) − (1.7v). This way, the SP_1 is reformulated into a linear single-level program (LSP_1):

$$(\mathrm{LSP}_1) \max_{\substack{f^p, \mathfrak{c}, m, v, u, w, z, \hat{v}, x, y, \hat{z}, \\ \nu, \phi, \psi, \lambda, \gamma, \tau, \theta, \rho}} \quad \sum_{i \in \mathscr{N}} \Big[f^d (u_i + z_i) + (f^p e_i - \rho_{1i} - \rho_{2i}) \Big] - \sum_{j \in \mathscr{M} \cup \mathscr{N}} \theta_j$$
$$- c^d \sum_{i \in \mathscr{M} \cup \mathscr{N}} \sum_{j \in \mathscr{M} \cup \mathscr{N}} t_{ij} x_{ij} - \sum_{i \in \mathscr{N}} (c^r z_i + \hat{c}^r \hat{z}_i) \tag{1.7a}$$

$$\text{s.t.} \quad [(1.1\text{b}) - (1.1\text{d}), \forall i \in \mathscr{M} \cup \mathscr{N}],$$
$$(1.3\text{b}) - (1.3\text{c}), (1.4\text{b}) - (1.4\text{k}), (1.6\text{f})$$
$$f^d (u_i + z_i) - \rho_{1i} - \rho_{2i} + c^k w_i + c^s \tau_i - \rho_i - c^p (\rho_{3i} + \rho_{4i})$$
$$\leq e_i (a_i - 1) \nu_i - e_i a_i \phi_i + (e_i - 1) \lambda_i, \forall i \in \mathscr{M} \cup \mathscr{N} \tag{1.7b}$$

$$-e_i \psi_i + (e_i - 1) \lambda_i \leq c^k + c^s m_i - \theta_i, \forall i \in \mathscr{M} \cup \mathscr{N} \tag{1.7c}$$

$$-\nu_i + \psi_i \leq f^d - f^p - c^p \gamma_i, \forall i \in \mathscr{N} \tag{1.7d}$$

$$-\phi_i \leq f^d - f^p - c^p \gamma_i, \forall i \in \mathscr{N} \tag{1.7e}$$

$$\gamma_i \geq t_{ij} v_{ij}, \forall i \in \mathscr{N}, \forall j \in \mathscr{M} \cup \mathscr{N} \tag{1.7f}$$

$$0 \leq \tau_j \leq m_j, \quad \forall j \in \mathscr{M} \cup \mathscr{N} \tag{1.7g}$$

$$m_j - M_1 (1 - w_j) \leq \tau_j \leq M_1 w_j, \quad \forall j \in \mathscr{M} \cup \mathscr{N} \tag{1.7h}$$

$$\theta_j \geq 0, \quad \forall j \in \mathscr{M} \cup \mathscr{N} \tag{1.7i}$$

$$\theta_j \geq \bar{\mathfrak{c}} m_j + \mathfrak{c}\bar{m} - \bar{c}\bar{m}, \quad \forall j \in \mathscr{M} \cup \mathscr{N} \tag{1.7j}$$

$$\theta_j \leq \bar{\mathfrak{c}} m_j, \quad \forall j \in \mathscr{M} \cup \mathscr{N} \tag{1.7k}$$

$$\theta_j \leq \mathfrak{c}\bar{m}, \quad \forall j \in \mathscr{M} \cup \mathscr{N} \tag{1.7l}$$

$$0 \leq \rho_j \leq \theta_j, \quad \forall j \in \mathscr{M} \cup \mathscr{N} \tag{1.7m}$$

$$\theta_j - M_2(1-w_j) \leq \rho_j \leq M_2 w_j, \quad \forall j \in \mathscr{M} \cup \mathscr{N} \tag{1.7n}$$

$$0 \leq \rho_{1i} \leq f^p, \quad \forall i \in \mathscr{N} \tag{1.7o}$$

$$f^p - M_3(1-u_i) \leq \rho_{1i} \leq M_3 u_i, \quad \forall i \in \mathscr{N} \tag{1.7p}$$

$$0 \leq \rho_{2i} \leq f^p, \quad \forall i \in \mathscr{N} \tag{1.7q}$$

$$f^p - M_3(1-z_i) \leq \rho_{2i} \leq M_3 z_i, \quad \forall i \in \mathscr{N} \tag{1.7r}$$

$$0 \leq \rho_{3i} \leq \gamma_i, \quad \forall i \in \mathscr{N} \tag{1.7s}$$

$$\gamma_i - M_4(1-u_i) \leq \rho_{3i} \leq M_4 u_i, \quad \forall i \in \mathscr{N} \tag{1.7t}$$

$$0 \leq \rho_{4i} \leq \gamma_i, \quad \forall i \in \mathscr{N} \tag{1.7u}$$

$$\gamma_i - M_4(1-z_i) \leq \rho_{4i} \leq M_4 z_i, \quad \forall i \in \mathscr{N}. \tag{1.7v}$$

The program LSP_1 is an equivalent reformulation of SP_1 that takes the form of a single-level mixed-integer linear program (MILP), where M_1, M_2, M_3, M_4 can be set to $\bar{m}, \bar{c}\bar{m}, \bar{f}^p, \max t_{ij}$, respectively.

Remark 1 *SP_1 is a non-linear program with $O((|\mathscr{M}|+|\mathscr{N}|)^2)$ variables and $O(2^{|\mathscr{M}|+|\mathscr{N}|})$ constraints. LSP_1 is a MILP with $O((|\mathscr{M}|+|\mathscr{N}|)^2)$ variables and $O(2^{|\mathscr{M}|+|\mathscr{N}|})$ constraints containing big M values. To be exact, the number of variables in SP_1 is $2+5|\mathscr{N}|+4(|\mathscr{M}|+|\mathscr{N}|)+2|\mathscr{N}|(|\mathscr{M}|+|\mathscr{N}|)+(|\mathscr{M}|+|\mathscr{N}|)^2$, and the number of constraints is $4+11|\mathscr{N}|+10(|\mathscr{M}|+|\mathscr{N}|)+3|\mathscr{N}|(|\mathscr{M}|+|\mathscr{N}|)+(|\mathscr{M}|+|\mathscr{N}|)^2$. To linearize SP_1 into LSP_1, an additional $5|\mathscr{N}|+3|\mathscr{M}|$ variables and $8|\mathscr{N}|+8(|\mathscr{M}|+|\mathscr{N}|)+|\mathscr{N}|(|\mathscr{M}|+|\mathscr{N}|)$ constraints are added.*

We then use the row generation method to solve the linear single-level model LSP_1 with exponentially many SECs (1.4j) to optimality by separating at all integer solutions

(Padberg and Rinaldi 1991). Specifically, we first solve the relaxed LSP_1 model (LSP_1^R), which is obtained by removing the SECs. Let h^* be an optimal solution of the LSP_1^R. We then identify a member of the SECs that is violated by h^* by searching for a subset $\mathscr{S} \subset \mathscr{M} \cup \mathscr{N}$ with $\sum_{i,j\in\mathscr{S}} x_{ij}^* > |\mathscr{S}| - 1$ and $2 \leq |\mathscr{S}| \leq |\mathscr{M} \cup \mathscr{N}| - 2$. If such a subset $\mathscr{S}$ exists, then constraint $\sum_{i,j\in\mathscr{S}} x_{ij} \leq |\mathscr{S}| - 1$ is violated. It is then added to the LSP_1^R using a lazy constraint callback routine, and optimization is resumed. This process is repeated until all the SECs are satisfied by h^*.

1.5.2 Customer Best Response Set

We now derive a linear program representation of each customer's and keeper's optimal solution set (also referred as "best response set") in order to further improve the solution efficiency of BP. Since each customer has finite number of choices, it is possible to enumerate the objective values achieved by all possible choices to confirm that a response is indeed best.

Proposition 2 *Given $i \in \mathscr{M} \cup \mathscr{N}$, f^p, $\mathfrak{c}$, and m_i, a solution (u_i, w_i, z_i) is optimal for model (1.1) if and only if it satisfies constraints* (1.1b)-(1.1e) *and there exists $\eta_i \in \mathfrak{R}$ such that:*

$$f^d(u_i + z_i) + \left[c^k + (c^s - \mathfrak{c})m_i\right] w_i + (f^p + c^p \max_{j\in\mathscr{M}\cup\mathscr{N}} t_{ij}v_{ij})(e_i - u_i - z_i) \leq \eta_i \tag{1.8a}$$

$$\eta_i \leq \begin{cases} f^d + (1 - a_i)\left[c^k + (c^s - \mathfrak{c})m_i\right] & \text{if } i \in \mathscr{N} \\ c^k + (c^s - \mathfrak{c})m_i & \text{otherwise} \end{cases} \tag{1.8b}$$

$$\eta_i \leq \begin{cases} f^d & \text{if } i \in \mathscr{N} \\ 0 & \text{otherwise} \end{cases} \tag{1.8c}$$

$$\eta_i \leq \begin{cases} f^p + c^p \max_{j\in\mathscr{M}\cup\mathscr{N}} t_{ij}v_{ij} & \text{if } i \in \mathscr{N} \\ 0 & \text{otherwise.} \end{cases} \tag{1.8d}$$

The proof of Proposition 2 is presented in Appendix 3.7. Using Proposition 2, we reformulate the BP into an equivalent single-level program (SP_2):

$$
\begin{aligned}
(\mathrm{SP}_2) \max_{\substack{f^p,\mathfrak{c},m,v,u,w,z,\\ \hat{v},x,y,\hat{z},\eta}} \quad & \sum_{i\in\mathscr{N}}\Big[f^d(u_i+z_i)+f^p(e_i-u_i-z_i)\Big]-\mathfrak{c}\sum_{j\in\mathscr{M}\cup\mathscr{N}} m_j \\
& -c^d\sum_{i\in\mathscr{M}\cup\mathscr{N}}\sum_{j\in\mathscr{M}\cup\mathscr{N}} t_{ij}x_{ij}-\sum_{i\in\mathscr{N}}(c^r z_i+\hat{c}^r\hat{z}_i) \\
\text{s.t.} \quad & [(1.1\mathrm{b})-(1.1\mathrm{f}),\forall i\in\mathscr{M}\cup\mathscr{N}],(1.3\mathrm{b})-(1.3\mathrm{c}),(1.4\mathrm{b})-(1.4\mathrm{k}), \\
& (1.8a)-(1.8d), \\
& \eta_i\in\Re,\forall i\in\mathscr{M}\cup\mathscr{N}.
\end{aligned}
$$

Similarly, we linearize the bilinear and nonlinear terms to derive a linear single-level program LSP_2 by letting $\gamma_i=\max_{j\in\mathscr{M}\cup\mathscr{N}} t_{ij}v_{ij}$, $\tau_j=m_jw_j$, $\theta_j=\mathfrak{c}m_j$, $\rho_j=\theta_jw_j$, $\rho_{1i}=f^pu_i$, $\rho_{2i}=f^pz_i$, $\rho_{3i}=\gamma_iu_i$, $\rho_{4i}=\gamma_iz_i$, and adding linear constraints (1.7f) $-$ (1.7v).

$$
\begin{aligned}
(\mathrm{LSP}_2) \max_{\substack{f^p,\mathfrak{c},m,v,u,w,z,\\ \hat{v},x,y,\hat{z},\eta,\gamma,\tau,\theta,\rho}} \quad & \sum_{i\in\mathscr{N}}\Big[f^d(u_i+z_i)+(f^pe_i-\rho_{1i}-\rho_{2i})\Big]-\sum_{j\in\mathscr{M}\cup\mathscr{N}}\theta_j \\
& -c^d\sum_{i\in\mathscr{M}\cup\mathscr{N}}\sum_{j\in\mathscr{M}\cup\mathscr{N}} t_{ij}x_{ij}-\sum_{i\in\mathscr{N}}(c^r z_i+\hat{c}^r\hat{z}_i) && (1.9\mathrm{a}) \\
\text{s.t.} \quad & [(1.1\mathrm{b})-(1.1\mathrm{f}),\forall i\in\mathscr{M}\cup\mathscr{N}],(1.3\mathrm{b})-(1.3\mathrm{c}), \\
& (1.4\mathrm{b})-(1.4\mathrm{k}),(1.7\mathrm{f})-(1.7\mathrm{v}) \\
& f^d(u_i+z_i)+c^kw_i+c^s\tau_i-\rho_i+(f^p+c^p\gamma_i)e_i \\
& -\rho_{1i}-\rho_{2i}-c^p(\rho_{3i}+\rho_{4i})\le\eta_i && (1.9\mathrm{b}) \\
& \eta_i\le\begin{cases} f^d+(1-a_i)\left(c^k+c^sm_i-\theta_i\right) & \text{if } i\in\mathscr{N} \\ c^k+c^sm_i-\theta_i & \text{otherwise}\end{cases} && (1.9\mathrm{c}) \\
& \eta_i\le\begin{cases} f^d & \text{if } i\in\mathscr{N} \\ 0 & \text{otherwise}\end{cases} && (1.9\mathrm{d}) \\
& \eta_i\le\begin{cases} f^p+c^p\gamma_i & \text{if } i\in\mathscr{N} \\ 0 & \text{otherwise}\end{cases} && (1.9\mathrm{e}) \\
& \eta_i\in\Re,\forall i\in\mathscr{M}\cup\mathscr{N}. && (1.9\mathrm{f})
\end{aligned}
$$

Remark 2 *SP_2 is a non-linear program, while LSP_2 is a MILP, both with $O((|\mathscr{M}|+|\mathscr{N}|)^2)$ variables and $O(2^{|\mathscr{M}|+|\mathscr{N}|})$ constraints. Compared to SP_1, SP_2 has $2|\mathscr{N}|$ fewer variables and $6|\mathscr{N}|$ fewer constraints. To linearize SP_2 into LSP_2, an additional $5|\mathscr{N}|+3|\mathscr{M}|$ variables and $8|\mathscr{N}|+8(|\mathscr{M}|+|\mathscr{N}|)+|\mathscr{N}|(|\mathscr{M}|+|\mathscr{N}|)$ constraints are added. Consequently, LSP_2 also features $2|\mathscr{N}|$ fewer variables and $6|\mathscr{N}|$ fewer constraints compared to LSP_1.*

Formulation SP_2 has the potential of being more efficient than formulation SP_1 because the search space of the former is smaller than that of the latter. Specifically, the difference between SP_1 and SP_2 is that the feasible solutions of (u_i, w_i, z_i) in the former formulation are partially constructed by the constraints (1.6b)−(1.6f), while those in the latter one are characterized by the constraints (1.8a)− (1.8d). By comparing these two sets of constraints, we note that SP_1 has $3|\mathscr{N}|+|\mathscr{M}|$ more dual variables that need to be considered in order to find the optimal value of the objective function. In contrast, SP_2 directly sets the upper bound of the objective function with $|\mathscr{N}|+|\mathscr{M}|$ new variables without the need to search for the optimal values of dual variables. Therefore, its search space is smaller, yielding a more efficient formulation. For more details, please refer to Appendix 1.8. Similar results also applied to LSP_2 and LSP_1, since they are equivalent linear formulations of SP_2 and SP_1, respectively.

1.5.3 Approximation Model with Estimated Travel Time

Solving the SP_1 model exactly for large instances may be computationally inefficient. The paradigm of crowdkeeping assumes that all participants (i.e. unavailable customers, available customers, and non-customer keepers) are drawn from the same group of people, the so-called crowd. It is therefore reasonable to assume that locations are drawn from the same distribution. Moreover, the decision of being available for crowdkeeping or not is typically made without knowledge of who are the other participants. This suggests that one might approximate the locations of the nodes that need to be visited as independent and identically distributed (i.i.d.) locations drawn from a probability density function f,

which captures the high and low population density areas in this region. Given a probability density function f on a two-dimensional region R, when nodes are i.i.d., Beardwood, Halton, and Hammersley (1959) show in their seminal work that:

$$\lim_{n\to\infty} \frac{\text{TSP}_n^*}{\sqrt{n}} \approx \beta \iint_R \sqrt{f(x,y)}\, dxdy,$$

where TSP_n^* is the optimal travel time, n is the number of nodes, $\iint_R \sqrt{f(x,y)}\, dxdy$ is the integral density of the region R, and β is a constant. If nodes are uniformly and independently scattered, the integral density is equal to the area of the region, A. In this case, $\beta\sqrt{nA}$ is asymptotically a good approximation for the optimal travel time as $n \to \infty$. Considering that our model is meant to serve real-world cases, where the node dispersion is unknown, the integral density cannot be computed. Therefore, we consider $\iint_R \sqrt{f(x,y)}\, dxdy$ as part of the approximation, similar to (Cavdar and Sokol 2015). We then use regression to approximate the term $\beta \iint_R \sqrt{f(x,y)}\, dxdy$ as $\hat{\beta}(n)$ for each instance region, since β also depends on the number of nodes (Franceschetti, Jabali, and Laporte 2017) and $\iint_R \sqrt{f(x,y)}\, dxdy$ depends on the node distribution. That is,

$$\text{TSP}(n) \approx \hat{\beta}(n)\sqrt{n}, \tag{1.10}$$

where $\text{TSP}(n)$ is the approximated optimal travel time of visiting n number of nodes. Both $\text{TSP}(n)$ and $\hat{\beta}(n)$ are functions of n, which is a auxiliary decision variable in our model with $n = \sum_{j \in \mathcal{M} \cup \mathcal{N}} y_j$. To rephrase, we aim not only to find the optimal tour given a certain number of nodes but also to determine the set of nodes to be visited in CDP. Therefore, among sets with the same number of nodes capable of covering the remaining nodes, a set is considered optimal only if it incurs the minimum travel time. To estimate $\hat{\beta}$ as a function of n for each region, we first find the *minimum value* of optimal travel times $\text{TSP}^*(n)$ among all instances with n visited nodes, and then use the pair data $(n, \text{TSP}^*(n))$ as input for model fitting. This approximation (1.10) may yield a more efficient formulation without an exponential number of constraints. We present the approximation model (AM_1) below:

$$
\begin{aligned}
(\mathrm{AM}_1) \max_{\substack{f^p,\mathfrak{c},m,v,u,w,z,\hat{v},y,\hat{z},\\ \nu,\phi,\psi,\gamma,\tau,\theta,\rho,n}} \quad & \sum_{i\in\mathscr{N}}\left[f^d(u_i+z_i)+(f^p e_i-\rho_{1i}-\rho_{2i})\right]-\sum_{j\in\mathscr{M}\cup\mathscr{N}}\theta_j \\
& -c^d\hat{\beta}(n)\sqrt{n}-\sum_{i\in\mathscr{N}}(c^r z_i+\hat{c}^r\hat{z}_i) \\
\text{s.t.} \quad & [(1.1\text{b})-(1.1\text{d}),\forall i\in\mathscr{M}\cup\mathscr{N}],(1.3\text{b})-(1.3\text{c}),(1.4\text{b})-(1.4\text{g}), \\
& (1.6\text{f}),(1.7\text{b})-(1.7\text{v}) \\
& n=\sum_{j\in\mathscr{M}\cup\mathscr{N}} y_j \\
& \hat{v}_{ij},y_j,\hat{z}_i\in\{0,1\},\forall i\in\mathscr{N},\forall j\in\mathscr{M}\cup\mathscr{N},
\end{aligned}
$$

where n represents the number of active nodes to be visited. We elaborate more on the shape of the function $\hat{\beta}(n)$ in Section 1.6.2.

The approximation model (AM_2) with the estimated travel time is:

$$
\begin{aligned}
(\mathrm{AM}_2) \max_{\substack{f^p,\mathfrak{c},m,v,u,w,z,\\ \hat{v},y,\hat{z},\eta,\gamma,\tau,\theta,\rho,n}} \quad & \sum_{i\in\mathscr{N}}\left[f^d(u_i+z_i)+(f^p e_i-\rho_{1i}-\rho_{2i})\right]-\sum_{j\in\mathscr{M}\cup\mathscr{N}}\theta_j \\
& -c^d\hat{\beta}(n)\sqrt{n}-\sum_{i\in\mathscr{N}}(c^r z_i+\hat{c}^r\hat{z}_i) \\
\text{s.t.} \quad & [(1.1\text{b})-(1.1\text{f}),\forall i\in\mathscr{M}\cup\mathscr{N}],(1.3\text{b})-(1.3\text{c}),(1.4\text{b})-(1.4\text{g}), \\
& (1.7\text{f})-(1.7\text{v}),(1.9\text{b})-(1.9\text{f}) \\
& n=\sum_{j\in\mathscr{M}\cup\mathscr{N}} y_j \\
& \hat{v}_{ij},y_j,\hat{z}_i\in\{0,1\},\forall i\in\mathscr{N},\forall j\in\mathscr{M}\cup\mathscr{N}.
\end{aligned}
$$

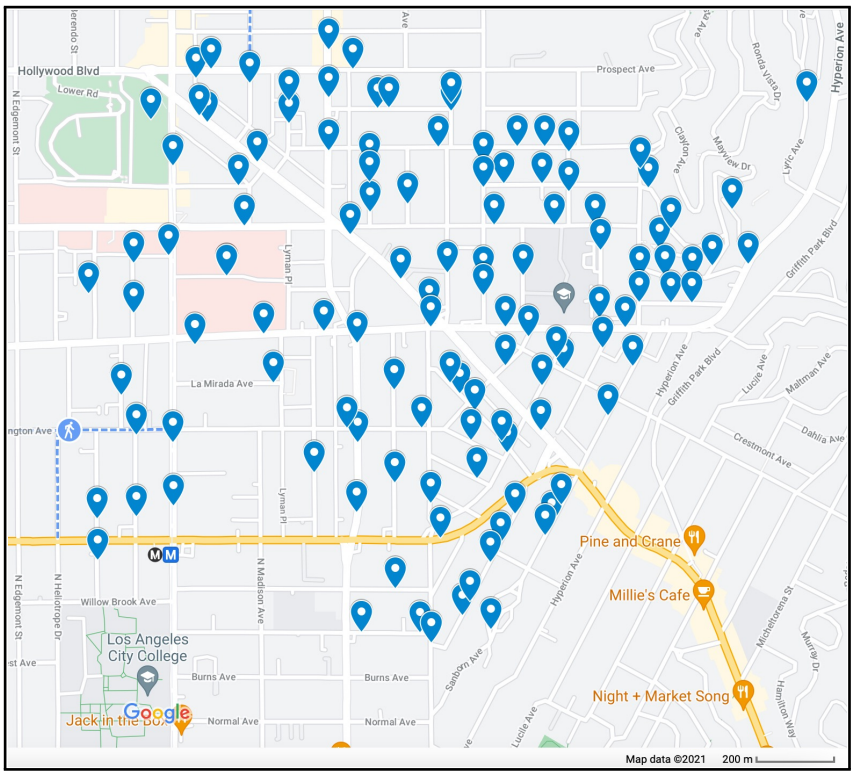
The term $\hat{\beta}(n)\sqrt{n}$ may yield a non-linear program, depending on the form of $\hat{\beta}(n)$. There are several possible approximation functions for $\hat{\beta}(n)$ Franceschetti, Jabali, and Laporte (2017), such as a constant, $\hat{\beta}(n)=\beta_1+\beta_2\frac{1}{n}$, $\hat{\beta}(n)=\beta_1\sqrt{n}+\beta_2\frac{1}{\sqrt{n}}$, or $\hat{\beta}(n)=\beta_1+\beta_2\frac{1}{\sqrt{n}}$. If $\hat{\beta}(n)=\beta_1\sqrt{n}+\beta_2\frac{1}{\sqrt{n}}$ provides the best fit, the term $\sqrt{n}$ is eliminated, resulting in a linear program. In other non-linear cases, we have a finite number of feasible solutions for n since $n=\sum_{j\in\mathscr{M}\cup\mathscr{N}} y_j$ and y_j is binary. Therefore, both AM_1 and AM_2 can be solved to optimality by solving a finite number of linear programs. In the case that n is large, we can use piecewise linear functions to approximate the term $\sqrt{n}$ and obtain a linear program.

1.6 Numerical Study

We now present the implementation details, the experimental settings, the computational performances, and the results.

1.6.1 Dataset and Implementation Details

We use a real-world dataset of vehicle routes that were executed by Amazon delivery trucks between July 19, 2018 and August 26, 2018 (Merchan et al. 2021). These routes are located in densely populated urban areas across the United States. The number of customers ranges between 33 and 238 with an average value of 146. The dataset contains information on customer locations including their latitudes, longitudes, zone IDs, and the travel time between customers. A sample customer set is shown in Figure 1.5.



Notes. The exact coordinates are perturbed by the data providers to anonymize the data. Google Map data ©2021.

Figure 1.5 – The region where a sample set of 118 nodes are assigned to a single vehicle in Los Angeles.

We now describe parameter settings in the benchmark instance and in the experimental design. We use 20 randomly selected instances for numerical studies. In each instance, we randomly draw 90% of points as customer locations who have parcels to be delivered and can be potential crowd keepers. We change this ratio between 10% and 90% to obtain samples with different number of customers. The customer absence ratio measures the percentage of customers who are absent from home among all customers. It is 5% for the benchmark instance and changes between 0 and 100% in the sensitivity analysis. The remaining 10% of the locations in the benchmark instance are taken as non-customer keepers who do not have parcels to be delivered but can store parcels for their neighbors. For each area, the customer set dynamically changes in different time periods (e.g. days). Thus, each instance will be drawn for 20 times to obtain the samples over 20 periods, which are used to evaluate the average performances of crowd keepers and fixed storage. The standard delivery fee f^d is set to \$2, which is high enough to obtain a positive profit. The inconvenience cost of rescheduling deliveries incurred by customers c^r is the same as the delivery fee as if the delivery has not yet taken place. The penalty for rescheduling a delivery $\hat{c}^r$ is set to \$4, and changes between \$0.6 and \$4 in the sensitivity analysis. The capacity of each keeper b is 10 parcels. We take the truck speed to be 4 times the walking speed. Considering the oil prices and the driver wages, the truck delivery cost per minute c^d is set as \$1 in the benchmark instance (implying truck travel costs equal to travel time) and changes between \$0 and \$2 in the sensitivity analysis. We take the customer inconvenience cost per minute of walking c^p as \$0.1 and change it between \$0 and \$2 in the sensitivity analysis. Both the keeper inconvenience cost c^k and the setup cost of fixed storage is taken as \$0.1. This implies that the setup cost of fixed storage is minimal and almost negligible, ensuring a fair comparison with crowd keepers. The marginal inconvenience cost for serving each parcel as a keeper c^s is set to \$0.01 in the benchmark instance and changes between \$0 and \$2 in the sensitivity analysis. Crowd keepers can only serve those customers located in the same zone and within a limited walk time. Zone IDs are given in the dataset, and the maximum walk time is set as 4 minutes in the benchmark instance and changes between 0 and 6 minutes in the sensitivity analysis.

We compare the performances of the “crowdkeeping”, the “fixed-storage", and the “no-storage" systems. For the system, two cases are considered: the high density fixed-storage system, in which the fixed storage has the density as high as the potential crowd keepers ($j \in \mathscr{M} \cup \mathscr{N}$), and the low density fixed-storage system in which the fixed storage has the density as high as the non-customer keepers ($j \in \mathscr{M}$). The main difference among different systems is that in the fixed-storage system, storage locations are always fixed in different periods and a fixed setup cost is incurred regardless of whether the storage is used or not. In the crowdkeeping system, on the other hand, crowd keeper selection decisions can adapt to the changing customer sets in different periods and the fixed cost is incurred only when the keeper is active. The no-storage system represents the case when there is no storage in the system, only delivering to doorsteps is considered, but rescheduling deliveries is possible. Different special cases of our model can solve these systems and the details are presented in Appendix 3.7.

To evaluate the performances of different formulations and different systems, we compare the platform profit (i.e., the optimal value of the platform model), the customer costs (i.e., the optimal values of the follower models), the truck delivery time implying pollution, and the average customer walk time for picking up. We also report the standard delivery fee, and the optimal values of the pickup fee and the compensation, to demonstrate how pricing decisions change in different scenarios. Additionally, the pickup proportion (defined as the percentage of customers who choose the pickup option) and the rescheduled proportion (defined as the percentage of rescheduled deliveries among all deliveries) are reported in order to investigate if keepers consolidate deliveries and eliminate failed deliveries.

We implement our algorithms using Python 3.7 on a computer with one 2 GHz Quad-Core Intel Core i5 processor and 16GB of RAM. We use Gurobi 9.0.2 as the solver, which employs piecewise linear functions to represent the term $\sqrt{n}$. The time limit is set as two hours.

1.6.2 Selection and Calibration of Optimal Tour Length Estimator

We consider various approximation functions for $\hat{\beta}(n)$ to determine the best fit, including a constant, $\hat{\beta}(n) = \beta_1 + \beta_2 \frac{1}{n}$, $\hat{\beta}(n) = \beta_1 \sqrt{n} + \beta_2 \frac{1}{\sqrt{n}}$, or $\hat{\beta}(n) = \beta_1 + \beta_2 \frac{1}{\sqrt{n}}$. In each region, for every n, we repeatedly draw 20 different instances and obtain the optimal travel time $\text{TSP}_k^*(n)$ for each instance k, where $k = \{1, 2, ..., 20\}$. Considering that determining the set of nodes to visit is the key decision within CDP, we cannot precisely know the exact number and locations of customers and keepers in the tour. Consequently, we treat customers and keepers as a unified distribution, drawing random samples from it. For each n, we find the *minimum* $\text{TSP}^*(n)$ among all instances with n visited nodes (i.e., $\min_k \text{TSP}_k^*(n)$), and use the pair data $(n, \text{TSP}^*(n))$ to estimate the continuous approximation formulation by linear regression. The reason for using the minimum value of the in-sample optimal travel time, rather than considering all in-sample values or the average value, is that in the CDP, our objective is not only to find the optimal tour given a certain set of nodes but also to determine the optimal set of nodes to be visited. Specifically, among sets with the same number of nodes capable of covering the remaining nodes, a set is considered optimal only if it incurs the minimum travel time. In other words, the n customers and keepers to be visited are not randomly selected.

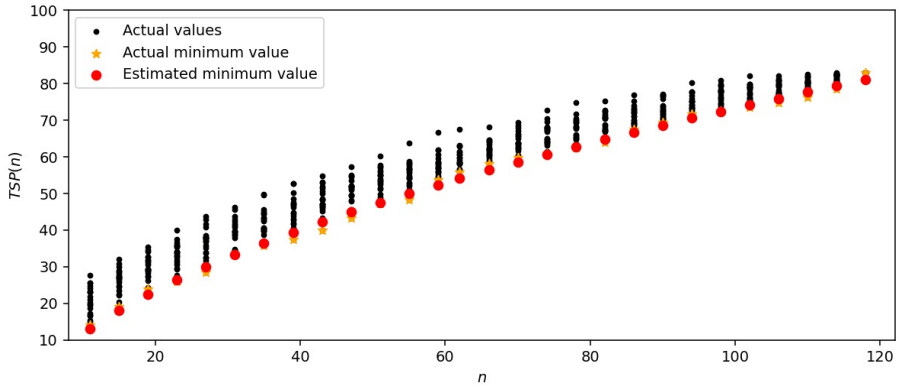


Figure 1.6 – Estimation of the optimal minimum TSP tour duration

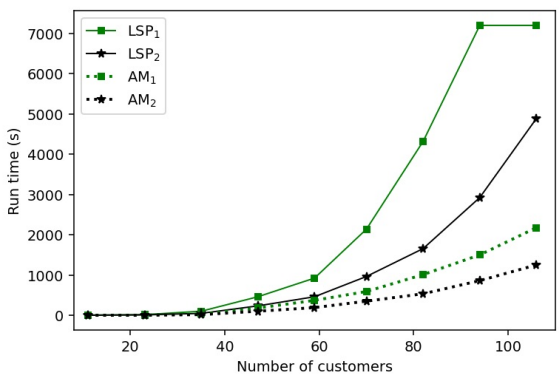
For example, Figure 1.6 is the estimation model of the optimal travel time under the sample region in Figure 1.5, where the total number of nodes is 118. For this instance,

the function $\text{TSP}(n) = 9\sqrt{n} - 17$ for $n \in [11, 118]$ can accurately estimate the minimum travel time of visiting n nodes with the out-of-sample $R^2 = 0.96$. With this approximation for the delivery time, we do not have to run the exact algorithm for each customer group everyday in this instance region, but instead use the approximation model and obtain the approximated solution with high efficiency and accuracy. We evaluate the performance of the optimal tour length estimator under various customer and keeper distributions in the next section and Appendix 1.8.

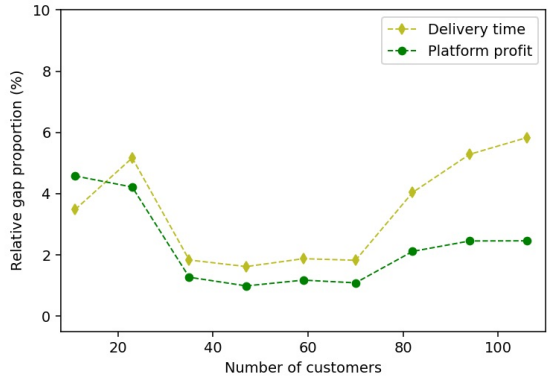
1.6.3 Effectiveness and Efficiency of Solution Procedures

We compare the efficiency and effectiveness of four different formulations: LSP_1, LSP_2, AM_1, and AM_2. The *effectiveness* represents the quality of the solutions in terms of realized costs, while the *efficiency* represents the computing time for obtaining the solutions.

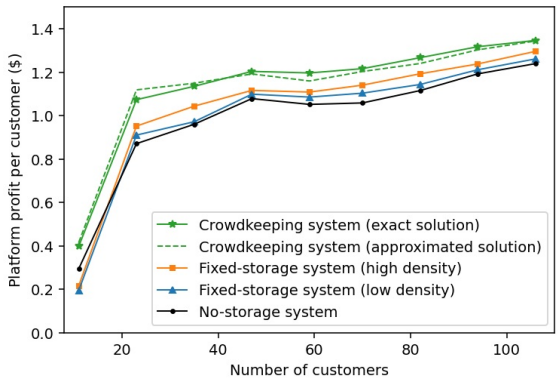
Regarding efficiency, Figure 1.7(a) shows the runtime of LSP_1, LSP_2, AM_1, and AM_2 models for instances with different number of customers. The approximate reformulation with best responses AM_2 yields the best performance with the highest efficiency. Regarding effectiveness, we compare the exact and approximated values. The *exact solution* is the output of the exact model LSP_1 (LSP_2), which are solved to optimality using the row generation algorithm. The *approximated solution* is the realized output by applying the optimal solution of approximation model AM_1 (AM_2) and by visiting the active nodes using the exact optimal tour. Figure 1.7(b) illustrates the change in the relative gap between exact and approximated solutions with varying customer densities. We observe fluctuations in the relative gaps for both platform profit and delivery time, always staying under 6%. The absolute gaps between exact and approximated solutions, shown in Figures 1.7(c)−(d), are also within an acceptable range. The runtime of the approximation model shown in Figure 1.7(a) decreases, on average, by more than 70% compared to that of the exact model. Therefore, both AM_1 and AM_2 have good performances on efficiency and effectiveness, but AM_2 is superior overall as it achieves good accuracy with higher efficiency.



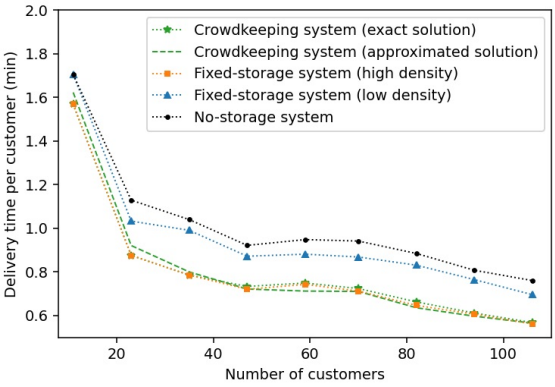
(a) Runtime of exact and approximated formulations



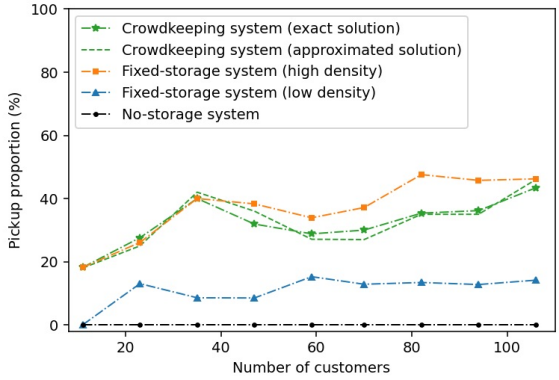
(b) The relative gap proportion



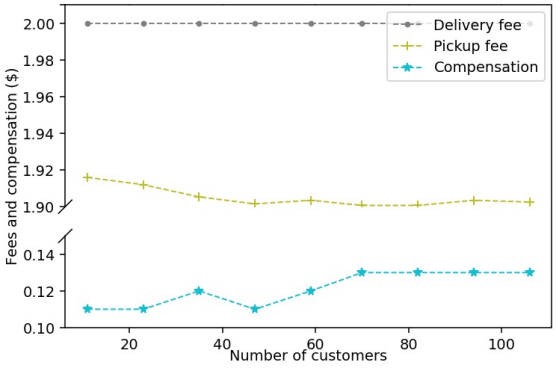
(c) Platform profit per customer



(d) Delivery time per customer



(e) Pickup proportion



(f) Pickup fee and compensation of crowdkeeping

Figure 1.7 – Efficiency and effectiveness of solution procedures under different number of customers

Figure 1.7 also presents detailed results on the platform profit, delivery time, pickup proportion for different systems, and the delivery fee, pickup fee, and compensation for the crowdkeeping system. We find that the crowdkeeping delivery system benefits from economies of scale due to the observation in Figures 1.7(c)−(d) that the platform obtains more profits by serving a larger group of customers with an increasing marginal profit and with a decreasing marginal delivery time for serving one more customer.

As shown in Figures 1.7(c)−(e), the profit of serving one more customer increases in both the fixed-storage and no-storage systems, similar to the crowdkeeping system. A high-density fixed-storage system ensures the highest pickup proportion and the shortest delivery time. However, its higher setup costs reduce marginal profit compared to the crowdkeeping system. Conversely, a low-density fixed-storage system lowers costs but also reduces usage, as fewer locations limit pickup opportunities. This reflects real-world trade-offs: opening more locations increases costs but attracts more pickups, whereas fewer locations lower costs but limit convenience for customers. In contrast, crowdkeeping offers a more flexible balance between cost and utilization. Crowd keepers are activated only when customers choose to pick up from them, allowing the system to flexibly adapt to customer demand and distribution. As a result, the crowdkeeping system achieves a pickup proportion slightly lower than the high-density fixed-storage system but higher than the low-density fixed-storage system, while maintaining the highest marginal profit. Therefore, we conclude that the crowdkeeping system offers the best overall performance, striking an optimal balance between delivery time, pickup proportion, and profitability.

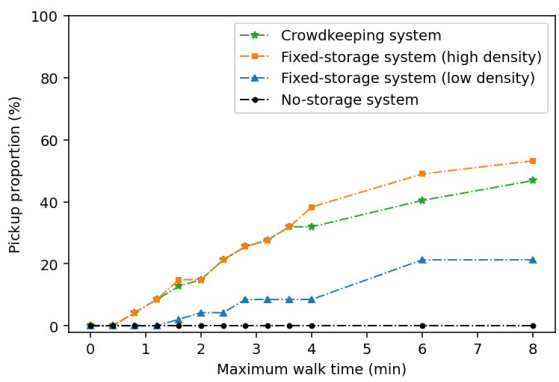
Additionally, as depicted in Figure 1.7(f), the pickup fee offered to customers slightly decreases, while the compensation offered to keepers slightly increases as the crowdkeeping system accommodates more customers, resulting in benefits for both parties.

1.6.4 Sensitivity Analysis

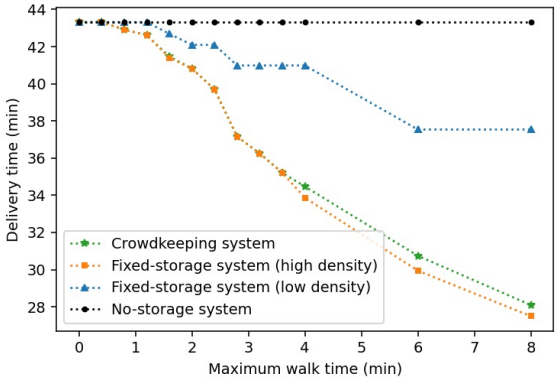
We now investigate the factors that may affect the decisions of participants in the delivery system and lead to different results. These sensitivity analyses are conducted by solving

the model LSP_2 to optimality with the exact solution method.

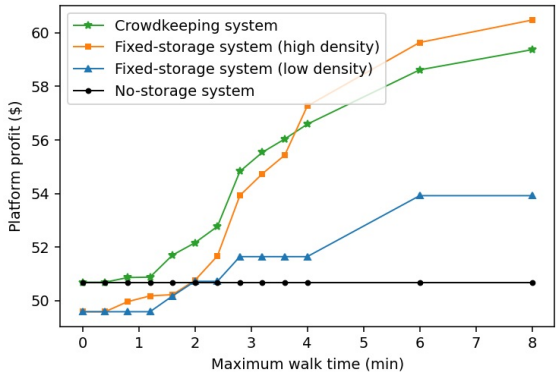
(1) The impact of the service range: The keeper service range is defined as the customer walk range. The larger the maximum walk time that customers can tolerate for picking up, the larger the service range. When the service range is zero, keepers cannot serve any customers, resulting in a scenario where the pickup proportion is zero. However, if keepers can serve customers, the pickup proportion may be higher. Note that this ratio is not necessarily 100%, as, for some customers, the closest pickup option might already be too inconvenient compared to a delivery. Figures 1.8(a)−(d) report the changes in the pickup proportion, the delivery time, the platform profit, and the customer cost, respectively, as the maximum walk time changes.



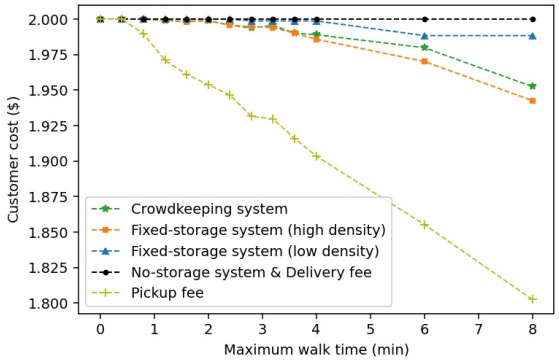
(a) Pickup proportion



(b) Delivery time



(c) Platform profit



(d) Customer cost

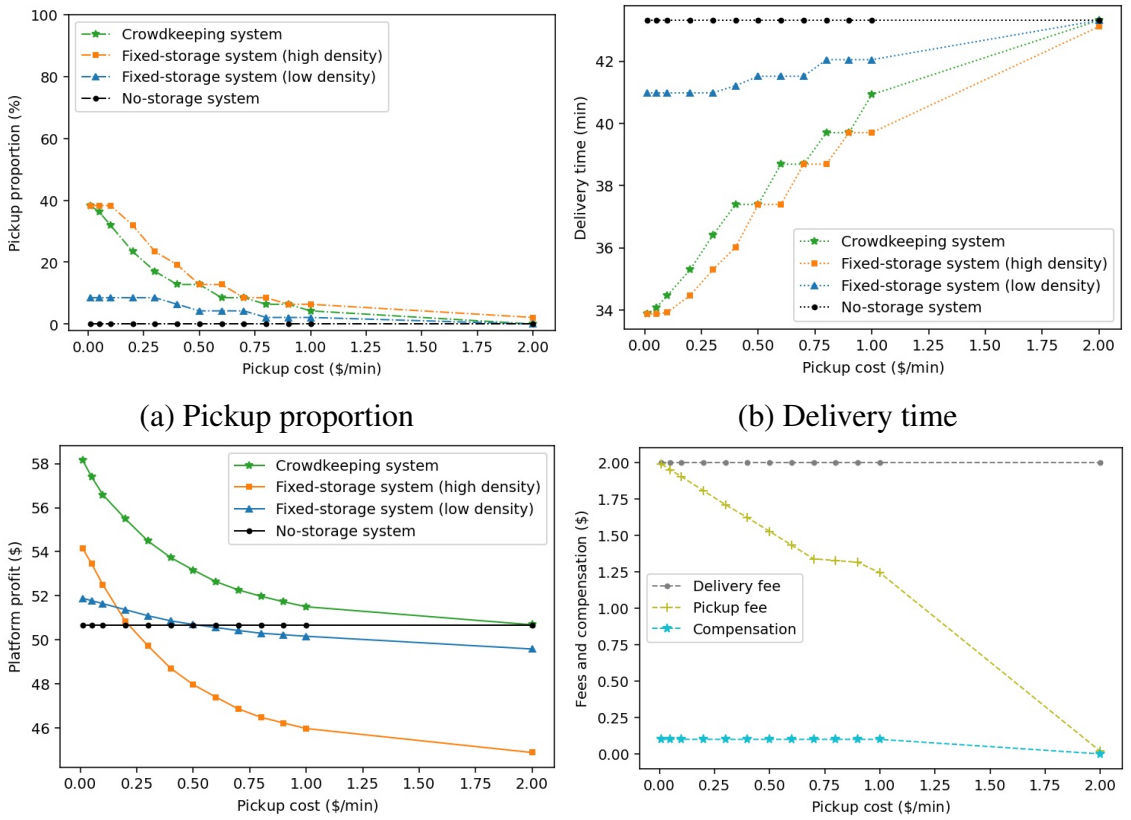
Figure 1.8 – The impact of the service range

We find that when the service range increases, more customers choose the pickup op-

tion (see Figure 1.8(a)), and the platform earns more profits (see Figure 1.8(c)). Moreover, the total delivery time for visiting all active nodes decreases (see Figure 1.8(b)), and this leads to less pollution for the environment (due to less truck utilization). In other words, the delivery system becomes more cost-efficient and environmentally friendly with larger service range. However, if the customer tolerance for walk time is higher than 4 minutes, the crowdkeeping system will be outperformed by the high density fixed-storage system (Figures 1.8(a)−(c)). This is because the availabilities of crowd keepers shrink and stay in short supply when there is a high pickup proportion, and because the marginal cost of serving one more customer increases, whereas the capacities of fixed storage are stable and its marginal cost of serving one more customer decreases. In addition, customer costs are always no higher than the delivery fee (Figure 1.8(d)), since the direct or rescheduled delivery to doorstep is always an alternative for customers and they have the potential to pay less for receiving parcels by choosing the pickup option and to earn compensation by working as keepers. When the service range increases, customers may face a longer walk time and find pickup less efficient than delivery, in which case, the platform has to decrease the pickup fee to make the pickup option more attractive (Figure 1.8(d)).

(2) The impact of the pickup cost: Customers who choose the pickup option need to walk to their appointed keepers, and this creates inconvenience for them. Therefore, in addition to the maximum pickup walk time, the inconvenience cost per minute for picking up (i.e., pickup cost) may also affect customer decisions. Figures 1.9(a)−(c) present how the pickup proportion, the delivery time, and the platform profit change, respectively, as the pickup cost changes, and Figure 1.9(d) shows how fees and compensation change.

We find that the increasing pickup cost makes the delivery option more attractive for more customers and thus there is a tendency for customers to choose the pickup option less often (see Figure 1.9(a)). This tendency reduces the efficiency of the system with an increasing delivery time (see Figure 1.9(b)) and cuts down the benefits of the platform (see Figure 1.9(c)), because both the platform and the delivery system benefit from the consolidation of deliveries. The lower pickup proportion leads to less consolidation, and performance deterioration. Therefore, to discourage more customers from changing their



(a) Pickup proportion

(b) Delivery time

(c) Platform profit

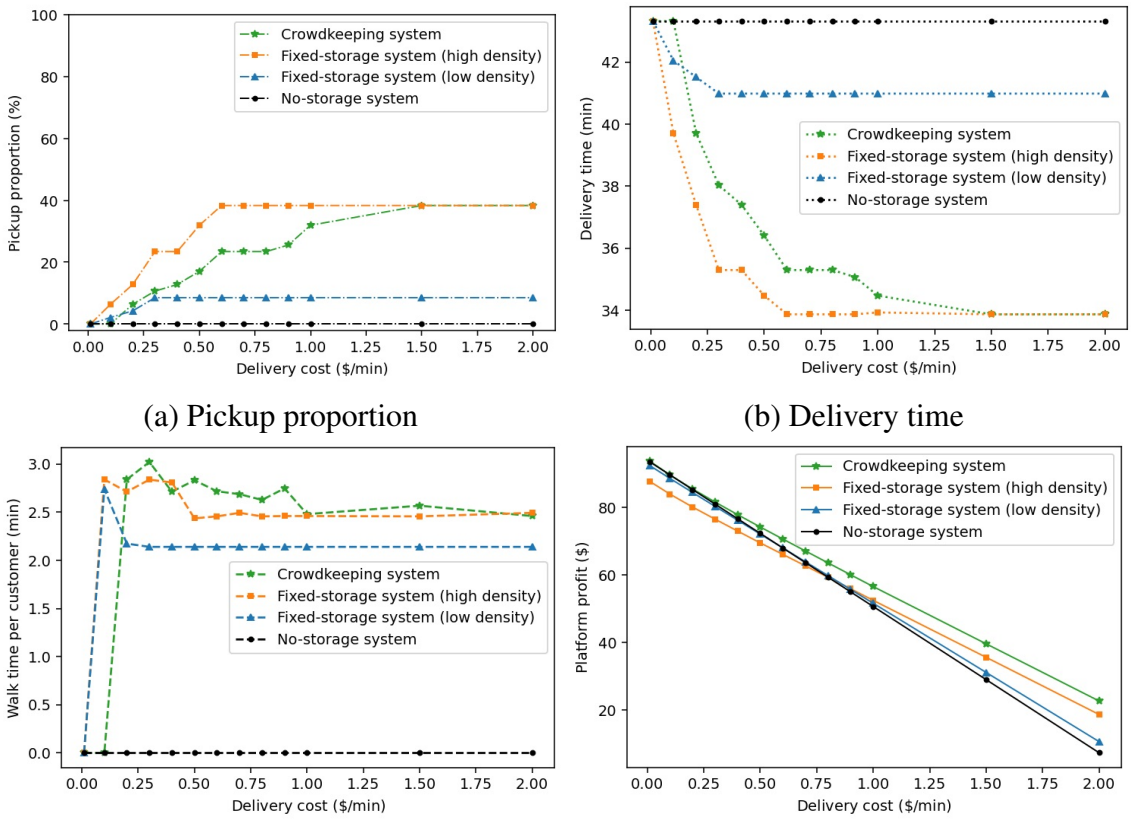
(d) Pickup fee and compensation of crowdkeeping

Figure 1.9 – The impact of the pickup cost

minds and relinquishing the pickup option, the platform must keep lowering the pickup fee to make up for the increasing pickup cost (see Figure 1.9(d)). The higher the pickup cost, the larger the gap between the standard delivery fee and the pickup fee, and the lower the platform's benefits. Although the crowdkeeping system continues to perform better than other systems on profit, the increasing pickup cost narrows the gap between their performances. Therefore, to maintain the high efficiency of the delivery system and guarantee a decent profit for the platform, crowd keepers with more accessible locations should be selected and used to decrease the pickup cost and reduce the inconvenience for customers.

(3) The impact of the delivery cost: Truck delivery costs account for a significant part of the total cost. Thus, the delivery cost per minute of travel time influences the

efficiency of the delivery system. Figures 1.10(a)−(d) present how the pickup proportion, the truck delivery time, the walk time per customer, and the platform profit change, respectively, as the delivery cost changes.



(a) Pickup proportion

(b) Delivery time

(c) Walk time per customer

(d) Platform profit

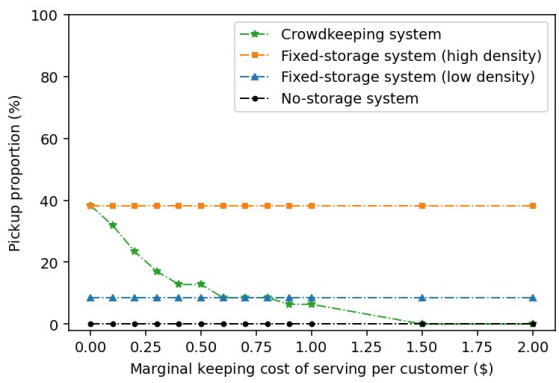
Figure 1.10 – The impact of the delivery cost

Figure 1.10(a) shows that the increasing delivery cost increases the pickup proportion both for the crowd-keeper system and for the fixed-storage system, leading to a decrease in truck delivery time (see Figure 1.10(b)) accompanied with a stable customer walk time (see Figure 1.10(c)). The pickup proportions and the customer walk time stabilize due to the limited keeper service range, and the crowd-keeper system always has a higher pickup proportion than the fixed-storage system due to the higher availability and flexibility. For a fixed delivery fee, the platform profit inevitably decreases when the delivery cost becomes larger, but the crowdkeeping system always yields the best performance in terms of the

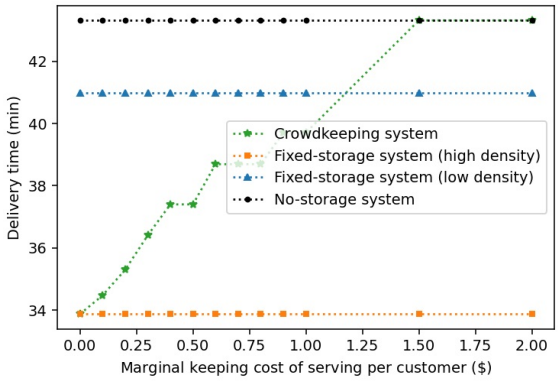
profit, compared to the no-storage and fixed-storage systems (see Figure 1.10(d)). The low density fixed-storage system has a higher profit than the high density one when the pickup proportion is low (i.e., the delivery cost is lower than \$0.8). Put differently, there is no reason to set up numerous fixed-storage locations when the delivery cost is low and when the pickup option is not attractive.

(4) The impact of the marginal keeping cost: Both for fixed storage and crowd keepers, the same fixed cost is taken. However, an additional inconvenience cost for keeping each parcel is included for crowd keepers, which may stem from reserving space, contacting the customer, and the associated hassle. This marginal keeping cost is also viewed as a lower bound of the marginal earnings for keepers, and therefore keeper availability highly depends on the keeping cost. Figures 1.11(a)−(c) present how the pickup proportion, the delivery time, the platform profit change, respectively, as the marginal keeping cost changes, and Figure 1.11(d) shows how the fees and compensation change.

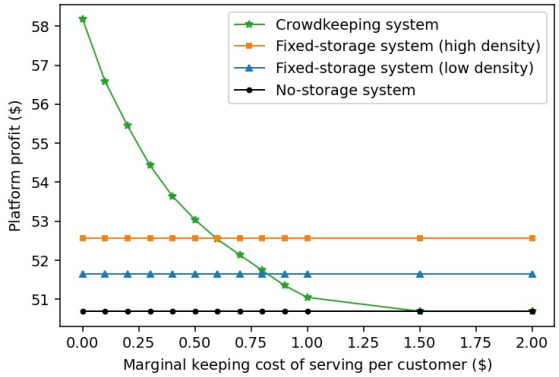
When the marginal keeping cost increases, the pickup proportion decreases (see Figure 1.11(a)), the delivery time increases (see Figure 1.11(b)), and the platform profit decreases (see Figure 1.11(c)). In other words, the increasing keeping cost makes the system less efficient and reduces the platform benefits. When crowd keepers suffer higher inconvenience, the platform has to increase the compensation offered to them to ensure their availability thus sacrificing part of its profits (see Figure 1.11(d)). Even in this case, many customers end up switching to the delivery option due to the reduced availability. When the marginal keeping cost increases to \$2, the inconvenience is so high that no keeper offers to provide their services, both the pickup proportion and the compensation decrease to zero, and the crowdkeeping system converges to the no-storage system. The crowdkeeping system performs worse than the high-density fixed-storage system when the keeping cost increases up to \$0.6, and worse than the low-density fixed-storage system when the keeping cost increases up to \$0.9, due to that the marginal cost of using crowd keepers becomes unreasonably expensive, while the fixed cost is \$0.1. We note that our study offers a rather optimistic view of fixed storage with the same fixed cost for being available as crowd keepers but without any extra cost of keeping one more parcel. We therefore



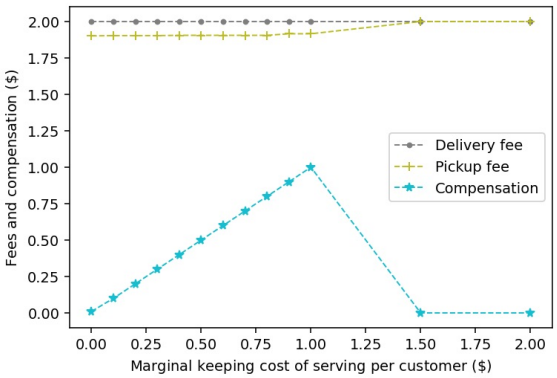
(a) Pickup proportion



(b) Delivery time



(c) Platform profit

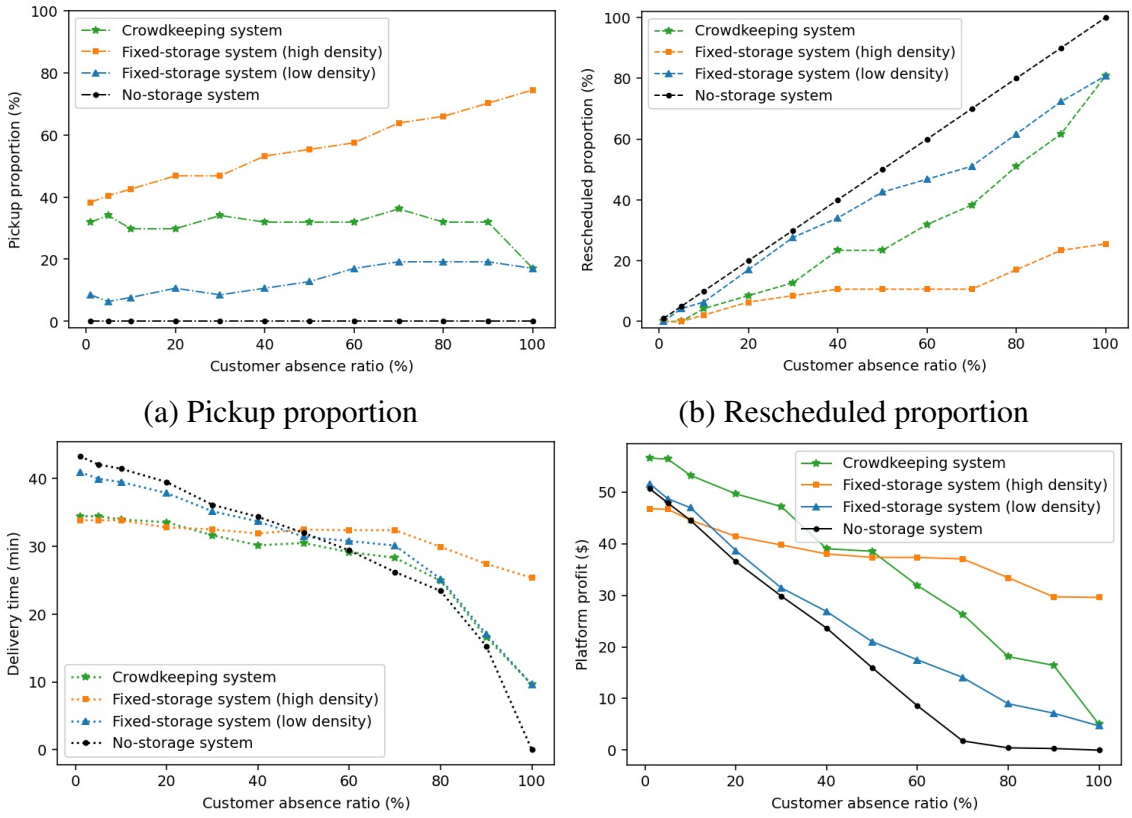


(d) Pickup fee and compensation of crowdkeeping

Figure 1.11 – The impact of the keeping cost

suspect that, in practice, crowdkeeping may still be the most beneficial option in some of these settings.

(5) The impact of the absence ratio: In the no-storage system, deliveries have to be rescheduled when customers are absent, leading to inefficiencies. We investigate if the fixed-storage and the crowd-keeper systems can eliminate this inefficiency. Figures 1.12(a)−(d) present how the pickup proportion, the rescheduled proportion, the delivery time, and the platform profit change, respectively, as the customer absence ratio changes. When the customer absence ratio increases, the pickup proportion of the crowdkeeping system is overall stable, but has a decreasing tendency (see Figure 1.12(a)). The rescheduled proportion increases (see Figure 1.12(b)), leading to a decrease in delivery time and profit (see Figures 1.12(c) and (d)). This is because the availability of customer keep-



(a) Pickup proportion

(b) Rescheduled proportion

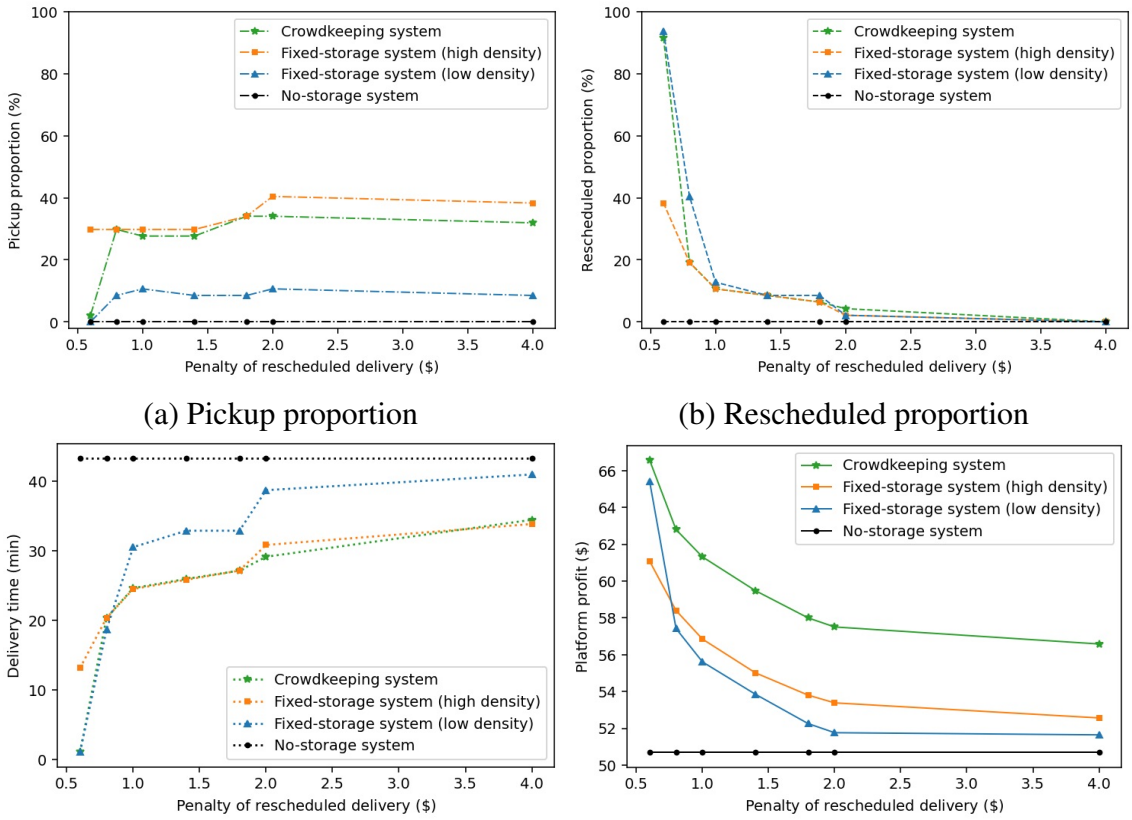
(c) Delivery time

(d) Platform profit

Figure 1.12 – The impact of the customer absence ratio

ers decreases when more customers are absent. Both fixed-storage systems have an increasing pickup proportion and rescheduled proportion, and the high density one tends to encourage customers to pick up with the highest pickup proportion, whereas the low density one tends to reschedule the deliveries with the high rescheduled proportion. For the no-storage system, the delivery time and the platform profit decrease to zero when the absence ratio is 100%. That is, all absent customers have no choice but reschedule their deliveries. For the crowd-keeper and fixed-storage systems, in addition to the rescheduled delivery, absent customers can choose to be served by keepers or storages, therefore leading to a positive profit. When the absence ratio is lower than 10%, the high density fixed-storage system has the worst performance on profit, but tends to become better as the absence ratio increases. It has the best performance due to a high consolidation of

parcels when the absence ratio is higher than 50%. As long as the absence ratio is lower than 50%, which is a common setting in the real world, the crowdkeeping system has the best performance on profit.



(a) Pickup proportion

(b) Rescheduled proportion

(c) Delivery time

(d) Platform profit

Figure 1.13 – The impact of the penalty for rescheduling deliveries

(6) The impact of the penalty for rescheduling deliveries: We model two types of rescheduled deliveries in the crowdkeeping system. The first one is incurred by customers due to their absence, and the second one is determined by the platform if those customers who prefer the pickup option cannot be served by any nearby available keeper. We fix the cost of the first type, and investigate the impact of the penalty for the second type. Figures 1.13(a)−(d) respectively show how the pickup proportion, the rescheduled proportion, the delivery time, and the platform profit change as the penalty of rescheduling deliveries changes. When the penalty of rescheduling deliveries increases, the pickup proportion

tends to increase but is overall stable (see Figure 1.13(a)), the rescheduled proportion decreases rapidly (see Figure 1.13(b)), leading to an increased delivery time (see Figure 1.13(c)) and a decreased profit (see Figure 1.13(d)). The crowdkeeping system always has the highest profit whether the penalty is high or low. Even though the low penalty yields a high profit, when implementing the system in practice, the penalty should be at least as high as the delivery fee in order to avoid unnecessary rescheduled deliveries, since customers would prefer to receive their parcels as soon as possible.

1.7 Conclusion

We have presented a new business model in last-mile deliveries, the key idea of which is to make use of the unused space owned by the crowd to provide crowdkeeping services. Crowd keepers have more flexibility, larger availability, and lower costs than what is offered by fixed storage, and this leads to a more efficient and a more profitable system for last-mile deliveries. We have constructed a bilevel program by considering customer preferences, keeper behaviors, and platform operations. We have used the strong duality to reformulate the bilevel program into an equivalent single-level program, have derived a mixed-integer linear programming model with subtour elimination constraints by applying linearization techniques, and have solved the model to optimality using a row generation algorithm. To improve the efficiency of the solution procedure, we have derived a more compact representation of the best response set of customers and keepers, and have developed an approximation model for the bilevel program by approximating the optimal travel time using linear regression.

The numerical study is implemented on a real-world dataset provided by Amazon. The results indicate that the crowdkeeping delivery system benefits from economies of scale, as platform profits increase by serving more customers with an increasing marginal profit and a decreasing marginal delivery time for serving one more customer. Additionally, both the platform and the system benefit from delivery consolidations. Specifically, the platform earns more profits, and the system causes less pollution in cases with a larger

service range, lower pickup costs, higher delivery costs, lower keeping costs, and higher customer keeper availabilities. These cases always accompany a higher pickup proportion, implying that more deliveries are consolidated. Compared to the no-storage and fixed-storage systems, the crowd-keeper system is beneficial for all participants in the last-mile delivery system by improving platform profits, reducing environmental pollutions due to fewer truck deliveries, and bringing about more savings for customers and extra earnings for keepers. The reason is that crowd keepers are capable of consolidating deliveries and eliminating failed deliveries, and this capability is higher than fixed storage due to their greater flexibility and larger availability provided by the crowd. If it was possible to set up a very dense fixed storage network with a low cost, crowdkeeping may not be advantageous (1) when customers have high tolerance for walking long time to pick up their parcels, (2) when crowd keepers have high marginal cost for keeping each parcel but fixed storage have no marginal cost, or (3) when most of the customers are absent to attend their deliveries. In these cases, the capability of fixed storage to consolidate deliveries is higher than that of crowd keepers. However, in all other common cases, the crowdkeeping system may deliver the best performance on the profitability.

The study can be extended in multiple directions. First, we assumed that the information is completely shared among all participants before optimizing the delivery operations. However, in practice the customer density may not be known when the pickup fee and the compensation decisions are made, and the keeper final availability could be uncertain before the delivery takes place. It would be interesting to consider both uncertainties. Additionally, a promising direction for future research is to accommodate multi-period real-time optimization by considering the sequential arrival of keepers and customers and facilitating dynamic service allocation among evolving groups of customers and keepers. In this case, modeling time-windows is an also important facet and can potentially better highlight the importance of crowd keepers. Moreover, co-existence of multiple delivery types such as store points, automated lockers and crowd keepers can be considered in the same problem under a customer choice model. Lastly, even though the new business model offers a win-win situation for participants, there may be additional complications

in real-world, such as the incentive of a long-term cooperation with e-commerce and delivery companies, which we leave for future studies.

1.8 Appendix

Appendix A: Delivery systems

We develop models for the no-storage system, the fixed-storage system, and the crowd-delivering-keeper system, and demonstrate their main differences.

A.1: The No-Storage System

In the no-storage setting, the distribution company has to deliver all the parcels to customers' home addresses. If customers are absent, a second visit is necessary. In this case, we consider the situation where customers can delay and reschedule their deliveries. Let $\mathscr{N}$ be the set of customers to be visited and $\mathscr{N}'$ be the set of absent ones. Since there is no storage or keeper in the system, the deliveries of absent customers have to be rescheduled, and the profit will not be captured. The optimization model aims to maximize the profit for visiting customers in $\mathscr{N} \setminus \mathscr{N}'$.

A.2: The Fixed-Storage System

The fixed storage, including the automated locker and pickup location, is safe and efficient, and has been implemented in real-world. If customers are not available, their parcels are kept in storages, from where customers can pick up at their convenience. However, the pickup locations are fixed, their capacities are limited, and they may have a high setup cost. In this case, customers have the choice of picking up from fixed storage by paying a lower pickup fee than the standard delivery fee (e.g., IKEA, Walmart, and Zara). The main difference between the fixed-storage and crowd-keeper systems is that in the former system, there is always a fixed setup cost whether being used or not but no compensation for serving each parcel. Since there is a trade-off between the setup cost and the storage density, we consider two versions of the fixed-storage system: (1) a "low density fixed-storage system" where the potential fix-storage locations, $j \in \mathscr{M}$, have the same density as the non-customer keepers, and (2) a "high density fixed-storage system" where

the potential fix-storage locations, $j \in \mathscr{N} \cup \mathscr{M}$, have the same density as the union set of keepers and customers. Even though it may be very costly in practice to set up such a high density fixed storage network, we take it as the best case on the performance of fixed storage. The optimal solutions can be obtained using the same methods as presented in Section 1.5.

A.3: The Crowd-Delivering-Keeper System

The Pickme company recently piloted a new service where customers have the option of requesting home delivery of their parcels by keepers. Keepers, who not only keep the parcels but also deliver them to customers, are named as *crowd-delivering-keepers* and earn €1.25 by serving each customer. In this setting, customers do not have to pick up from their specified keepers. Therefore, they do not have to make any decisions, but need to pay the delivery fee to receive their parcels at their location. In other words, there is no difference from customers' point of view between the no-storage and crowd-delivering-keeper systems. Crowd-delivering-keepers need to determine if they are willing to work as a keeper (represented by the variable w) and if they accept each request of visiting the customer assigned to them ($\hat{v}$). A bilevel program is constructed when both the platform and keepers maximize their profits.

The model for the crowd-delivering-keeper $j \in \mathscr{M} \cup \mathscr{N}$ is

$$
\begin{aligned}
G_j(\mathfrak{c}, v) \triangleq \max_{\hat{v}_{:j}, w_j} \quad & \left(\sum_{i \in \mathscr{N}} (\mathfrak{c} - c^p t_{ij}) \hat{v}_{ij} \right) - c^k w_j && \text{(1.11a)} \\
\text{s.t.} \quad & w_j \leq 1 - a_j, && \text{(1.11b)} \\
& \hat{v}_{ij} \leq v_{ij}, \forall i \in \mathscr{N} && \text{(1.11c)} \\
& \sum_{i \in \mathscr{N}} \hat{v}_{ij} \leq b_j w_j, \forall i \in \mathscr{N} && \text{(1.11d)} \\
& \hat{v}_{ij}, w_j \in \{0,1\}, \forall i \in \mathscr{N}, && \text{(1.11e)}
\end{aligned}
$$

where $v_{:j}$ denotes a column vector. Objective function (1.11a) states that keepers accept the allocated deliveries only when the total to-be-earned compensation is higher than the inconvenience cost for keeping and delivering services. In other words, keepers are

active only when the total profit is positive. The constraint (1.11b) states that the crowd can be keepers only when they are available. The constraints (1.11c) imply that keepers can only choose to serve the customers that are assigned to them by the platform. The constraints (1.11d) state that only active keepers can serve customers and that the number of customers served by each keeper should not be greater than its capacity. The constraints (1.11e) are domain restrictions.

The platform model is

$$G_P \triangleq \max_{\mathfrak{c},v} \quad nf^d - \mathfrak{c} \sum_{i\in\mathscr{N}} \sum_{j\in\mathscr{M}\cup\mathscr{N}} \hat{v}_{ij} - g(w,\hat{v}) \tag{1.12a}$$

$$\text{s.t.} \quad v_{ij} \le r_{ij}, \forall i \in \mathscr{N}, \forall j \in \mathscr{M}\cup\mathscr{N} \tag{1.12b}$$

$$v_{ij} \in \{0,1\}, \mathfrak{c} \in [0,\bar{\mathfrak{c}}], \tag{1.12c}$$

with the third-stage model

$$g(w,\hat{v}) \triangleq \min_{x,y,\hat{z}} \quad c^d \sum_{i\in\mathscr{M}\cup\mathscr{N}} \sum_{j\in\mathscr{M}\cup\mathscr{N}} t_{ij}x_{ij} + \hat{c}^r \sum_{i\in\mathscr{N}} \hat{z}_i \tag{1.13a}$$

$$\text{s.t.} \quad y_j \ge w_j, \forall j \in \mathscr{M}\cup\mathscr{N} \tag{1.13b}$$

$$y_i + \sum_{j\in\mathscr{M}\cup\mathscr{N}} \hat{v}_{ij} + \hat{z}_i = 1, \forall i \in \mathscr{N} \tag{1.13c}$$

$$y_i \le 1 - a_i, \forall i \in \mathscr{N} \tag{1.13d}$$

$$\hat{z}_i \le a_i, \forall i \in \mathscr{N} \tag{1.13e}$$

$$\sum_{i\in\mathscr{M}\cup\mathscr{N}} x_{ij} = y_j, \forall j \in \mathscr{M}\cup\mathscr{N} \tag{1.13f}$$

$$\sum_{i\in\mathscr{M}\cup\mathscr{N}} x_{ji} = y_j, \forall j \in \mathscr{M}\cup\mathscr{N} \tag{1.13g}$$

$$\sum_{i,j\in\mathscr{S}} x_{ij} \le |\mathscr{S}| - 1, \forall \mathscr{S} \subset \mathscr{M}\cup\mathscr{N}, 2 \le |\mathscr{S}| \le |\mathscr{M}\cup\mathscr{N}| - 2 \tag{1.13h}$$

$$x_{ij} \in \{0,1\}, \forall i \in \mathscr{M}\cup\mathscr{N}, \forall j \in \mathscr{M}\cup\mathscr{N} \tag{1.13i}$$

$$y_j, z_i \in \{0,1\}, \forall i \in \mathscr{N}, \forall j \in \mathscr{M}\cup\mathscr{N}. \tag{1.13j}$$

The parameters $(f^d, c^d, \hat{c}^r, t_{ij}, a_i, b_j, r_{ij})$ and decisions $(\mathfrak{c}, v, x, y, \hat{z})$ are described in Table 1.2, where $\mathfrak{c}, v$ are the first-stage decisions, and $x, y, \hat{z}$ are the third-stage decisions after

keepers make their decisions $\hat{v}, w$ in the second-stage. Objective function (1.12a) represents the platform revenue, which is equal to the delivery fee paid by all n customers, minus the total cost of offering the compensation, visiting all active nodes, and rescheduling deliveries. The constraints (1.12b) ensure that customers can only be assigned to keepers in the same zone. The constraints (1.12c) are domain restrictions for the first-stage decisions. The constraints (1.13b) and (1.13c) enforce that active keepers and customers are visited. The constraints (1.13c) also ensure that customer i has to be served either by direct delivery ($y_i = 1$), by rescheduled delivery ($\hat{z}_i = 1$), or by a keeper ($\sum_j \hat{v}_{ij} = 1$). The constraints (1.13d) and (1.13e) state that absent customers cannot be served by direct delivery, and that available customers' deliveries should not be rescheduled. The constraints (1.13f)−(1.13j) are degree constraints, subtour elimination constraints, and domain restrictions for visiting all active customers and keepers.

The bilevel program is

$$\begin{aligned}
\max_{\mathfrak{c},\hat{v},x,y,z,w,v} \quad & nf^d - \mathfrak{c} \sum_{i\in\mathscr{N}} \sum_{j\in\mathscr{M}\cup\mathscr{N}} \hat{v}_{ij} - c^d \sum_{i\in\mathscr{M}\cup\mathscr{N}} \sum_{j\in\mathscr{M}\cup\mathscr{N}} t_{ij}x_{ij} - \sum_{i\in\mathscr{N}} \hat{c}^r \hat{z}_i \\
\text{s.t.} \quad & (1.12b) - (1.12c), (1.13b) - (1.13j) \\
& (\hat{v}_{:j}, w_j) \in \arg\max G_j(\mathfrak{c}, v), \forall j \in \mathscr{M}\cup\mathscr{N}.
\end{aligned}$$

Then, we reformulate the bilevel program into an equivalent single-level program.

$$\begin{aligned}
\max_{\mathfrak{c},\hat{v},x,y,z,w,v} \quad & nf^d - \mathfrak{c} \sum_{i\in\mathscr{N}} \sum_{j\in\mathscr{M}\cup\mathscr{N}} \hat{v}_{ij} - c^d \sum_{i\in\mathscr{M}\cup\mathscr{N}} \sum_{j\in\mathscr{M}\cup\mathscr{N}} t_{ij}x_{ij} - \sum_{i\in\mathscr{N}} \hat{c}^r \hat{z}_i \\
\text{s.t.} \quad & [(1.11b) - (1.11e), \forall j \in \mathscr{M}\cup\mathscr{N}], (1.12b) - (1.12c), (1.13b) - (1.13j) \\
& \mathfrak{c} \sum_{i\in\mathscr{N}} v_{ij} \geq c^k w_j + \sum_{i\in\mathscr{N}} c^p t_{ij} \hat{v}_{ij}, \forall j \in \mathscr{M}\cup\mathscr{N} \\
& c^p t_{ij} \hat{v}_{ij} \leq \mathfrak{c}, \forall i \in \mathscr{N}, \forall j \in \mathscr{M}\cup\mathscr{N}.
\end{aligned}$$

Compared to BP, the main differences in this model are the following. (1) Some decisions made by the platform are reduced, including the pickup fee, the maximum number of customers proposed to be served by keepers, and the final assignment. (2) Customers do not express their preferences since the form of delivery is decided through the negotiation between the platform and the crowd-delivering-keepers. (3) Keepers decide which subset

of customers they accept to serve and are responsible for the delivery of their parcels. This model can be solved to optimality using the same techniques applied in Section 1.5.

A.4: Figures of Delivery Systems

To demonstrate the main differences of different delivery systems more explicitly, we show the graph for each system when serving the same customer set. Figure 1.14(a) is the

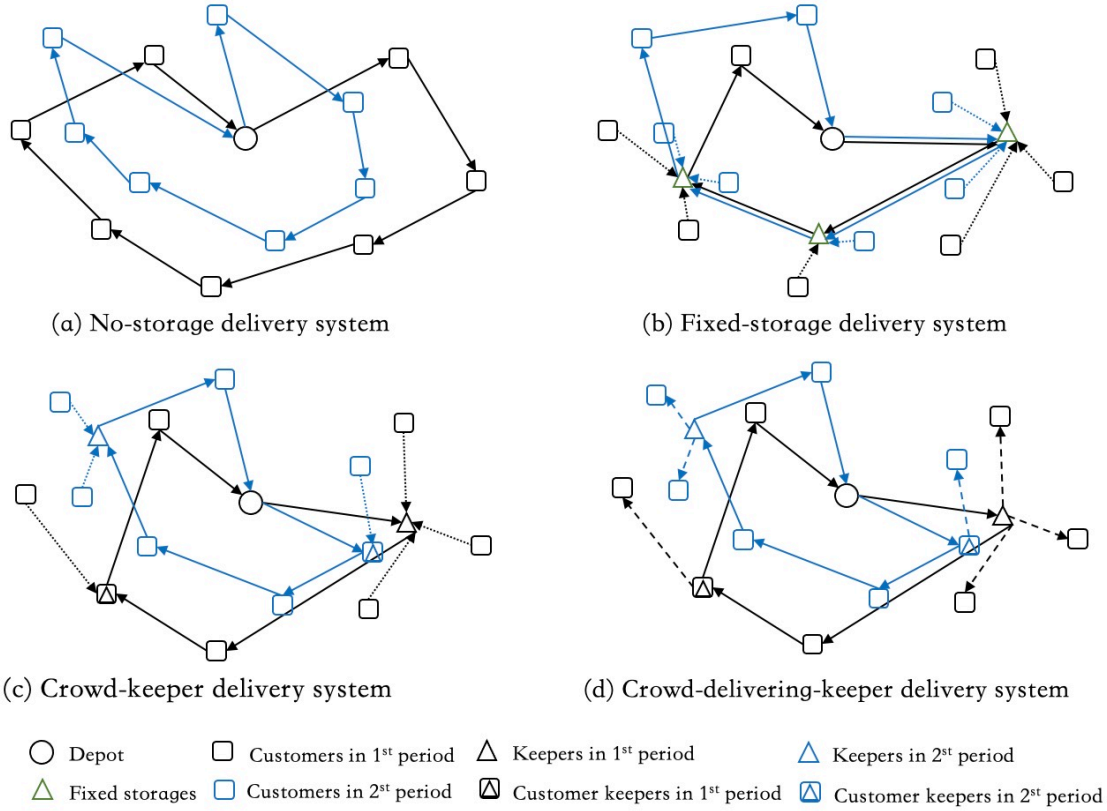


Figure 1.14 – Different systems.

no-storage system, in which all customers in each period are visited by a route. Figure 1.14(b) represents the fixed-storage system, in which customers pick up their parcels from fixed storage. However, storage locations such as automated lockers are always fixed and they need a fixed setup cost to be installed. Figure 1.14(c) represents the crowd-keeper system. In addition to the consolidation, keepers locations are flexible and adapt to the different customer locations. Furthermore, a cost is paid only when the keeper is active,

leading to a higher flexibility than fixed storage. In the crowd-delivering-keeper system in Figure 1.14(d), keepers make deliveries to customers. The main difference between Figure 1.14(c) and 1.14(d) is the direction of the dashed arrows. That is, (c) represents that customers pick up parcels from keepers and (d) represents that keepers deliver parcels to customers.

Appendix B: Proofs

B.1: Proof of Proposition 1

Proof. When $m_i \leq b_i$, let H_i^R be the model formed by relaxing the integrality requirements in model (1.1). Suppose that all optimal solutions to H_i^R have fractional entries. We study the cases $e_i = 1$ and $e_i = 0$ separately. (1) When $e_i = 1$, let (u_i^*, w_i^*, z_i^*) be an optimal solution to H_i^R with $0 < u_i^* < 1$. Due to the constraints (1.1b) and (1.1c) and the fact that $u_i^* > 0$, we have $a_i = 0$ and $z_i^* = 0$. Next, we consider the case that $c^k + (c^s - \mathfrak{c})m_i \geq 0$, where $w_i^* := 0$ is necessarily optimal. In this case, either we can set $u_i^* := 0$ as optimal when $f^d \geq f^p + c^p \max_j t_{ij} v_{ij}$, or set $u_i^* := 1$ as optimal when $f^d < f^p + c^p \max_j t_{ij} v_{ij}$. Alternatively, we have the case that $c^k + (c^s - \mathfrak{c})m_i < 0$ with $w_i^* := u_i^*$ being optimal. In this case, either we can set $u_i^* := 0$ when $f^d + c^k + (c^s - \mathfrak{c})m_i \geq f^p + c^p \max_j t_{ij} v_{ij}$, or set $u_i^* := 1$ when $f^d + c^k + (c^s - \mathfrak{c})m_i < f^p + c^p \max_j t_{ij} v_{ij}$. This contradicts the fact that all optimal solutions are fractional. Similar yet simpler arguments also apply to z_i.

(2) When $e_i = 0$ (i.e., model (1.2)), let w_i^* with $0 < w_i^* < 1$ be an optimal solution. Given that $m_i \leq b_i$, we can set $w_i^* = 1$ if $c^k + (c^s - \mathfrak{c})m_i \leq 0$, or set $w_i^* = 0$ otherwise, without increasing the objective function value. Again, we observe a contradiction. ∎

B.2: Proposition 3 and its Proof

Proposition 3 *A feasible solution in the BP always satisfies $m_j \leq b_j$. Therefore, a relaxed customer model without the constraint $m_j w_j \leq b_j$ can be used when the bilevel model is transformed into its single-level equivalent.*

Proof. Consider a feasible solution of the BP in which there exists a keeper j such that $m_j > b_j$. Since the solution is feasible, constraint (1.1e) ($m_j w_j \leq b_j$) is satisfied, and this implies that $w_j = 0$. Furthermore, constraint (1.4c) ($\hat{v}_{ij} \leq w_j$ for all $i \in \mathscr{N}$) implies that $v_{ij} = 0$ for all $i \in \mathscr{N}$. Due to that we minimize $\mathfrak{c}\sum_j m_j$ in the objective function (1.4a), m_j is zero when the constraint (1.4b) ($\sum_{i\in\mathscr{N}} \hat{v}_{ij} \leq m_j$) holds, leading to a contradiction. We therefore conclude that the BP necessarily has an optimal solution with $m_j \leq b_j$ for all $j \in \mathscr{M} \cup \mathscr{N}$, and that a relaxed customer model without the constraint $m_j w_j \leq b_j$ can be used when the bilevel model is transformed into its single-level equivalent. ∎

B.3: Proof of Proposition 2

Proof. We prove that the optimal solution set of model (1.1) can be characterized by the set of constraints in (1.8). Let η_i^* be the optimal objective function value of model (1.1). The optimal solution set is:

$$\left\{ (u_i, w_i, z_i) \middle| (1.1b) - (1.1e),\ \begin{pmatrix} f^d(u_i + z_i) + [c^k + (c^s - \mathfrak{c})m_i]w_i \\ + \left(f^p + c^p \max\limits_{j\in\mathscr{M}\cup\mathscr{N}} t_{ij} v_{ij} \right)(e_i - u_i - z_i) \end{pmatrix} \leq \eta_i^* \right\}.$$

We are therefore left with the task of showing that:

$$\eta_i^* = \min\left\{ f^d + (1 - a_i)[c^k + (c^s - \mathfrak{c})m_i],\ f^d,\ f^p + c^p \max_{j\in\mathscr{M}\cup\mathscr{N}} t_{ij} v_{ij} \right\}, \forall i \in \mathscr{N},$$

and

$$\eta_i^* = \min\left\{ c^k + (c^s - \mathfrak{c})m_i,\ 0 \right\}, \forall i \in \mathscr{M}.$$

We start with the case $i \in \mathscr{N}$ and exploit the fact that there are three feasible solutions to model (1.1). Namely, the feasible solutions are:

$$\mathscr{X} := \{(u_i = 1 - a_i, w_i = 1 - a_i, z_i = a_i),\ (u_i = 1 - a_i, w_i = 0, z_i = a_i),\ (u_1 = 0, w_i = 0, z_i = 0)\},$$

Hence, by replacing (u_i, w_i, z_i) with their feasible value, we have

$$\eta^* = \min_{(u_i,w_i,z_i)\in\mathscr{X}} f^d(u_i + z_i) + [c^k + (c^s - \mathfrak{c})m_i]w_i + \left(f^p + c^p \max_{j\in\mathscr{M}\cup\mathscr{N}} t_{ij} v_{ij} \right)(e_i - u_i - z_i)$$

$$= \min\left\{ f^d + (1-a_i)[c^k + (c^s - \mathfrak{c})m_i],\ f^d,\ f^p + c^p \max_{j \in \mathscr{M}\cup\mathscr{N}} t_{ij} v_{ij} \right\}.$$

In the case that $i \in \mathscr{M}$, the argument is similar and relies on the feasible solution set taking the form:

$$\mathscr{X} := \{(u_i = 0, w_i = 1, z_i = 0),\ (u_i = 0, w_i = 0, z_i = 0)\}\ .$$

Hence,

$$\eta^* = \min\left\{ c^k + (c^s - \mathfrak{c})m_i,\ 0 \right\}.$$

∎

Appendix C: Comparison of SP_1 and SP_2

We let the customer objective function be $F_i(u_i, z_i, w_i)$:

$$F_i(u_i, z_i, w_i) := f^d(u_i + z_i) + \left[c^k + (c^s - \mathfrak{c})m_i \right] w_i + (f^p + c^p \max_{j \in \mathscr{M}\cup\mathscr{N}} t_{ij} v_{ij})(e_i - u_i - z_i).$$

In SP_2, we have that

(1) If $e_i = 1$ (if $i \in \mathscr{N}$):

$$F_i \le \min\left\{ f^d + (1-a_i)\left[c^k + (c^s - \mathfrak{c})m_i \right], f^d, f^p + c^p \max_{j \in \mathscr{M}\cup\mathscr{N}} t_{ij} v_{ij} \right\}.$$

(2) If $e_i = 0$ (if $i \in \mathscr{M}$):

$$F_i \le \min\left\{ c^k + (c^s - \mathfrak{c})m_i, 0 \right\}.$$

In SP_1, we have less tight upper bounds of F_i. However, we can always derive the same upper bounds as in SP_2 by fixing the value of dual variables ϕ, ν, and λ. To be specific,

(1) If $e_i = 1$ (if $i \in \mathscr{N}$):

1.1) When $a_i = 1$ and $\phi_i = \min\left\{ f^p + c^p \max_{j \in \mathscr{M}\cup\mathscr{N}} t_{ij} v_{ij} - f^d, 0 \right\}$,

$$F_i \le \min\left\{ f^d, f^p + c^p \max_{j \in \mathscr{M}\cup\mathscr{N}} t_{ij} v_{ij} \right\}.$$

1.2) When $a_i = 0$ and $v_i = \min \left\{ \begin{array}{l} f^p + c^p \max\limits_{j \in \mathscr{M} \cup \mathscr{N}} t_{ij} v_{ij} - f^d - \left[c^k + (c^s - \mathfrak{c}) m_i \right], \\ f^p + c^p \max\limits_{j \in \mathscr{M} \cup \mathscr{N}} t_{ij} v_{ij} - f^d, 0 \end{array} \right\}$,

$$F_i \leq \min \left\{ f^d + \left[c^k + (c^s - \mathfrak{c}) m_i \right], f^d, f^p + c^p \max_{j \in \mathscr{M} \cup \mathscr{N}} t_{ij} v_{ij} \right\}.$$

(2) If $e_i = 0$ (if $i \in \mathscr{M}$):

When $\lambda_i = \min \left\{ -c^k - (c^s - \mathfrak{c}) m_i, 0 \right\}$,

$$F_i \leq \min \left\{ c^k + (c^s - \mathfrak{c}) m_i, 0 \right\}.$$

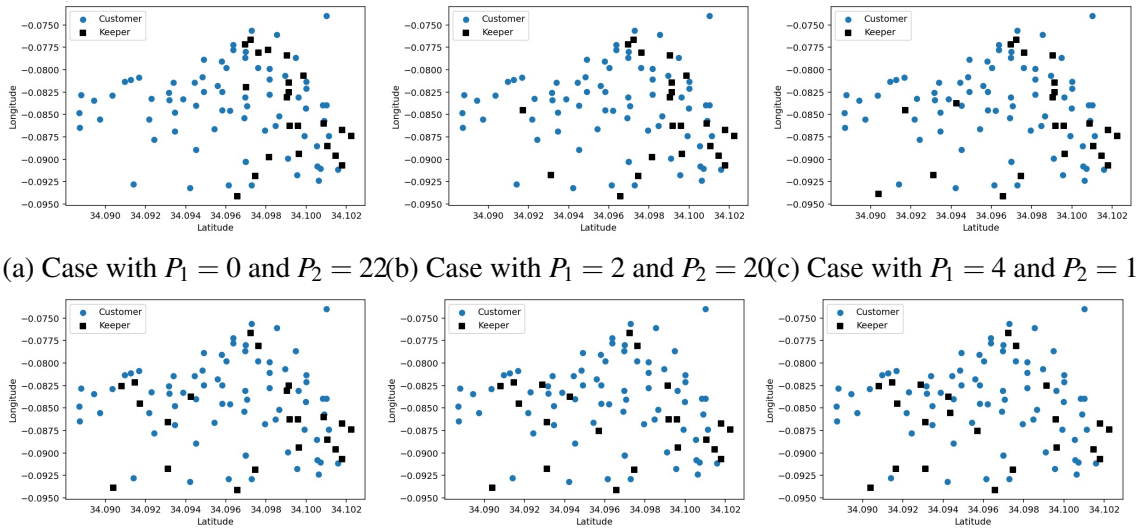
Overall, we observe that SP_2 is more efficient than SP_1 because it does not require a search for optimal dual variables and therefore has a smaller search space. Since LSP_1 and LSP_2 are the equivalent formulations of SP_1 and SP_2, respectively, we conclude that LSP_2 is also more efficient than LSP_1 for the same reason.

Appendix D: TSP(n) for Varying Non-customer Keeper Distributions

In Section 1.6.2, we proposed a procedure to calibrate an optimal tour length estimator based on randomly selected locations from a dataset of historical customer locations for the region of interest. In Section 1.6.3, we then evaluated the quality of estimation under varying customer locations and densities by fixing the non-customer keeper locations. In this appendix, we further evaluate the performance of estimator with varying non-customer keeper locations and densities.

To distinguish the non-customer keeper distribution from the customer distribution, we consider their distributions as follows. The entire region (e.g., Figure 1.5) is divided into "left" and "right" parts. In each instance, we choose 72 points as customer locations and designate the remaining 44 nodes as potential locations of non-customer keepers. Customers are uniformly distributed on both parts, while non-customer keeper distributions may vary. For example, in Figure 1.15, we distinguish non-customer keeper distributions in the left and right regions by adjusting the number of keepers in each part, while maintaining the total number of keepers fixed. From Figure 1.15(a) to (f), non-customer

keepers are initially spatially clustered in the right region, and then they gradually transition to a more uniform distribution across the entire region. In Figure 1.16, we alter the non-customer keeper distributions by varying the total number of non-customer keepers from 0 to 44 across the entire region.

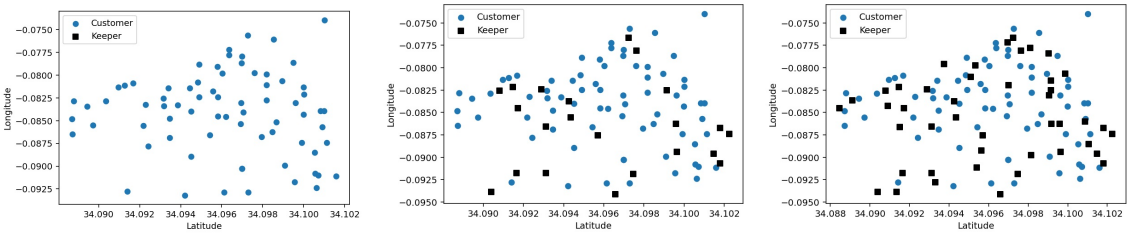


(a) Case with $P_1 = 0$ and $P_2 = 22$ (b) Case with $P_1 = 2$ and $P_2 = 20$ (c) Case with $P_1 = 4$ and $P_2 = 18$

(d) Case with $P_1 = 7$ and $P_2 = 15$ (e) Case with $P_1 = 9$ and $P_2 = 13$ (f) Case with $P_1 = 11$ and $P_2 = 11$

Figure 1.15 – Sample cases with varying distributions of non-customer keepers, featuring a fixed number and changing locations.

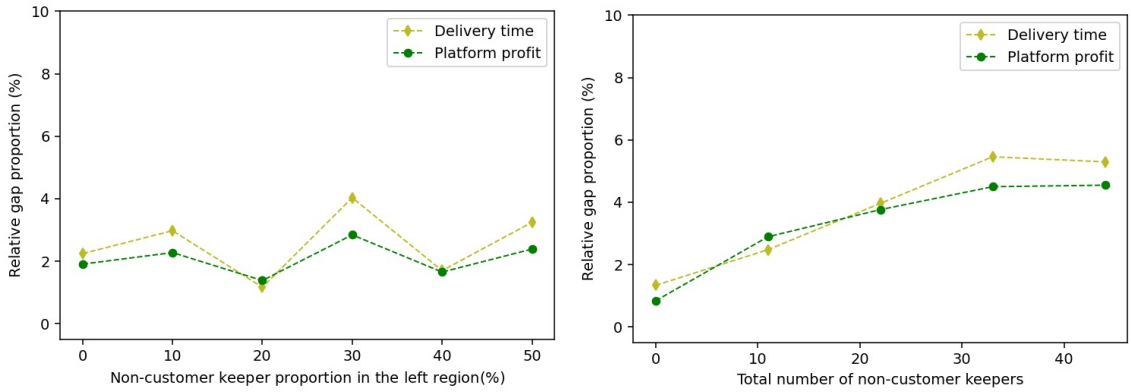
Notes. There are a total of 72 customers and 22 potential keepers. P_1 and P_2 are the numbers of keepers in the left and right region, respectively. Cases are obtained with a random seed of 0.



(a) Case without keepers (b) Case with 22 keepers (c) Case with 44 keepers

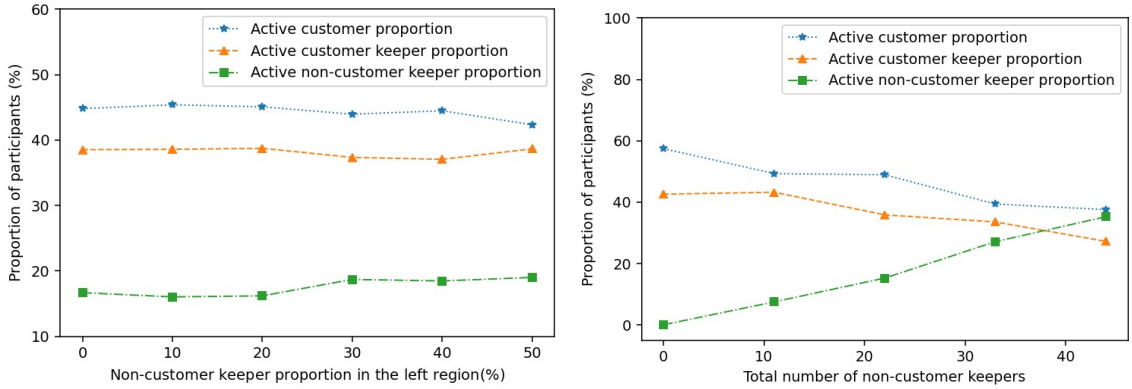
Figure 1.16 – Sample cases with varying distributions of non-customer keepers, including changes in both the number and locations.

Notes. There are a total of 72 customers and at most 44 potential keepers.



(a) Effect of keeper proportion in the left region (b) Effect of number of keepers

Figure 1.17 – The effect of non-customer keeper distribution on the relative gap between exact and approximated solutions.



(a) Effect of keeper proportion in the left region (b) Effect of number of keepers

Figure 1.18 – The effect of non-customer keeper distribution on the proportions of participants visited in the optimal tour.

We derive the exact solutions and utilize the optimal tour length estimator $\mathrm{TSP}(n) = 9\sqrt{n} - 17$ specifically to obtain the approximated solutions. We present the results as follows. In Figure 1.17, we reported the relative gap between the exact and approximated solutions by changing two factors: the *non-customer keeper proportion in the left region*, defined as the proportion of non-customer keepers located in the left region among all non-customer keepers in the entire region, and the *number of non-customer keepers in the entire region*. The average approximation performance across all these instances, as measured by the relative gap in delivery time or platform profit, is below 6% and 5%,

respectively. We note that the optimal tour length estimator and its performance may vary from region to region. As a future direction to enhance the estimator's performance further, we could consider incorporating more instances collected from daily life, categorizing them into different distributions, and developing specific fits for each distribution.

We also presented the proportions of different participants visited in the optimal tour, including the active customers (who choose delivery but do not serve others), active customer keepers (who receive direct deliveries and serve others), and active non-customer keepers (who are not customers but declare availability to serve others), while varying the non-customer keeper proportion in the left region in Figure 1.18(a), and while varying the total number of non-customer keepers in Figure 1.18(b). When non-customer keepers are more uniformly distributed, there is a slight increase in the proportion of active non-customer keepers from 16% to 19%, indicating that more non-customer keepers are selected to serve customers. However, the proportion of active non-customer keepers remains stable regardless of their locations, as most customers who choose pickup are served by customer keepers instead of non-customer keepers. This is evidenced by the active customer keeper proportion being 38.6% and the active non-customer keeper proportion being 19%. As the total number of non-customer keepers increases, the proportion of active non-customer keepers significantly increases, while the proportions of active customers and customer keepers both decrease. This is because there are more available non-customer keepers to be selected to serve customers, and more customers opt for pickup, being served by non-customer keepers.

References

Agatz, Niels et al. (2011). “Time slot management in attended home delivery”. In: *Transportation Science* 45.3, pp. 435–449.

Alnaggar, Aliaa, Fatma Gzara, and James H. Bookbinder (2021). “Crowdsourced delivery: A review of platforms and academic literature”. In: *Omega* 98, p. 102139.

Applegate, David L et al. (2007). *The Traveling Salesman Problem: A Computational Study*. Princeton: Princeton University press.

Archetti, Claudia, Francesca Guerriero, and Giusy Macrina (2021). “The online vehicle routing problem with occasional drivers”. In: *Computers & Operations Research* 127, p. 105144.

Arslan, Alp M et al. (2019). “Crowdsourced delivery—A dynamic pickup and delivery problem with ad hoc drivers”. In: *Transportation Science* 53.1, pp. 222–235.

Arslan, Okan (2021). “The location-or-routing problem”. In: *Transportation Research Part B: Methodological* 147, pp. 1–21.

Beardwood, Jillian, John H Halton, and John Michael Hammersley (1959). “The shortest path through many points”. In: *Mathematical Proceedings of the Cambridge Philosophical Society* 55.4, pp. 299–327.

Bruck, Bruno P., Filippo Castegini, et al. (2020). “A Decision Support System for Attended Home Services”. In: *INFORMS Journal on Applied Analytics* 50.2, pp. 137–152.

Bruck, Bruno P., Jean-François Cordeau, and Manuel Iori (2018). “A practical time slot management and routing problem for attended home services”. In: *Omega* 81, pp. 208–219.

CaiNiao (2023). *How to make money by being a keeper? (In Chinese)*. Last accessed on Jan 01, 2023. URL: `https://cn.alicdn.com/video/cainiao/yizhan/main.mp4`.

Campbell, Ann Melissa and Martin Savelsbergh (2006). “Incentive schemes for attended home delivery services”. In: *Transportation Science* 40.3, pp. 327–341.

Carbone, Valentina, Aurélien Rouquet, and Christine Roussat (2017). "The rise of crowd logistics: a new way to co-create logistics value". In: *Journal of Business Logistics* 38.4, pp. 238–252.

Cavdar, Bahar and Joel Sokol (2015). "A distribution-free TSP tour length estimation model for random graphs". In: *European Journal of Operational Research* 243.2, pp. 588–598.

Dantzig, George, Ray Fulkerson, and Selmer Johnson (1954). "Solution of a large-scale traveling-salesman problem". In: *Journal of the operations research society of America* 2.4, pp. 393–410.

Dayarian, Iman and Martin Savelsbergh (2020). "Crowdshipping and same-day delivery: Employing in-store customers to deliver online orders". In: *Production and Operations Management* 29.9, pp. 2153–2174.

Deloison, T et al. (2020). *The Future of the Last-Mile Ecosystem*. Tech. rep. World Economic Forum, Switzerland.

Devari, Aashwinikumar, Alexander G Nikolaev, and Qing He (2017). "Crowdsourcing the last mile delivery of online orders by exploiting the social networks of retail store customers". In: *Transportation Research Part E: Logistics and Transportation Review* 105, pp. 105–122.

Franceschetti, Anna, Ola Jabali, and Gilbert Laporte (2017). "Continuous approximation models in freight distribution management". In: *Top* 25.3, pp. 413–433.

Gendreau, Michel, Gilbert Laporte, and Frédéric Semet (1997). "The covering tour problem". In: *Operations Research* 45.4, pp. 568–576.

Hasija, Sameer, Zuo-Jun Max Shen, and Chung-Piaw Teo (2020). "Smart city operations: Modeling challenges and opportunities". In: *Manufacturing & Service Operations Management* 22.1, pp. 203–213.

Jacobs, K et al. (2019). *The last-mile delivery challenge*. Tech. rep. Capgemini Research Institute, Paris.

Joerss, Martin et al. (2016). *Parcel delivery: The future of last mile*. Tech. rep. McKinsey & Company, Chicago.

Jozefowiez, Nicolas (2014). “A branch-and-price algorithm for the multivehicle covering tour problem”. In: *Networks* 64.3, pp. 160–168.

Kartal, Zuhal, Servet Hasgul, and Andreas T Ernst (2017). “Single allocation p-hub median location and routing problem with simultaneous pick-up and delivery”. In: *Transportation Research Part E: Logistics and Transportation Review* 108, pp. 141–159.

Klein, Robert, Jochen Mackert, et al. (2018). “A model-based approximation of opportunity cost for dynamic pricing in attended home delivery”. In: *OR Spectrum* 40.4, pp. 969–996.

Klein, Robert, Michael Neugebauer, et al. (2019). “Differentiated time slot pricing under routing considerations in attended home delivery”. In: *Transportation Science* 53.1, pp. 236–255.

Koch, Sebastian and Robert Klein (2020). “Route-based approximate dynamic programming for dynamic pricing in attended home delivery”. In: *European Journal of Operational Research* 287.2, pp. 633–652.

Le, Tho V et al. (2019). “Supply, demand, operations, and management of crowd-shipping services: A review and empirical evidence”. In: *Transportation Research Part C: Emerging Technologies* 103, pp. 83–103.

Lin, Yun Hui et al. (2020). “Last-mile delivery: Optimal locker location under multinomial logit choice model”. In: *Transportation Research Part E: Logistics and Transportation Review* 142, p. 102059.

Martinez-Sykora, Antonio et al. (2020). “Optimised solutions to the last-mile delivery problem in London using a combination of walking and driving”. In: *Annals of Operations Research* 295.2, pp. 645–693.

McKinnon, Alan C and Deepak Tallam (2003). “Unattended delivery to the home: an assessment of the security implications”. In: *International Journal of Retail & Distribution Management* 31(1), pp. 30–41.

Merchan, Daniel et al. (2021). *Amazon Last Mile Routing Research Challenge Dataset*. Accessed January 6, 2022. Seattle: Amazon.com. URL: `https://registry.opendata.aws/amazon-last-mile-challenges`.

Padberg, Manfred and Giovanni Rinaldi (1991). "A branch-and-cut algorithm for the resolution of large-scale symmetric traveling salesman problems". In: *SIAM review* 33.1, pp. 60–100.

Perboli, Guido, Roberto Tadei, and Daniele Vigo (2011). "The two-echelon capacitated vehicle routing problem: Models and math-based heuristics". In: *Transportation Science* 45.3, pp. 364–380.

Pickme (2024). *Let's build tomorrow's delivery together! (In French)*. Last accessed on Feb 11, 2023. URL: `https://www.mypickme.com/`.

Qi, Wei et al. (2018). "Shared mobility for last-mile delivery: Design, operational prescriptions, and environmental impact". In: *Manufacturing & Service Operations Management* 20.4, pp. 737–751.

Rohmer, Sonja and Bernard Gendron (2020). *A Guide to Parcel Lockers in Last Mile Distribution: Highlighting Challenges and Opportunities from an OR Perspective*. Tech. rep. Montreal: CIRRELT.

Savelsbergh, Martin and Tom Van Woensel (2016). "50th anniversary invited article—city logistics: Challenges and opportunities". In: *Transportation Science* 50.2, pp. 579–590.

Schwerdfeger, Stefan and Nils Boysen (2020). "Optimizing the changing locations of mobile parcel lockers in last-mile distribution". In: *European Journal of Operational Research* 285.3, pp. 1077–1094.

Spliet, Remy and Adriana F. Gabor (2015). "The Time Window Assignment Vehicle Routing Problem". In: *Transportation Science* 49.4, pp. 721–731.

Toth, Paolo and Daniele Vigo (2014). *Vehicle Routing: Problems, Methods, and Applications*. Ed. by Daniele Vigo and Paolo Toth. Philadelphia: Society for Industrial and Applied Mathematics.

Ulmer, Marlin and Martin Savelsbergh (2020). "Workforce Scheduling in the Era of Crowdsourced Delivery". In: *Transportation Science* 54.4, pp. 1113–1133.

Ulmer, Marlin W (2020). "Dynamic pricing and routing for same-day delivery". In: *Transportation Science* 54.4, pp. 1016–1033.

Vidal, Thibaut, Gilbert Laporte, and Piotr Matl (2020). “A concise guide to existing and emerging problem variants”. In: *European Journal of Operational Research* 286.2, pp. 401–416.

Yang, Xinan and Arne K Strauss (2017). “An approximate dynamic programming approach to attended home delivery management”. In: *European Journal of Operational Research* 263.3, pp. 935–945.

Yildiz, Baris and Martin Savelsbergh (2020). “Pricing for delivery time flexibility”. In: *Transportation Research Part B: Methodological* 133, pp. 230–256.

Chapter 2

Network Design and Service Guarantee in Ultra-Fast Delivery

Abstract

Ultra-fast delivery revolutionizes food and grocery services, with several companies advertising delivery times under 15 to 30 minutes. Motivated by the multi-billion-dollar industry that has emerged in recent years within the delivery business, we investigate the network design problem for ultra-fast delivery services. This involves decisions on micro-depot locations and customer allocations, considering various service guarantee levels. We develop robust probabilistic envelope constrained (PEC) programs to handle uncertainties in travel times and customer order arrivals, and jointly optimize the protection level to avoid both excessive risk and conservatism. To enhance the tractability of PEC models, we derive their equivalent semi-infinite linear programs and propose inner and outer approximations with finite linear constraints. We validate the accuracy of these approximations through extensive experiments using real-world data from Amazon and the Google API, along with a comparative study of different formulations. Varying service levels in ultra-fast delivery affect profitability and reliability, contingent on service level definitions and compliance probabilities of these guaranteed service levels. We find

that a daily service level with multi-layer partial protection outperforms other policies investigated in this paper, yielding higher profitability and mild violations of service level guarantees, and it proves to be an effective strategy for profitable and reliable ultra-fast delivery without over-committing or under-delivering, regardless of ordering times or traffic conditions. Additionally, empirical evidence indicates that providing ultra-fast delivery in rural areas poses unique challenges compared to urban settings.

2.1 Introduction

Ultra-fast delivery is a new form of the fast and reliable delivery of food and groceries from micro-depots to customers. For example, an ultra-fast delivery company, Getir, promises to deliver groceries to the customer's doorstep within 15 minutes (Kavuk et al. 2022). Investors and entrepreneurs (e.g., Getir, Gopuff, Gorillas) invest heavily in such services and the projected market volume reaches up to $251.50 billions by 2028 (Statista 2023). They expect to attract a large market share by offering urgently needed items without customers having to leave the comfort of their homes, and aim to reduce waste by taking the role of the traditional fridge and storage (Repko 2021).

Ultra-fast delivery has its roots in the 15-minute city concept proposed by Carlos Moreno in 2016 (Moreno et al. 2021). This concept suggests that cities could be designed with the intention of having amenities and most services located within a 15-minute walking or driving distance, thereby fostering a new neighborhood approach. To relieve or confront the climate crisis and potential future pandemics, the 15-minute city and other similar ideas such as the 20-minute neighborhood (Capasso Da Silva, King, and Lemar 2019) have recently gained popularity. The key idea is decentralization in city design, that is, developing different services for each district, encouraging local shops, facilitating short commutes, and enabling access to key services within proximity.

Similar to the 15-minute city, ultra-fast delivery promises to bring advantages of proximity, sustainability, and accessibility, and therefore reduce car dependency, fuel consumption and pollution, and improve customer satisfaction. However, the reality shows

that many startups providing ultra-fast delivery services are facing severe capital shortages or even go bankrupt (Chandler 2022) because of four main reasons: costly infrastructure, high labor cost, low coverage, and unsafe driver behaviors (W. Zhang et al. 2022). Delivery companies have competed for customers in two main ways: being faster or offering large discounts. Companies set up numerous micro-depots near customers and employ many drivers to quickly respond to customer orders and ensure fast, on-time deliveries. However, the setup costs for rent and essential equipment to open micro-depots are quite expensive. Due to the substantial investments and narrow profit margins, these ultra-fast delivery companies struggle to survive once venture capitalists stop funding them (Senzamici 2024). Despite setting up numerous micro-depots, many regions still remain unserved due to a shortage of micro-depot locations. Additionally, customers have a low tolerance for delivery delays, especially when they are provided with an estimated time of arrival (ETA) at the time of placing their orders (Salari, S. Liu, and Shen 2022). Usually, the ETA is calculated based on historical expected travel times, which can sometimes be overly optimistic, as they do not account for real-time traffic and weather conditions. Consequently, this can result in frequent delivery delays and decreased customer satisfaction. In fact, many companies have begun to reconsider the necessity of serving all customers within 15 minutes and attempt to backtrack on their initial delivery promise. For instance, Getir (2022), which initially operated in Turkey and recently expanded its services to Europe and the United States, originally offered deliveries within 15 minutes but extended its delivery time to up to 45 minutes with customer approval. Meanwhile, Gorillas (2022) in Europe initially focused on delivering within 10 minutes but later extended their delivery time to around 60 minutes. Goodfood (2022) in Canada, which aimed to provide fast delivery services within 30 minutes, is discontinuing its on-demand grocery delivery service due to financial struggles (Goodfood 2022).

To help bridge the gap between the theory and practice, we aim to investigate how ultra-fast delivery can be a profitable and reliable business while maintaining high customer service levels that are neither overly optimistic nor pessimistic. In particular, we investigate how different measures of service can lead to distinct levels of cost and cus-

tomer satisfaction. To maintain a high *service level*, the hope is to serve customers within a target *delivery time* (defined as the duration taken for goods to be delivered) with high reliability. Our purpose is to introduce models for the network design of ultra-fast delivery services in the presence of uncertain travel time distributions and unknown time periods when customers place orders. These models aim to maximize the profit while ensuring a certain service level by making the optimal decisions of micro-depot location and customer order allocation. To reach this goal, our paper makes the following contributions.

- We develop probabilistic envelope constrained (PEC) programs for the ultra-fast delivery problem with various service measures, including *period* and *daily* service levels, which focus on equal performance for each period and weighted-average daily performance, respectively. We solve the problem under *partial* and *full* protection of the service level, compare the performance of these measures under different guarantees, and identify the ones that yield a favorable trade-off between the profit and the violation of service level constraints.

- To address the practical challenge that available data may not fully reflect reality, we develop robust programs when both the distribution of travel time and the probability of customers placing orders in different time periods are not explicitly known. We then derive equivalent semi-infinite linear programs and more tractable linear approximations with a finite number of constraints, ensuring both high efficiency and accuracy.

- We carry out extensive experiments on a real-world dataset obtained from Amazon and the Google API, and derive the following insights:

  - There is a trade-off between the profitability and reliability of ultra-fast delivery. A shorter delivery time promise results in higher demand and increased profit, but at the cost of more frequent violations of on-time delivery.

  - The robust formulation yields better out-of-sample performance, evident from its lower probability of violating the target delivery time and smaller devia-

tions from the target. This, in turn, promotes safer decision-making in scenarios with limited data. Although it does entail a slight reduction in profits, this trade-off could be deemed acceptable in light of the improved reliability of timely delivery.

- The daily service level with multi-layer partial protection on the promised delivery times outperforms other strategies overall due to its higher profitability and reliability. This approach prioritizes time periods with higher order frequencies, ensuring that delivery targets are more effectively met during peak demand periods. Additionally, setting hierarchical delivery targets, each associated with a corresponding probability of achieving those targets, provides a more flexible and reliable approach to managing deliveries, helping ultra-fast delivery companies run a profitable business while maintaining high service levels.
- Compared to urban areas, providing ultra-fast delivery services in rural areas, where customers are more dispersed, is more challenging. This is due to the longer distances between delivery locations and the necessity of setting up more micro-depots in rural regions.

The rest of the paper is organized as follows. We review the related work in Section 2.2, and then introduce the ultra-fast delivery design problem in Section 2.3. Next, we present stochastic programming models and their equivalent reformulations in Section 2.4. In Section 2.5, we report the results of numerical studies using real-world datasets to evaluate the effectiveness of our proposed models. Finally, we conclude with managerial insights in Section 3.6.

2.2 Literature Review

In this section, we review the main studies relevant to our research from three points of view: facility location, ultra-fast delivery, and robust chance constraint programming.

2.2.1 Facility Location

The network design of ultra-fast delivery services can be seen as a variant of the Facility Location Problem (FLP), which is a well-known optimization problem in operations research and has been widely studied (e.g. Aikens 1985; Verter 2011). The FLP aims to determine the optimal placement of facilities such as stores, warehouses, factories, hospitals, and schools while satisfying the customer demand, in order to minimize the cost or maximize the profit. Numerous studies focusing on the FLP and its variants have taken into account various forms of uncertainty in demand (e.g. Laporte, Louveaux, and Hamme 1994), risk of facility failure (e.g. Shen, Zhan, and J. Zhang 2011; Cheng, Adulyasak, and Rousseau 2021), service times at facilities, or travel times between demand points and facilities, leading to stochastic or robust location problems (e.g. Snyder 2006). The stochastic FLP is still a prominent research topic, as researchers explore novel perspectives to model the problem and develop efficient algorithms to improve solution procedures. For example, Y. Li et al. (2022) study the reliable uncapacitated facility location problem, in which facilities are subject to uncertain and correlated disruptions. They propose a cutting-plane algorithm that outperforms the best-known algorithm in the literature for the stochastic problem under independent disruptions, specifically the search and cut algorithm proposed by Aboolian, Cui, and Shen (2013). T. Liu et al. (2022) focus on a broad class of facility location problems in the context of adaptive robust stochastic optimization under state-dependent demand uncertainty, and propose a nested Benders decomposition algorithm to solve the model exactly. Shehadeh (2023) proposes two distributionally robust optimization models for a mobile facility fleet-sizing, routing, and scheduling problem with time-dependent and random demand, and solve the problem using a decomposition-based algorithm.

In contrast to existing studies on stochastic or robust location problems, our study focuses on ensuring timely delivery service to customers under two sources of uncertainty: the travel time from facilities to customers and the time period during which customers will place their orders.

2.2.2 Ultra-fast Delivery

Ultra-fast delivery is a special case of last-mile delivery and is popular in the food and grocery industry, where it has extensively expanded in recent years with the rise of online ordering and delivery applications. Some researchers, such as M. Chen, Hu, and J. Wang (2022) and Feldman, Frazelle, and Swinney (2023), investigate the revenue allocation between the restaurant and the food delivery platform and propose practical contracts to improve the profitability of food delivery services. Others propose novel ideas to enhance the efficiency of food delivery services. For example, Cao and Qi (2023) propose the idea of selling grocery in public spaces with wheeled stalls (i.e., self-driving mini grocery stores) to facilitate mobility, proximity, and flexibility of grocery delivery by avoiding the "last 100 meters". We share the same goal of providing better service and generating more benefits for food and grocery delivery. However, our perspective differs from theirs as we prioritize providing ultra-fast service.

Travel time is an important performance metric for ultra-fast delivery services. Mak (2022) emphasizes the importance of improving efficiency in city operations and effectively managing fulfillment operations under tight delivery time windows for omnichannel retailers. With a common goal of offering efficient operations and on-time delivery, many researchers also consider delivery time as a key measure in their work. Some researchers aim to estimate travel times accurately to improve the delivery service. Perakis and Roels (2006) investigate the effect of congestion on travel time and derive an analytical travel-time function that integrates traffic dynamics and shock effects. Hildebrandt and Ulmer (2022) present offline and online-offline estimation approaches to estimate arrival times, and find that accurate arrival times not only raise service perception but also improve the overall delivery system by guiding customer selections, effectively resulting in faster deliveries. Other researchers investigate the impact of delivery time and utilize optimization to facilitate fast deliveries. Deshpande and Pendem (2023) provide empirical evidence to show that fast deliveries drive sales by analyzing a mechanism that connects delivery performance to sales through logistics ratings. Fatehi and Wagner (2022)

notice that customers demand faster and cheaper delivery services, and propose a crowd-sourcing optimization model to provide fast and guaranteed delivery services utilizing independent crowd drivers. Reed, Campbell, and Thomas (2022) develop a capacitated autonomous vehicle assisted delivery problem involving the vehicle driving time, person walking time, and package loading time, and demonstrate that autonomous vehicles can help save time for last-mile deliveries. S. Liu, He, and Max Shen (2021) investigate the impact of delivery data on the on-time performance of food delivery service, and develop an order assignment problem with travel-time predictors. Motivated by a large grocery chain store that offers fast on-demand delivery services, S. Liu and Luo (2023) present a finite-horizon stochastic dynamic program for driver dispatching and routing problem where on-time performance is the main target. Among those that utilize optimization theory to foster fast deliveries, some of them also apply stochastic or robust optimization since there are many sources of uncertainty when offering last-mile delivery services (see Fatehi and Wagner 2022; Y. Chen et al. 2022; Mousavi, Bodur, and Roorda 2022; S. Liu, He, and Max Shen 2021; S. Liu and Luo 2023). However, to the best of our knowledge, the only paper that mentions ultra-fast delivery is Kavuk et al. (2022), who propose a real-life application of deep reinforcement learning to address the order dispatching problem of Getir, an ultra-fast delivery company whose goal is to deliver to as many customers as possible within 15 minutes. Their deep reinforcement learning models predict which orders to accept and reject based on the order characteristics such as the estimated delivery time.

Compared to these papers, our work shares the same purpose of facilitating fast deliveries. The difference is that we model it as a network design problem and aim to provide reliable and flexible ultra-fast delivery services by considering various service measures across different levels of protection, by accounting for uncertainties in travel time and order placement periods, and by viewing demand as a variable linked to travel time.

2.2.3 Robust Chance Constraints and Probabilistic Envelope Constraints

A robust chance constraint is a type of constraint in optimization models requiring that a specific condition should be satisfied with a certain probability, even when the underlying probability distribution of the uncertain parameters is not fully known or might vary within certain bounds. Its goal is to create solutions that are robust and reliable when faced with perturbations in the uncertain parameters. Calafiore and Ghaoui (2006) introduce a distributionally robust formulation for chance-constrained linear programs, and propose a model that considers the worst-case distribution of the uncertain parameters instead of assuming a specific distribution. Hanasusanto et al. (2015) investigate joint chance constraints where uncertain parameter distributions are only known to belong to an ambiguity set defined by the mean and support or an upper bound on dispersion, giving rise to pessimistic or optimistic ambiguous chance constraints. Postek et al. (2018) consider a robust optimization problem with ambiguous stochastic constraints, where only the mean and dispersion information of the distribution of the uncertain parameters are known. Ghosal and Wiesemann (2020) study the distributionally robust chance-constrained vehicle routing problem, which assumes that the customer demands follow a probability distribution that is only partially known, and impose chance constraints on the vehicle capacities for all distributions that are deemed plausible in view of the available information.

A robust probabilistic envelope constraint (PEC), also known as a robust first-order stochastic dominance (FSD) constraint, is a generalization of the robust chance constraint. FSD allows a decision-maker to manage risk in an optimization setting by requiring their decision to yield a random outcome which stochastically dominates a reference outcome in the first order. This technique has been investigated in Dentcheva and Ruszczyński (2004), Luedtke (2008), Armbruster and Delage (2015), and Dai et al. (2023). A PEC compensates for a deficiency in chance constraints, which is that the violation magnitude of the bounds can be very large. This is because chance constraints only control the probability of success but provide no control in the event of a failure. Instead, A PEC is

able to bound the uncertainty by restricting both the violation magnitude and probability. Xu, Caramanis, and Mannor (2012) consider the robust optimization problem under probabilistic envelope constraints, show that the problem of requiring different probabilistic guarantees at each level of constraint violation can be reformulated as a semi-infinite optimization problem, and provided conditions that guarantee polynomial-time solvability of the resulting semi-infinite formulation. Peng, Delage, and J. Li (2020) provide a two-stage stochastic programming model for locating emergency medical service (EMS) stations, consider probabilistic envelope constraints to account for the uncertainty in the requests of EMS services, and apply the model to a real-world EMS system to demonstrate its effectiveness in improving the EMS response times. In contrast to these papers, we apply robust PEC to offer speedy and reliable delivery services and jointly optimize the location and allocation decisions and the service level guarantees.

2.3 Network Design Problem for Ultra-fast Delivery

In this section, we define the network design problem for ultra-fast delivery services, derive the demand function that depends on the delivery time, and introduce a deterministic formulation for the problem.

Definition 2 ***The network design problem for ultra-fast delivery*** *(NDP-UD) is a multi-period problem that involves locating micro-depots and determining service quality levels for customer deliveries while maximizing the profit and ensuring reliable on-time delivery services. It accounts for the relationship between demand volume and delivery speed, as well as uncertainties in the distribution of travel times and the probability of customers placing orders in different time periods.*

2.3.1 Notation

Let $(\mathscr{N}, \mathscr{A})$ represent a directed bipartite network, where the node set $\mathscr{N}$ includes the set of customer locations $\mathscr{I}$ and the set of potential micro-depot locations $\mathscr{J}$, and where

the edge set $\mathscr{A}$ contains edges (j,i) from micro-depot j to customer i with travel distance l_{ij} and edges $(0,j)$ from the central depot to micro-depot j with travel distance l_{0j}. We consider a planning horizon of $|\mathscr{T}|$ time periods and assume that the length of each period $t \in \mathscr{T}$ is long enough to travel between nodes. We use boldface letters to denote column vectors. Row vectors are represented using the transpose (superscript T) of the column vectors. To distinguish between the uncertain and deterministic values, we use a superscript $\sim$ for the random variable and a superscript $\wedge$ for the expected value. The notation $\tilde{\boldsymbol{\tau}} \sim \mathscr{F}$ indicates that $\tilde{\boldsymbol{\tau}}$ follows the distribution $\mathscr{F}$, and $\mathscr{F} \in \mathscr{D}$ states that distribution $\mathscr{F}$ resides in an ambiguity set $\mathscr{D}$. To simplify notation, we use $\forall i$, $\forall j$, and $\forall t$ in place of $\forall i \in \mathscr{I}$, $\forall j \in \mathscr{J}$, and $\forall t \in \mathscr{T}$, respectively.

We assume that customer orders are homogeneous. The nominal demand (i.e., the number of potential customers) at location i in period t is $\bar{d}_{it}$, and the revenue obtained by fulfilling per unit demand at customer location i is r_i. The setup cost to open micro-depot j is o_j, and the delivery cost per unit distance for driving is c. The cost of hiring a driver for one period is h, and each driver serves an average of m customers in each period. The delivery time is defined as the duration of delivering the goods. Let $\tilde{s}_{ijt}$ represent the travel time from micro-depot j to customer i in period t, which is the main source of uncertainty in reality due to real-time traffic and unpredictable weather conditions. Let a_{ijt} denote the average order preparation time, which includes the required time for selecting and packing items in each order. The delivery time of serving customer i from micro-depot j in period t is $\tilde{\boldsymbol{\tau}}_{ijt} = \tilde{s}_{ijt} + a_{ijt}$, and we let $\hat{\boldsymbol{\tau}}_{ijt} = \mathbb{E}[\tilde{\boldsymbol{\tau}}_{ijt}]$. The target delivery time is $\bar{\tau}$. We use variable $y_j = 1$ to denote that micro-depot j is open, and $y_j = 0$ otherwise. The variable x_{ijt} takes value 1 if the demand at location i is served by micro-depot j in period t, and 0 otherwise. The variable z_t is the number of drivers needed in period t. A summary of notation is provided in Appendix A.

2.3.2 Demand Function

Customers generally have several options when ordering groceries, and they make their choices by maximizing their utility. Given the demand volume $\bar{d}_{it}$ at location i in period t, we assume that, customers are more likely to choose deliveries that arrive faster when all else factors are equal. We use the Multinomial Logit (MNL) customer choice model to represent the customer behavior and choice probability. The MNL choice model is defined by the following:

(1) The *decision maker* is a customer who chooses a mode of ordering groceries.

(2) The *choice set* contains three options, including the ultra-fast delivery service, the best competitor, and opting out.

(3) The *attributes* include the delivery time and an independent source of randomness. Other features, such as prices, are assumed to be the same for all options, although this assumption can be relaxed if needed.

(4) The *decision rule* is based on the customer utility. The higher the customer utility of an option, the greater the probability of choosing it. The deterministic utility obtained by a customer at location i from placing an order with the ultra-fast delivery service in period t is denoted as V_{it}, and it depends on the ultra-fast delivery time τ^u_{it}. The random part is ε_{it} and is assumed to be independent and identically Gumbel distributed (Talluri, Van Ryzin, and Van Ryzin 2004). Likewise, the deterministic utility derived from placing an order using the competitor's delivery service is denoted as V^c_{it}. This utility depends on the best competitor delivery time τ^c_{it}, with the addition of a random component ε^c_{it}. We thus have the total utilities U_{it} and U^c_{it} as:

$$U_{it} = V_{it} + \varepsilon_{it}, \text{ where } V_{it} = g(\tau^u_{it}) = \beta_0 + \beta_1 \tau^u_{it},$$

$$U^c_{it} = V^c_{it} + \varepsilon^c_{it}, \text{ where } V^c_{it} = g(\tau^c_{it}) = \beta_0 + \beta_1 \tau^c_{it}.$$

The utility of opting out is zero (i.e., $V^o_{it} = 0$). The probability of customers at location i choosing the ultra-fast delivery in period t is:

$$P_{it}(\text{ultra-fast}) = \frac{e^{\mu V_{it}}}{e^{\mu V_{it}} + e^{\mu V^c_{it}} + 1}, \forall i, t,$$

where μ is a strictly positive scaling parameter that affects the level of randomness, and is assumed to be the same for all individuals and alternatives (Ben-Akiva and Bierlaire 1999). We assume that the independence from irrelevant alternatives (IIA) property is satisfied. That is, the relative likelihood of choosing any two options is independent of the presence of other alternatives. As stated by R. Wang (2021), to relax the IIA assumption and allow more flexible substitution within the choice set, some generalizations such as the nested logit model can be applied. We use the MNL model as a showcase to examine the effect of travel time on the demand volume.

Given that the delivery time is contingent on the decision of which micro-depot will serve customers, and that customers base their decisions to place an order on the estimated delivery time presented to them, we further decompose $P_{it}(\text{ultra-fast})$ into $P_{ijt}(\text{ultra-fast})$, i.e., the probability of customers at location i choosing ultra-fast delivery in period t if they are served by micro-depot j. Namely,

$$P_{ijt}(\text{ultra-fast}) = \frac{e^{\mu g(\hat{\tau}_{ijt})}}{e^{\mu g(\hat{\tau}_{ijt})} + e^{\mu g(\tau^c_{it})} + 1}, \forall i, j, t,$$

where the utility of choosing ultra-fast delivery is $g(\hat{\tau}_{ijt}) = \beta_0 + \beta_1 \hat{\tau}_{ijt}$, and where the estimated delivery time displayed to customers is the expected delivery time from micro-depot j to customer location i in period t, $\hat{\tau}_{ijt} = \hat{s}_{ijt} + a_{ijt}$. Under this choice model, the expected demand volume at location i for ultra-fast delivery services served by micro-depot j in period t, d_{ijt}, can be calculated as follows:

$$d_{ijt} = P_{ijt} \bar{d}_{it} x_{ijt} = \frac{e^{\mu g(\hat{\tau}_{ijt})}}{e^{\mu g(\hat{\tau}_{ijt})} + e^{\mu g(\tau^c_{it})} + 1} \bar{d}_{it} x_{ijt}, \forall i, j, t.$$

2.3.3 Deterministic Formulation

In practice, due to the real-time traffic congestion and variable weather conditions, the travel time from a micro-depot to a customer location is uncertain. One way of handling this uncertainty is to measure the average performance, leading to the following deterministic program (DP) for NDP-UD:

$$\text{(DP)} \quad \max_{x,y,d,z} \quad \sum_i \sum_j \sum_t \left(r_i - c\, l_{ij}\right) d_{ijt} - \sum_j \left(o_j + c\, l_{0j}\right) y_j - \sum_t h z_t \qquad (2.1a)$$

$$\text{s.t.} \quad \sum_j x_{ijt} \leq 1, \forall i,t \tag{2.1b}$$

$$x_{ijt} \leq y_j, \forall i,j,t \tag{2.1c}$$

$$d_{ijt} = \frac{e^{\mu g(\hat{\tau}_{ijt})}}{e^{\mu g(\hat{\tau}_{ijt})} + e^{\mu g(\tau^c_{it})} + 1} \bar{d}_{it} x_{ijt}, \forall i,j,t \tag{2.1d}$$

$$x \in \mathscr{X}_{AVG} \tag{2.1e}$$

$$z_t \geq \frac{1}{m} \sum_i \sum_j d_{ijt}, \forall t \tag{2.1f}$$

$$x \in \{0,1\}^{|\mathscr{I}|\times|\mathscr{J}|\times|\mathscr{T}|}, y \in \{0,1\}^{|\mathscr{J}|}, z \in \mathbb{Z}_+^{|\mathscr{J}|}. \tag{2.1g}$$

The objective (2.1a) is to maximize the expected profit, taking into account the revenue generated from all demands, the outbound cost for deliveries from micro-depots to customers, the opening cost of micro-depots, the inbound cost for deliveries from a central depot to micro-depots, and the driver hiring costs across all periods. We assume that one driver can on average serve m customers in each time period, and that if the order is accepted, the duration between the order arrival and the successful assignment to a driver is included in the preparation time. The constraints (2.1b) and (2.1c) require that each customer is served by at most one micro-depot in each period, and that only open micro-depots serve customers. Using the findings in Section 2.3.2, the constraints (2.1d) indicate that the demand is a function of customer utilities on different delivery choices and is contingent upon average travel time.

Definition 3 Average Service Level *is a service policy that ensures on-time delivery for every customer in each period by considering the average delivery time performance:*

$$\mathscr{X}_{AVG} = \left\{ x \in \mathbb{R}^{|\mathscr{I}|\times|\mathscr{J}|\times|\mathscr{T}|} \middle| \sum_j \hat{\tau}_{ijt} x_{ijt} \leq \bar{\tau}, \forall i,t \right\},$$

where $\mathscr{X}_{AVG}$ contains all the allocation solutions that satisfy the average on-time delivery service.

The constraint (2.1e) conveys that the average delivery time of serving each customer in any period should be no later than the target delivery time $\bar{\tau}$. The constraints (2.1f) stipulate that the number of hired drivers in each period must be adequate to fulfill all

orders, under the assumption that the supply of drivers is sufficient. The constraints (2.1g) are domain restrictions. We note that DP is a mixed-integer linear program.

2.4 Probabilistic Envelope Constrained Programs

Bounding only the expected travel time may be too lenient. Therefore, we now present a probabilistic envelope constraint approach, which is an extension of chance constraint programming, to achieve different on-time delivery service levels with different probabilities. We then derive tractable formulations when the travel time distribution is explicitly known or unknown. We define and model the *period service level* with an equal level at each period, and the *daily service level* by considering the average service level throughout the entire day with uncertain frequency of customer orders. Finally, we present a stochastic program for the NDP-UD, which can accommodate different service policies and handle various sources of uncertainty, and also extend the program by jointly optimizing NDP-UD and the service level guarantees to avoid excessive conservatism.

2.4.1 Chance Constraints

The delivery time $\tilde{\tau}_{ijt}$ is a key performance measure of the service level and it is uncertain due to the uncertain travel time. The chance constraint (CC) helps us model the condition that, for every customer served in every period, the uncertain delivery time should be below the target delivery time $\bar{\tau}$ with probability at least $\beta \in [0,1]$. This restriction is represented by the following constraints:

$$\mathbb{P}_{\tilde{\tau}}\left(\tilde{\tau}_{ijt} \leq \bar{\tau}\right) \geq \beta, \quad \forall i,j,t \in \left\{ i \in \mathscr{I}, j \in \mathscr{J}, t \in \mathscr{T} \middle| x_{ijt} = 1 \right\}.$$

Since we have $x \in \{0,1\}$ and $\bar{\tau} \geq 0$, the chance constraint is equivalent to

$$\mathbb{P}_{\tilde{\tau}}\left(\tilde{\tau}_{ijt} x_{ijt} \leq \bar{\tau}\right) \geq \beta, \forall i,j,t.$$

Since $\sum_j x_{ijt} \leq 1$, the chance constraint is also equivalent to

$$\mathbb{P}_{\tilde{\tau}}\left(\sum_j \tilde{\tau}_{ijt} x_{ijt} \leq \bar{\tau}\right) \geq \beta, \forall i,t.$$

2.4.2 Probabilistic Envelope Constraints

A major downside of chance constraints is that they cannot avoid the long tail phenomenon. That is, for the violated cases which might occur with probability $1-\beta$, the magnitude of the violation could be very large. To deal with this issue, we use the probabilistic envelope constraint (PEC) to bound the uncertain delivery time by restricting both the probability and the degree of violation.

Compared to the chance constraint that guarantees a good delivery service at one specific level, the PEC ensures that the customer satisfaction is protected at several levels under the uncertain delivery time. For instance, to guarantee ultra-fast delivery, the retailer may require that any order should be delivered within 10 minutes with probability at least 70%, within 30 minutes with probability at least 80%, and within one hour with probability at least 99%. Some violations are allowed on the initial target (i.e., 10 minutes), but for different magnitude (i.e., 20 minutes and 50 minutes), the probability of the violation (i.e., 20% and 1%) is bounded. Define the magnitude of the violation as v, and the probability of satisfying the new target $\bar{\tau}+v$ as $\beta(v)$. For each customer i served by any micro-depot in each period t, for any non-negative v, the uncertain delivery time should be below $\bar{\tau}+v$ with probability at least $\beta(v)$. The probabilistic envelope constraint is

$$\textbf{PEC: } \mathbb{P}_{\tilde{\tau}}\left(\sum_{j}\tilde{\tau}_{ijt}x_{ijt}\leq\bar{\tau}+v\right)\geq\beta(v),\forall i,t,\forall v\geq 0, \tag{2.2}$$

where $\beta:\mathbb{R}^{+}\rightarrow[0,1]$, and $\beta(v)$ is a non-decreasing continuous function in v.

Definition 4 Period Service Level *is a service policy that ensures on-time delivery for every customer in each period and guarantees a certain level of reliability for every possible delivery time:*

$$\mathscr{X}_{PEC}:=\left\{x\in\mathbb{R}^{|\mathscr{I}|\times|\mathscr{J}|\times|\mathscr{T}|}\middle|\mathbb{P}_{\tilde{\tau}}\left\{\sum_{j}\tilde{\tau}_{ijt}x_{ijt}\leq\bar{\tau}+v\right\}\geq\beta(v),\forall i,t,\forall v\geq 0\right\}. \tag{2.3}$$

In other words, the set $\mathscr{X}_{PEC}$ contains all the allocation solutions that satisfy PEC (2.2).

Example 1 *Suppose that* $\beta(v) := 1/(\frac{\gamma}{v+\alpha}+1), v \geq 0$ *with nonnegative* γ *and strictly positive* α*. The inverse function of* $\beta(\cdot)$ *is* $\beta^{-1}(p) = \gamma/(\frac{1}{p}-1) - \alpha$*, for* $\frac{\alpha}{\gamma+\alpha} < p < 1$*. See Figure 2.1 for an illustration of the* $\beta(\cdot)$ *function for selected sample* α *and* γ *values.*

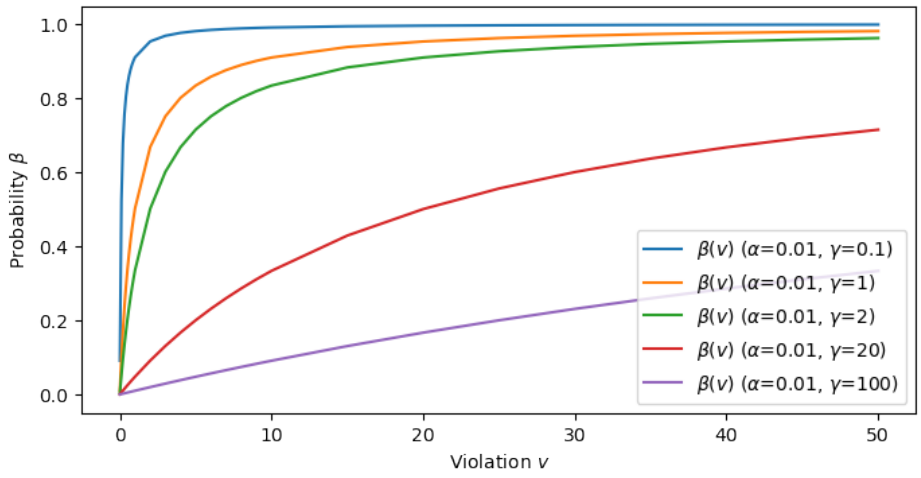


Figure 2.1 – $\beta(v)$ envelope for selected sample α and γ values.

Given a specific value of $\bar{v}$, the delivery time of any order should not exceed $\bar{\tau}+\bar{v}$ with probability at least $\beta(\bar{v})$. In this case, the constraint implies a single chance constraint. Therefore, PEC represents a stronger constraint than CC.

Definition 5 Period Service Level with One-Layer Guarantee *is a service policy that guarantees on-time delivery for a specific delivery time:*

$$\mathscr{X}_{CC}(\bar{v}) := \left\{ x \in \mathbb{R}^{|\mathscr{I}|\times|\mathscr{J}|\times|\mathscr{T}|} \middle| \mathbb{P}_{\tilde{\tau}}\left(\sum_j \tilde{\tau}_{ijt} x_{ijt} \leq \bar{\tau}+\bar{v} \right) \geq \beta(\bar{v}), \forall i,t \right\},$$

where $\bar{v}$ *is a given value. The set* $\mathscr{X}_{CC}$ *contains all the allocation solutions that provide on-time delivery service within* $\bar{\tau}+\bar{v}$ *minutes with probability at least* $\beta(\bar{v})$*.*

2.4.2.1 Reformulation with Known Distribution.

One can assume that the randomness of the travel time follows a known distribution $\mathscr{F}$ and obtain a tractable reformulation of $\mathscr{X}_{PEC}$.

Proposition 4 *If uncertainty $\tilde{\tau}$ follows a known distribution $\mathscr{F}$, $\mathscr{X}_{PEC}$ can be reformulated as*

$$\mathscr{X}_{PEC} = \left\{ x \in \mathbb{R}^{|\mathscr{I}|\times|\mathscr{J}|\times|\mathscr{T}|} \middle| x_{ijt} \leq \Theta_{ijt}, \forall i,j,t \right\}, \tag{2.4}$$

where $\Theta_{ijt} := \mathbb{I}\left\{ \sup_{v\geq 0} \left(\Psi^{-1}_{\tilde{\tau}_{ijt}}(\beta(v)) - \bar{\tau} - v \right) \leq 0 \right\}$, $\mathbb{I}\{\cdot\}$ *is the indicator function,* $\Psi_{\tilde{\tau}_{ijt}}$ *is the cumulative probability function of* $\tilde{\tau}_{ijt}$*, and* $\Psi^{-1}_{\tilde{\tau}_{ijt}}(\beta)$ *is its quantile at probability* β*.*

The proof is presented in Appendix B.1.

Remark 3 *While $\mathscr{X}_{PEC}$ only imposes an upper bound on x, calculating this bound requires evaluations of a supremum over $v \in \mathbb{R}^+$. Fortunately, one can exploit a piecewise constant approximation of $\beta(\cdot)$.*

For any $\beta(v)$, we can derive an outer and inner approximation of $\beta(v)$:

$$\beta^{outer}(v) = \sum_{k=1}^{|\mathscr{K}|} \beta(v^{k+1}) \mathbb{I}\left\{ v \in [v^k, v^{k+1}[\right\} \tag{2.5a}$$

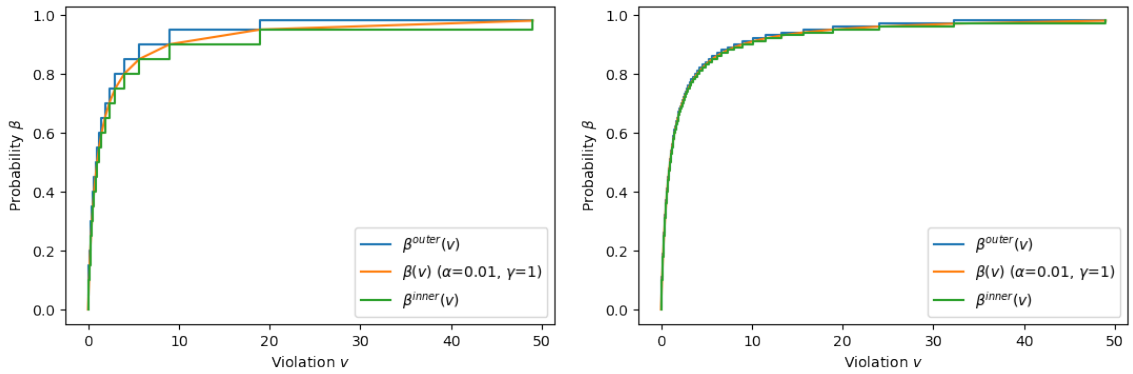
$$\beta^{inner}(v) = \sum_{k=1}^{|\mathscr{K}|} \beta(v^{k}) \mathbb{I}\left\{ v \in [v^k, v^{k+1}[\right\}, \tag{2.5b}$$

where $\{v^k\}_{k\in\mathscr{K}}$ is a discretization of $[0,\infty)$ and $\mathscr{K} = \{1,2,...,|\mathscr{K}|\}$.

As shown in Figure 2.2, $\beta^{outer}(v)$ and $\beta^{inner}(v)$ are step functions under a finite number of steps $k \in \mathscr{K}$. A smaller step size represents a larger number of steps $|\mathscr{K}|$, and leads to tighter approximations. Compared to $\beta(v)$, $\beta^{outer}(v)$ yields a smaller feasible set for x by requiring a higher probability of meeting the target, while $\beta^{inner}(v)$ yields a larger feasible set by requiring a lower probability of meeting the target (i.e., $\beta^{outer}(v) \geq \beta(v) \geq \beta^{inner}(v), \forall v \geq 0$).

Corollary 1 *When $\beta(v)$ is approximated by its outer step function* (2.5a) *and inner step function*(2.5b)*, the value of the indicator function on the right hand side is known, leading to the approximated reformulation of $\mathscr{X}_{PEC}$ with a finite number of linear constraints, as follows:*

$$\mathscr{X}^{outer}_{PEC} \subseteq \mathscr{X}_{PEC} \subseteq \mathscr{X}^{inner}_{PEC}$$



(a) $|\mathscr{K}| = 20$ with the step size $\beta = 0.05$. (b) $|\mathscr{K}| = 100$ with the step size $\beta = 0.01$.

Figure 2.2 – Inner and outer approximations of $\beta(v)$.

with

$$\mathscr{X}_{PEC}^{inner} := \left\{ x \in \mathbb{R}^{|\mathscr{I}|\times|\mathscr{J}|\times|\mathscr{T}|} \middle| x_{ijt} \leq \Theta_{ijt}^{inner}, \forall i,j,t \right\}, \tag{2.6}$$

$$\mathscr{X}_{PEC}^{outer} := \left\{ x \in \mathbb{R}^{|\mathscr{I}|\times|\mathscr{J}|\times|\mathscr{T}|} \middle| x_{ijt} \leq \Theta_{ijt}^{outer}, \forall i,j,t \right\}, \tag{2.7}$$

where $\Theta_{ijt}^{inner} := \min_k \mathbb{I}\left\{ \Psi_{\tilde{\tau}_{ijt}}^{-1}(\beta(v^k)) - \bar{\tau} - v^k \leq 0 \right\}$,
and $\Theta_{ijt}^{outer} := \min_k \mathbb{I}\left\{ \Psi_{\tilde{\tau}_{ijt}}^{-1}(\beta(v^{k+1})) - \bar{\tau} - v^{k+1} \leq 0 \right\}$.

2.4.2.2 Reformulation with Unknown Distribution.

Under the case where the exact distribution of travel time may not be explicitly known, we introduce the robust PEC:

$$\textbf{Robust PEC: } \inf_{F \in \mathscr{D}} \mathbb{P}_{\tilde{\tau} \sim \mathscr{F}} \left(\sum_j \tilde{\tau}_{ijt} x_{ijt} \leq \bar{\tau} + v \right) \geq \beta(v), \forall i,t,\forall v \geq 0, \tag{2.8}$$

where $\mathscr{D}$ is the ambiguity set containing the true distribution.

Assumption 1 *We consider that the distribution of travel times is unknown, but partial information such as moments can be obtained from the dataset. In this case, the ambiguity set* $\mathscr{D}$ *represents a family of distributions whose mean and covariance information are given:*

$$\mathscr{D} := \left\{ \mathscr{F} \mid \tilde{\tau} = \hat{\tau} + \tilde{\delta},\ \mathbb{E}_{\mathscr{F}}\left[\tilde{\delta}_t\right] = 0,\ \mathbb{E}_{\mathscr{F}}\left[\tilde{\delta}\tilde{\delta}^T\right] = \Sigma \right\}.$$

Let $x \in \mathscr{X}_{R-PEC}$ be the solutions that satisfy the robust PEC (2.8). With the ambiguity set $\mathscr{D}$,

$$\mathscr{X}_{R-PEC} := \left\{ x \in \mathbb{R}^{|\mathscr{I}|\times|\mathscr{J}|\times|\mathscr{T}|} \middle| \inf_{\tilde{\delta}_{it}\sim(0,\Sigma_{it})} \mathbb{P}\left\{ \left(\hat{\tau}_{it}+\tilde{\delta}_{it}\right)^T x_{it} \leq \bar{\tau}+v \right\} \geq \beta(v), \forall i,t,\forall v \geq 0 \right\} \quad (2.9)$$

where $\tilde{\delta}_{it} \sim (0,\Sigma_{it})$ considers all the random vectors $\tilde{\delta}_{it} \in \mathbb{R}^{|\mathscr{J}|}$ with mean 0 and covariance Σ_{it} such that $[\Sigma_{it}]_{j_1,j_2} = [\Sigma]_{(i,j_1,t)(i,j_2,t)}$.

Remark 4 *The NDP-UD with $x \in \mathscr{X}_{R-PEC}$ is a semi-infinite program with an infinite number of constraints, since the constraint has to be satisfied under any distribution in ambiguity set $\mathscr{D}$ and for any v.*

Similar to Calafiore and Ghaoui (2006) and Xu, Caramanis, and Mannor (2012), who derived an equivalent and tractable reformulation for the robust CC and PEC, respectively, we present the following result.

Lemma 1 *$\mathscr{X}_{R-PEC}$ can be equivalently reformulated as follows:*

$$\mathscr{X}_{R-PEC} = \left\{ x \in \mathbb{R}^{|\mathscr{I}|\times|\mathscr{J}|\times|\mathscr{T}|} \middle| \hat{\tau}_{it}^T x_{it} + \sqrt{\frac{\beta(v)}{1-\beta(v)}} \sqrt{x_{it}^T \Sigma_{it} x_{it}} \leq \bar{\tau}+v, \forall i,t,\forall v \geq 0 \right\} \quad (2.10)$$

Proposition 5 *$\mathscr{X}_{R-PEC}$ has an equivalent linear reformulation*

$$\mathscr{X}_{R-PEC} = \left\{ x \in \mathbb{R}^{|\mathscr{I}|\times|\mathscr{J}|\times|\mathscr{T}|} \middle| x_{ijt} \leq \Theta_{ijt}, \forall i,j,t \right\}, \quad (2.11)$$

where $\Theta_{ijt} = \mathbb{I}\left\{ \sup_{v\geq 0} \hat{\tau}_{ijt} + \sqrt{\frac{\beta(v)}{1-\beta(v)}}\sigma_{ijt} - \bar{\tau} - v \leq 0 \right\}$. Specifically, in the case defined in Example 1 that $\beta(v) = \frac{1}{\frac{\gamma}{v+\alpha}+1}$, we have $\Theta_{ijt} = \mathbb{I}\left\{ \hat{\tau}_{ijt} + \alpha + \frac{\sigma_{ijt}^2}{4\gamma} - \bar{\tau} \leq 0 \right\}$.

The proof is presented in Appendix B.2. The outer and inner approximations of $\mathscr{X}_{R-PEC}$ with discretized v are provided in Appendix C.1.

2.4.3 Probabilistic Envelope Constraints with Two Forms of Uncertainty

In practical scenarios, customers may order more frequently during lunchtime and dinnertime, and less frequently in the early morning or late at night. Instead of providing an equal service level in each period, we can evaluate the overall daily service level and prioritize those time periods with higher order frequencies. Consequently, it becomes essential to consider the probability distribution of time periods during which orders are placed and to ensure a certain service level across all periods within the entire day.

For each customer i served by any micro-depot j, the uncertain delivery time under uncertain period $\tilde{t}$ should be no more than $\bar{\tau}+v$ with probability at least $\beta(v)$. The probabilistic envelope constraint with period uncertainty (PECP) is

$$\textbf{PECP: } \mathbb{P}_{\tilde{\tau},\tilde{t}}\left(\sum_j \tilde{\tau}_{ij\tilde{t}}\, x_{ij\tilde{t}} \leq \bar{\tau}+v \middle| \sum_j x_{ij\tilde{t}} = 1\right) \geq \beta(v), \forall i, \forall v \geq 0. \tag{2.12}$$

Definition 6 Daily Service Level *is a service policy that ensures on-time delivery service for each customer throughout the entire day and guarantees a certain reliability for every possible delivery time:*

$$\mathscr{X}_{PECP} := \left\{ x \in \mathbb{R}^{|\mathscr{I}|\times|\mathscr{J}|\times|\mathscr{T}|} \middle| \begin{array}{l} \mathbb{P}_{\tilde{\tau},\tilde{t}}\left(\sum_j \tilde{\tau}_{ij\tilde{t}}\, x_{ij\tilde{t}} \leq \bar{\tau}+v \middle| \sum_j x_{ij\tilde{t}} = 1\right) \geq \beta(v), \\ \forall i : \mathbb{P}\left(\sum_j x_{ij\tilde{t}} = 1\right) > 0, \forall v \geq 0 \end{array} \right\}. \tag{2.13}$$

The set $\mathscr{X}_{PECP}$ contains all the allocation solutions that satisfy PECP (2.12).

2.4.3.1 Reformulation with Known Distribution.

Similar to Section 2.4.2.1, we assume full knowledge of distribution of travel time from micro-depots to customers. Additionally, we consider a finite number of periods in which each customer places orders with certain probabilities. We now reformulate $\mathscr{X}_{PECP}$ into a tractable formulation.

Proposition 6 *Consider a finite number of periods $t \in \mathscr{T}$. In each period t, customer i places an order with known probability q_{it}. If the uncertainty $\tilde{\tau}_{ijt}$ follows a known*

distribution $\mathscr{F}$, we reformulate $\mathscr{X}_{PECP}$ into

$$\mathscr{X}_{PECP} = \left\{ x \in \mathbb{R}^{|\mathscr{I}|\times|\mathscr{J}|\times|\mathscr{T}|} \middle| \sum_t q_{it} \left(\sum_j \left[\Psi_{\tilde{\tau}_{ijt}}(\bar{\tau}+v) - \beta(v) \right] x_{ijt} \right) \geq 0, \forall i, \forall v \geq 0 \right\} \quad (2.14)$$

where $\Psi_{\tilde{\tau}_{ijt}}$ is the cumulative probability function of $\tilde{\tau}_{ijt}$.

The proof is presented in Appendix B.3. This formulation states that for each customer i, the weighted-average difference between the realized frequency and promised frequency is non-negative. The outer and inner approximations of $\mathscr{X}_{PECP}$ are provided in Appendix C.2.

2.4.3.2 Reformulation with Unknown Distribution.

A second interesting case is when both the travel time distribution and the probability of customers placing orders in each period are unknown. In this case, we deal with the robust PECP.

$$\textbf{Robust PECP: } \inf_{q_i \in \mathscr{Q}_i} \inf_{\left\{\tilde{\delta}_{it} \sim (0,\Sigma_{it})\right\}_{t=1}^{|\mathscr{T}|}} \mathbb{P}_{\tilde{t} \sim q} \left\{ \left(\hat{\tau}_{i\tilde{t}} + \tilde{\delta}_{i\tilde{t}} \right)^T x_{i\tilde{t}} \leq \bar{\tau} + v \right\} \geq \beta(v), \forall i, \forall v \geq 0, \quad (2.15)$$

where $\mathscr{Q}_i \subseteq \Delta^{|\mathscr{T}|}$, the probability simplex in $\mathbb{R}^{|\mathscr{T}|}$.

Let $\mathscr{X}_{R-PECP}$ be the set of solutions that satisfy the robust PECP, we have

$$\mathscr{X}_{R-PECP} := \left\{ x \in \mathbb{R}^{|\mathscr{I}|\times|\mathscr{J}|\times|\mathscr{T}|} \middle| \inf_{q_i \in \mathscr{Q}_i} \sum_t q_{it} \left(\sum_j \left[\Upsilon_{ijt}(v) - \beta(v) \right] x_{ijt} \right) \geq 0, \forall i, \forall v \geq 0 \right\},$$

where $\Upsilon_{ijt}(v) = \inf_{\tilde{\delta}_{ijt} \sim (0,\sigma^2_{ijt})} \mathbb{P}\left\{ \hat{\tau}_{ijt} + \tilde{\delta}_{ijt} \leq \bar{\tau} + v \right\}$. Now, the computational challenge comes from two parts: the uncertainty set $\mathscr{Q}_i$ and $\Upsilon_{ijt}(v)$. To handle $\mathscr{Q}_i$, we make the following assumption.

Assumption 2 *The uncertainty about q_i is captured by*

$$\mathscr{Q}_i := \left\{ q_i \in \mathbb{R}^{|\mathscr{T}|} \mid q_i^T e = 1,\ 0 \leq q_i \leq 1,\ \left\| \Sigma_{q_i}^{-\frac{1}{2}} (q_i - \hat{q}_i) \right\|_1 \leq \Gamma \right\},$$

where $\hat{q}_i$ is the center of the uncertainty set, Σ_{q_i} defines the shape of the set, and Γ is the radius.

Proposition 7 *If Assumption 1 and Assumption 2 are satisfied, $\mathscr{X}_{R-PECP}$ has an equivalent semi-infinite linear reformulation*

$$\mathscr{X}_{R-PECP} = \left\{ x \in \mathbb{R}^{|\mathscr{I}|\times|\mathscr{J}|\times|\mathscr{T}|} \;\middle|\; \begin{array}{ll} \forall v \geq 0, & \exists u_1 \in \mathbb{R}^{|\mathscr{I}|\times|\mathscr{T}|}, \theta_1 \in \mathbb{R}^{|\mathscr{I}|}, \theta_2 \in \mathbb{R}^{|\mathscr{I}|} \\ & \hat{q}_i^T u_{1i} + \Gamma\theta_{1i} + \theta_{2i} \leq 0, \forall i \\ & u_{1it} + \theta_{2i} \geq \beta(v) x_{it}^T I - x_{it}^T \Upsilon_{it}(v), \forall i,t \\ & \theta_{1i} \geq u_{1i}^T [\Sigma_{q_i}^{\frac{1}{2}}]_t, \forall i,t \\ & \theta_{1i} \geq -u_{1i}^T [\Sigma_{q_i}^{\frac{1}{2}}]_t, \forall i,t \end{array} \right\}, \tag{2.16}$$

where θ_1, θ_2, u_1 are dependent on v, $[\Sigma_{q_i}^{\frac{1}{2}}]_t$ is the t^{th} column of the matrix $\Sigma_{q_i}^{\frac{1}{2}}$, and $[\Upsilon_{it}(v)]_j = \frac{(\bar{\tau}+v-\hat{\tau}_{ijt})_+^2}{(\bar{\tau}+v-\hat{\tau}_{ijt})_+^2+\sigma_{ijt}^2}$ with $(y)_+ = \max(0,y)$.

Note that $\Upsilon_{it}(v)$ can be preprocessed and taken as a fixed value. The proof is presented in Appendix B.4. The outer and inner approximations of $\mathscr{X}_{R-PECP}$ are provided in Appendix C.3.

Remark 5 *When $\Gamma = 0$ and $\Sigma_{q_i} > 0$, the last constraint in the uncertainty set $\mathscr{Q}_i$ states that q_i is explicitly known and equal to $\hat{q}_i$ (i.e., $\mathscr{Q}_i := \{\hat{q}_i\}$). In this case, $\mathscr{X}_{R-PECP}$ is reduced to $\mathscr{X}_{R-PECP}$ only with uncertain travel time distribution:*

$$\mathscr{X}_{R-PECP_T} := \left\{ x \in \mathbb{R}^{|\mathscr{I}|\times|\mathscr{J}|\times|\mathscr{T}|} \;\middle|\; \sum_t \hat{q}_{it} \left(\sum_j \left[\Upsilon_{ijt}(v) - \beta(v) \right] x_{ijt} \right) \geq 0, \forall i, \forall v \geq 0 \right\} \tag{2.17}$$

where $\Upsilon_{ijt}(v) = \frac{(\bar{\tau}+v-\hat{\tau}_{ijt})_+^2}{(\bar{\tau}+v-\hat{\tau}_{ijt})_+^2+\sigma_{ijt}^2}$.

Remark 6 *When Γ is a large value that makes the uncertainty set large enough to cover any possible distribution of q_i, the last constraint in uncertainty set $\mathscr{Q}_i$ becomes redundant. For example, if Σ_{q_i} is diagonal, the lowest upper bound of Γ is $\max_i \sum_t \max\left\{ [\Sigma_{q_i}^{-\frac{1}{2}}]_{tt}(1-\hat{q}_{it}), [\Sigma_{q_i}^{-\frac{1}{2}}]_{tt}\hat{q}_{it} \right\}$. Intuitively, if Γ is large enough to cover the furthest node from the average value in terms of standard deviations, the robust PECP is reduced to robust PEC.*

Remark 7 *If the delivery time follows a known distribution, but the probability of placing orders in each period is uncertain, $\mathscr{X}_{R-PECP}$ is reduced to $\mathscr{X}_{R-PECP_P}$ only with uncertain period probability, which has the following equivalent linear reformulation:*

$$\mathscr{X}_{R-PECP_P} := \left\{ x \in \mathbb{R}^{|\mathscr{I}|\times|\mathscr{J}|\times|\mathscr{T}|} \;\middle|\; \begin{array}{ll} \forall v \geq 0, & u_1 \in \mathbb{R}^{|\mathscr{I}|\times|\mathscr{T}|}, \theta_1 \in \mathbb{R}^{|\mathscr{I}|}, \theta_2 \in \mathbb{R}^{|\mathscr{I}|} \\ & \hat{q}_i^T u_{1i} + \Gamma\theta_{1i} + \theta_{2i} \leq 0, \forall i \\ & u_{1it} + \theta_{2i} \geq \beta(v) x_{it}^T I - x_{it}^T \, \Psi_{it}(v), \forall i,t \\ & \theta_{1i} \geq u_{1i}^T [\Sigma_{q_i}^{\frac{1}{2}}]_t, \forall i,t \\ & \theta_{1i} \geq -u_{1i}^T [\Sigma_{q_i}^{\frac{1}{2}}]_t, \forall i,t \end{array} \right\},$$

where θ_1, θ_2, u_1 are dependent on v, and $[\Psi_{it}(v)]_j$ is the cumulative probability function of $\tilde{\delta}_{ijt}$.

2.4.4 Stochastic Program and Linear Reformulation

If the daily service level is applied, the stochastic program under the uncertainty of the travel time distribution and period probability is

$$(\mathrm{SP}_1) \max_{x,y,d,z} \quad \sum_i \sum_j \sum_t \left(r_i - cl_{ij}\right) d_{ijt} - \sum_j \left(o_j + cl_{0j}\right) y_j - \sum_t h z_t \tag{2.18a}$$

$$\text{s.t.} \quad (2.1\text{b}) - (2.1\text{d}), (2.1\text{f}) - (2.1\text{g})$$

$$x \in \mathscr{X}, \tag{2.18b}$$

where $\mathscr{X}$ can be any one of the following sets: $\mathscr{X}_{CC}$, $\mathscr{X}_{PEC}$, $\mathscr{X}_{R-PEC}$, $\mathscr{X}_{PECP}$, or $\mathscr{X}_{R-PECP}$. The objective is the maximization of the expected profit. The location and allocation decisions are made to reach a certain service level that depends on $\mathscr{X}$, including the period service level related to $\mathscr{X}_{PEC}$, daily service level related to $\mathscr{X}_{PECP}$, and their variants. The computational challenge arises from the constraint (2.18b), which can be reformulated as an equivalent semi-infinite linear program based on the linear reformulations presented in Propositions 4 to 7. Furthermore, it can be approximated by a mixed-integer linear program (MILP) with a finite number of constraints using the outer and inner approximations provided in Corollary 1 and Appendix C. To rephrase,

$\mathscr{X}^{outer} \subseteq \mathscr{X} \subseteq \mathscr{X}^{inner}$. Take $\mathscr{X}_{R-PECP}$ as an example, we have the following formulation SP_1^R, which is an approximation of SP_1:

$$
\begin{aligned}
(\text{SP}_1^R) \max_{x,y,d,z,u,\theta} \quad & \sum_i \sum_j \sum_t \left(r_i - cl_{ij}\right) d_{ijt} - \sum_j \left(o_j + cl_{0j}\right) y_j - \sum_t h z_t && \text{(2.19a)} \\
\text{s.t.} \quad & (2.1\text{b}) - (2.1\text{d}), (2.1\text{f}) - (2.1\text{g}) \\
& \sum_t \hat{q}_{it} u_{1it}^k + \Gamma \theta_{1i}^k + \theta_{2i}^k \leq 0, \forall i,k && \text{(2.19b)} \\
& u_{1it}^k + \theta_{2i}^k \geq \sum_j \left[\beta(v^{k+\varepsilon}) - \Upsilon_{ijt}(v^k)\right] x_{ijt}, \forall i,t,k && \text{(2.19c)} \\
& \theta_{1i}^k \geq \sum_{t'} (u_{1it'}^k)(\Sigma_{q_i})_{tt'}^{\frac{1}{2}}, \forall i,t,k && \text{(2.19d)} \\
& \theta_{1i}^k \geq -\sum_{t'} (u_{1it'}^k)(\Sigma_{q_i})_{tt'}^{\frac{1}{2}}, \forall i,t,k && \text{(2.19e)} \\
& \Upsilon_{ijt}(v^k) = \frac{(\bar{\tau} + v^k - \hat{\tau}_{ijt})_+^2}{(\bar{\tau} + v^k - \hat{\tau}_{ijt})_+^2 + \sigma_{ijt}^2}, \forall i,j,t,k. && \text{(2.19f)}
\end{aligned}
$$

SP_1^R provides a relaxation or restriction of SP_1 depending on whether $\varepsilon = 0$ or 1, respectively.

2.4.5 Stochastic Program with Optimized PEC and Linear Reformulation

In the chance constraint $\mathbb{P}_{\tilde{\tau}}\left(\sum_j \tilde{\tau}_{ijt} x_{ijt} \leq \bar{\tau} + \bar{v}\right) \geq \beta(\bar{v})$, target $\bar{\tau} + \bar{v}$ being reached with probability at least $\beta(\bar{v})$ may lead to a high degree of violation on target or lead to a low profit, depending on the value of $\bar{v}$ and the shape of the $\beta(\cdot)$ function. To obtain a better service level with a lower violation on target, we proposed model SP_1, where the service level has been fully protected on any possible violations. However, such restrictive requirements could be too conservative in practice, inspiring us to jointly optimize the service level along with the decisions. This optimization aims to ensure not only a good service level but also a decent profit. To be specific, any set $\mathscr{X}$ containing v (i.e., $\mathscr{X}_{PEC}$, $\mathscr{X}_{R-PEC}$, $\mathscr{X}_{PECP}$, or $\mathscr{X}_{R-PECP}$) can be considered as a variant $\mathscr{X}(\underline{v})$ that depends on $\underline{v}$. In particular, for any $\underline{v} \geq 0$,

$\mathscr{X}_{R-PECP}(\underline{v}) := \left\{ x \in \mathbb{R}^{|\mathscr{I}|\times|\mathscr{J}|\times|\mathscr{T}|} \middle| \inf_{q_i \in \mathscr{Q}_i} \sum_t q_{it} \left(\sum_j \left[\Upsilon_{ijt}(v) - \beta(v) \right] x_{ijt} \right) \geq 0, \forall i, \forall v \geq \underline{v} \right\}$.
Other sets are similarly defined. In this case, protections are imposed on any $v \geq \underline{v}$ instead of $v \geq 0$, and $\underline{v}$ is considered as a decision variable to find the optimal service level guarantees.

$$
\begin{aligned}
(\mathrm{SP}_2) \max_{x,y,d,z,\underline{v}} \quad & \sum_i \sum_j \sum_t \left(r_i - cl_{ij} \right) d_{ijt} - \sum_j \left(o_j + cl_{0j} \right) y_j - \sum_t h z_t && \text{(2.20a)} \\
\text{s.t.} \quad & (2.1\text{b}) - (2.1\text{d}), (2.1\text{f}) - (2.1\text{g}) \\
& x \in \mathscr{X}(\underline{v}), \forall \underline{v} \geq 0, && \text{(2.20b)}
\end{aligned}
$$

where $\mathscr{X}(\underline{v})$ can be $\mathscr{X}_{PEC}(\underline{v})$, $\mathscr{X}_{R-PEC}(\underline{v})$, $\mathscr{X}_{PECP}(\underline{v})$, or $\mathscr{X}_{R-PECP}(\underline{v})$. We then discretize $\underline{v}$ into finite steps and find the optimal steps that yield the maximum profit while maintaining a certain service level. Take $\mathscr{X}_{R-PECP}(\underline{v})$ as an example, the stochastic program can be reformulated into

$$
\begin{aligned}
(\mathrm{SP}_2^R) \max_{x,y,d,z,u,\theta} \quad & \sum_i \sum_j \sum_t \left(r_i - cl_{ij} \right) d_{ijt} - \sum_j \left(o_j + cl_{0j} \right) y_j - \sum_t h z_t && \text{(2.21a)} \\
\text{s.t.} \quad & (2.1\text{b}) - (2.1\text{d}), (2.1\text{f}) - (2.1\text{g}), (2.19\text{c}) - (2.19\text{f}) \\
& \sum_t \hat{q}_{it} u_{1it}^k + \Gamma \theta_{1i}^k + \theta_{2i}^k \leq 0, \forall i, \forall k \in [|\mathscr{K}| + 1 - n, |\mathscr{K}|], && \text{(2.21b)}
\end{aligned}
$$

where $n \in [0, |\mathscr{K}|]$ is the number of the to-be-guaranteed service levels, and $|\mathscr{K}|$ is the total number of steps in the step function of $\beta(v)$. When $n = |\mathscr{K}|$, the constraints (2.21b) are imposed for all service levels. If $n = 0$, the constraints can be interpreted in the way that our objective is to serve all the customers without restricting the delivery time. The constraints (2.21b) specify that the service level is implemented starting from serving customers within a long delivery duration $\bar{\tau} + v^{|\mathscr{K}|}$, which is defined as a low service level; and ending with serving customers within a short duration $\bar{\tau}$, which is defined as a high service level. If the higher service level is achieved (e.g. $k = |\mathscr{K}| - 1$), the lower one has to be satisfied (e.g. $k = |\mathscr{K}|$). The larger the number of the guaranteed levels, the shorter the target delivery duration. Other formulations for SP_1 and SP_2 under different scenarios for uncertainty are presented in Appendix D.

2.5 Numerical Study

In this section, we first introduce the real-world dataset, the performance metrics, and the implementation details. We then evaluate the performance of β approximation functions and compare formulations under different service levels and uncertainties, including the period and daily service levels, the full, partial and one-layer protection, and the robust and non-robust models. We also investigate the impact of different factors and finally analyze the trade-off between the profitability and reliability for urban and rural areas.

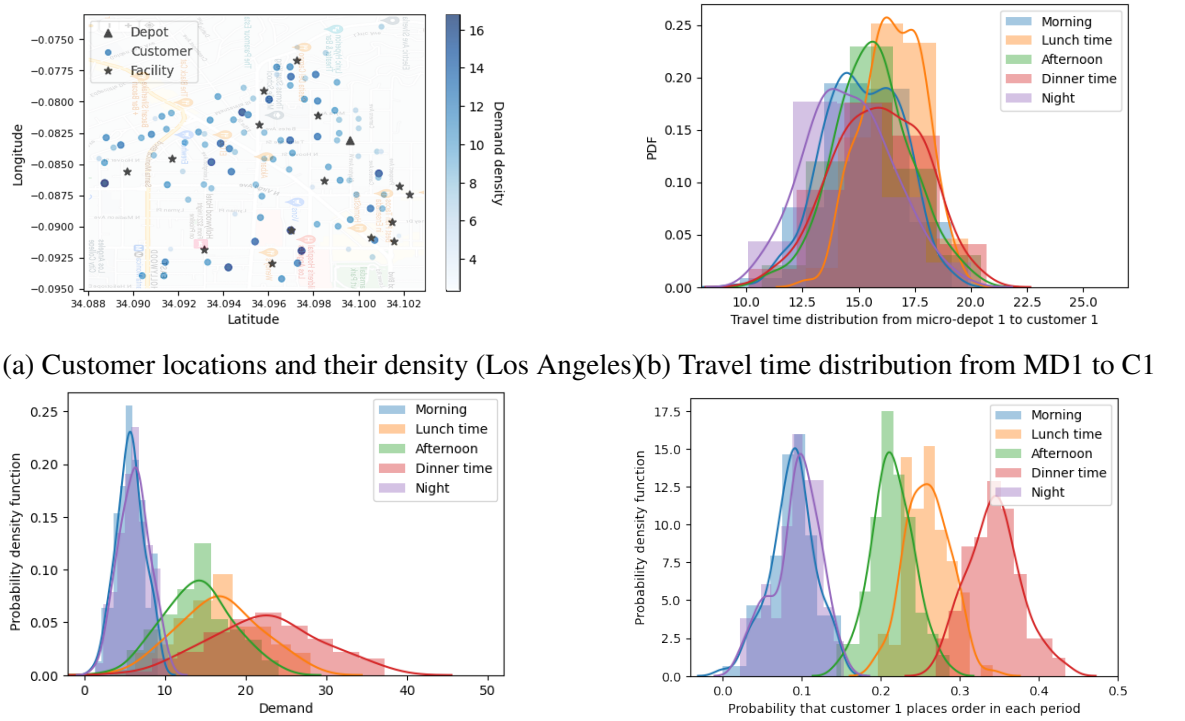
2.5.1 Dataset and Implementation Details

We use the customer location dataset from four regions in the US (Los Angeles, Seattle, Tacoma, and Orange) provided by Amazon (Merchan et al. 2021), which indicates the locations and density of residents inclined to purchase online. For example, the customer location and density in Los Angeles are shown in Figure 2.3(a). The darker the point, the higher the demand volume. For each area, we randomly generate 15 candidate locations for micro-depots. We obtain the distance and real-time travel time from the Google API. Specifically, for each arc between customer and micro-depot locations, we collected 500 travel time samples at different time points from Jan 05, 2023, to Jan 19, 2023. For example, Figure 2.3(b) shows the travel time distribution from micro-depot #1 (MD1) to customer location #1 (C1). To evaluate the out-of-sample performance of ultra-fast delivery in terms of profitability and reliability, for each arc in each period, we generate 500 travel time samples using the gamma distribution, which best fits the real-world dataset, with the same moment information (i.e., mean, variance, skewness) obtained from the real-world dataset. We use 300 samples as training and 200 samples as testing datasets.

We simulate the demand distribution, the probability of customers placing orders in each period, and other cost parameters as follows. We generate the nominal demand distribution for 100 customer locations over 100 days using a normal distribution with a mean of (5, 16, 14, 22, 6) for five periods (morning, lunchtime, afternoon, dinner time, and night) and a variance of 10. The demand distribution for each period is presented in

Figure 2.3(c). The probability distribution of customers placing orders in each period is generated based on the demand distribution. In other words, for each location and each day, the probability of placing orders in each period is proportional to the demand for that period relative to the total demand. Figure 2.3(d) illustrates the probability of placing orders in each period for C1. The revenue of each order r is set at \$3, the delivery cost per kilometer c is \$1, and the hiring cost h of each driver serving per unit demand in each period is \$1. Each driver serves an average of 10 units of demand in each period. The setup cost o_j for opening the micro-depot j in all periods of one day is \$100, and changes between 0 and \$500 in our sensitivity analysis. The initial target delivery time $\bar{\tau}$ is set to 6 minutes, and varies from 5 to 8 minutes in our sensitivity analysis. Since the allowed violation fluctuates from 0 to 38 minutes, the potential target delivery time changes from 5 to 46 minutes. The competitor delivery time τ^c is set to 15 minutes, and varies from 2 to 20 minutes in our sensitivity analysis. The customer utility function is set as $g(\tau) = 1 + \frac{1}{\tau}$, indicating that faster deliveries result in higher utility. This function is assumed to be the same for all customers.

To evaluate the performance of different formulations under various service levels and protection, we compare the profit (i.e., the optimal objective value), the customer coverage proportion (i.e., $\frac{\sum_{i,j,t} x_{ijt}}{|\mathscr{I}||\mathscr{T}|} \times 100\%$), the demand fulfillment proportion (i.e., $\frac{\sum_{i,j,t} \hat{d}_{ijt}}{\sum_{i,t} \hat{d}_{it}} \times 100\%$), the number of open micro-depots (i.e., $\sum_j y_j$), the violation probability, and the violation degree. The violation probability V^p is defined as the average violation probability among all customers in all periods for all discretized chance constraints that correspond to each service level (i.e., $V^p = \frac{1}{|\mathscr{I}||\mathscr{T}||\mathscr{K}|} \sum_{i,t,k} V^p_{itk}$). Specifically, for each customer i in each period t, if the chance constraint at level k is violated, the violation probability is the gap between the target probability and the true probability of serving customers on time (i.e., $V^p_{itk} = \beta(v^k) - P_{\mathscr{F}_o}\left(\sum_j \tau_{ijt} x_{ijt} \leq \bar{\tau} + v^k\right)$, where $\mathscr{F}_o$ is the out-of-sample distribution); otherwise, the violation probability is zero (i.e., $V^p_{itk} = 0$). The violation degree is defined as the maximum amount of time that is beyond the target delivery time among all customers in all periods for all discretized chance constraints (i.e., $V^d = \max_{i,t,k} V^d_{itk}$). Specifically, for each customer i in each period t, if chance constraint k is violated, the delayed time



(a) Customer locations and their density (Los Angeles) (b) Travel time distribution from MD1 to C1

(c) Demand distribution (d) Probability distribution of C1 placing orders

Figure 2.3 – Statistic description of simulation environment

V^d_{itk} is the gap between the highest possible delivery time and the target delivery time (i.e., $V^d_{itk} = \max_{\tilde{\tau} \sim \mathscr{F}_o} \sum_j \tilde{\tau}_{ijt} x_{ijt} - \bar{\tau} - v^k$, where $\mathscr{F}_o$ is the out-of-sample distribution). The profitability is the proportion of the profit that can be achieved compared to the best case that all customers can be served by ultra-fast delivery.

We implement our algorithms using Python 3.7 on a computer with one 2 GHz Quad-Core Intel Core i5 processor and 16GB of RAM. We use Gurobi 9.0.2 as the solver.

2.5.2 Benchmark

We compare the different formulations from three aspects: (1) **Service measures:** period and daily service levels. (2) **Service level guarantees:** one-layer on the service level (i.e., $n = 1$), full protection with the all-layer guarantee (i.e., $n = |\mathscr{K}|$), and partial protection with the multi-layer guarantee (i.e., $n = [2, |\mathscr{K}| - 1]$). Specifically, we employ the inner and outer approximations of $\beta(v)$ as illustrated in Figure 2.2(a), with $|\mathscr{K}| = 20$

and a step size of β set to 0.05. In this case, we implement a 20-layer guarantee as the all-layer guarantee and a 15-layer guarantee (determined to strike an optimal balance between profitability and reliability) as the multi-layer guarantee. (3) **Source of uncertainty:** formulations with or without the uncertainty in travel time distribution and period probability (see Table 2.1).

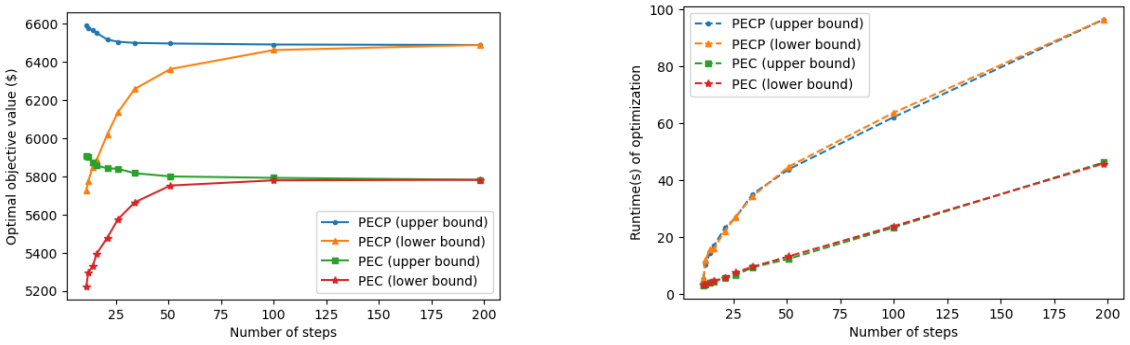
Table 2.1 – Reformulations of different service level under different level of uncertainty

Service level	**Formulation**	**Uncertainty**	**Set**	**Linear reformulation**
Period	PEC	None	$\mathscr{X}_{PEC}$	See Proposition 4
	Robust PEC$_T$	Travel time distribution	$\mathscr{X}_{R-PEC}$	See Proposition 5
Daily	PECP	None	$\mathscr{X}_{PECP}$	See Proposition 6
	Robust PECP$_T$	Travel time	$\mathscr{X}_{R-PECP_T}$	See Remark 5
	Robust PECP$_P$	Period probability	$\mathscr{X}_{R-PECP_P}$	See Remark 7
	Robust PECP$_{TP}$	Travel time distribution; Period probability	$\mathscr{X}_{R-PECP}$	See Proposition 7

*Notes. The subscript is the uncertainty of the robust formulation. For example, Robust PECP$_{TP}$ can be read as **Robust** **P**robabilistic **E**nvelope **C**onstraint when considering **P**eriod probability under uncertain **T**ravel time distribution and **P**eriod probability.*

2.5.3 Performance of β Step Function

To derive a linear reformulation with a finite number of constraints, we use the β step function to approximate the β function. The larger the number of steps, the higher the accuracy, but the lower the efficiency of the solution procedure. Figure 2.4 illustrates the performance of the approximation for different numbers of steps. In the PEC formulation, $\beta^{outer}(v)$ (i.e., lower bound) and $\beta^{inner}(v)$ (i.e., upper bound) converge rapidly, resulting in a gap ratio of 6.63% and an average runtime of 6 seconds when the number of steps is set to 20. In contrast, for the PECP formulation, convergence is slightly slower, with a gap ratio of 8.24% and an average runtime of 23 seconds at 20 steps. Moreover, the upper bound tends to stabilize when the number of steps exceeds 20. In other words, using the approximation $\beta^{inner}(v)$ to approximate the original formulation yields limited



(a) Optimal objective value (b) Runtime

Figure 2.4 – Performance of approximation for different numbers of steps

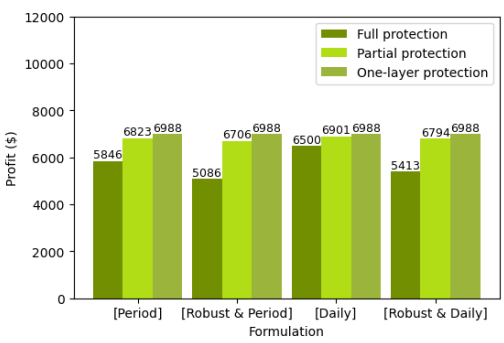
improvement when increasing the number of steps from 20 to larger values. The gap ratio eventually converges to zero at 200 steps, but at the cost of a lengthy preprocessing time, averaging 20 minutes, and 1-3 minutes runtime for optimization.

Insight 1 *The inner and outer approximations are tight when the number of steps exceeds the number of samples in the travel time distribution, as also noted by Peng, Delage, and J. Li (2020). The approximations with 20 steps and a step size of β set to 0.05 perform well, yielding good results in terms of both efficiency and accuracy.*

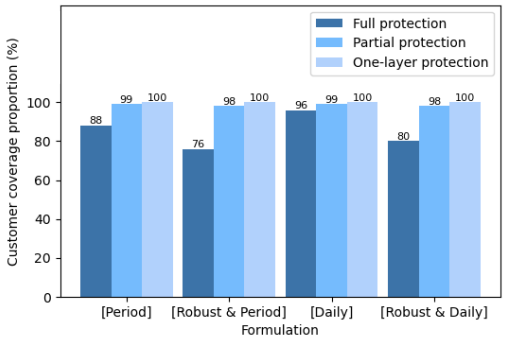
2.5.4 Comparison Under Different Service Levels and Uncertainties

We compare the daily and period service levels with various layers of protection under different uncertainties, as described in Section 2.5.2. Figure 2.5 displays the profit, customer coverage proportion, and the average performance in terms of out-of-sample violation probability and degree. As shown in each sub-figure, the robust formulation always yields a lower violation but at the cost of some loss in profit. For example, the robust formulation with daily service level under partial protection yields a lower out-of-sample violation probability (i.e., 7.0%), a lower out-of-sample violation degree (i.e., 1.21 minutes), but also a lower profit (i.e., $6794) than the non-robust formulation (i.e., 7.9%, 1.54 minutes, and $6901, respectively). That is, the violation probability and violation

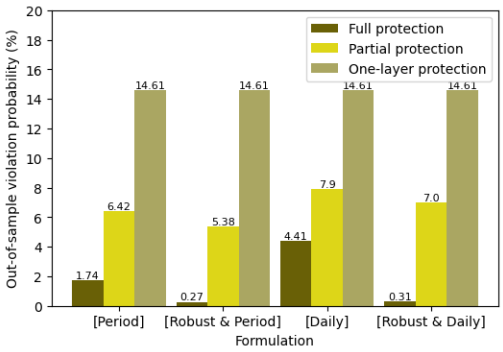
degree decrease by 13% and 21%, respectively, in a positive manner. However, the profit decreases by approximately 1.5%.



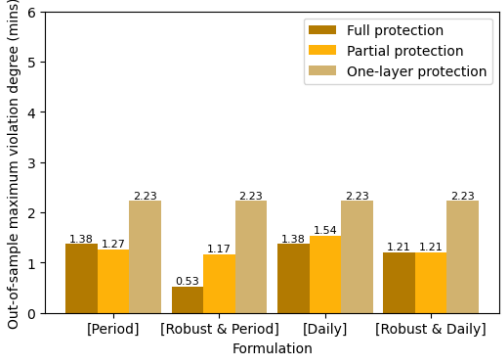
(a) Profit



(b) Customer coverage proportion



(c) Out-of-sample average violation probability



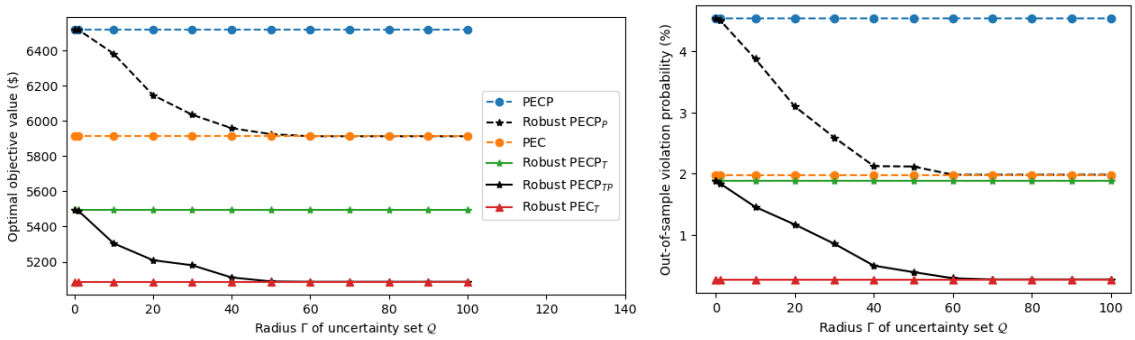
(d) Out-of-sample maximum violation degree

Figure 2.5 – Performance on profit, coverage proportion, and violation

Table 2.2 – Results of different formulations

Formulation	Optimal profit ($)	Number of open micro-depots	Unused micro-depot indices	Customer coverage proportion	Violation probability	Violation degree (minutes)
PECP	6500	10	[1,4,7,8,14]	96%	4.41%	1.38
PEC	5846	11	[1,4,7,14]	88%	1.74%	1.38
Robust PECP$_T$	5413	11	[1,6,7,14]	80%	0.31%	1.21
Robust PEC$_T$	5086	12	[1,7,14]	76%	0.27%	0.53

Notes. The number of potential micro-depot locations is 15 to serve 100 customers.



(a) Profit (b) Out-of-sample average violation probability

Notes. The three dashed lines represent the cases with the explicitly known travel time distribution, and the three solid lines represent the cases with the unknown travel time distribution.

Figure 2.6 – The impact of radius Γ of the uncertainty set $\mathscr{Q}$ for the period probability q.

Figure 2.6 illustrates the change in the optimal objective value and the out-of-sample violation probability as the radius Γ of the uncertainty set for the period probability q varies. When considering PECP with daily service level, increasing Γ leads to larger uncertainty sets, higher protection against uncertain probabilities of order placement in each period, worse objective values, decreased customer coverage, and reduced violations. The best case for PECP occurs when the probability of placing orders in each period is given ($\Gamma = 0$), while the worst case is observed with high uncertainty on the probability of placing orders ($\Gamma \geq 60$), which reduces to PEC with period service level. This observation holds true regardless of whether the travel time distribution is explicitly known or not (see Remark 6).

Table 2.2 displays the open micro-depots under period and daily service levels corresponding to different Γ, ranging from the deterministic case to the most robust scenario. We observe that greater robustness leads to lower profits, reduced customer coverage, decreased violation probabilities, and a higher number of open micro-depots. In other words, the ultra-fast delivery company opens more micro-depots to mitigate risk, yet the coverage of customer locations still diminishes. This suggests that the significant perturbations in customer order frequency and travel time can result in high costs and low revenue.

Insight 2 ***Value of the robustness****: There is a trade-off between high profit and low violation in serving customers on time. The robust formulations can yield lower violation probability and degree, but at the cost of a loss in profit, reaching up to 16.7% in the experimental study.*

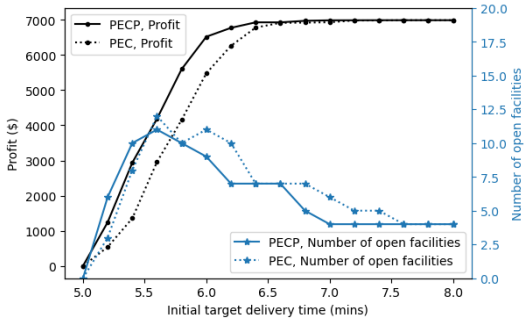
As illustrated in Figure 2.5(a) and (b), the formulation with one-layer protection yields the highest profit due to the highest coverage proportion. However, Figure 2.5(c) indicates that the violation probability under the one-layer protection is much higher than that under full protection. The profit of the formulation with full protection is significantly lower than that of the formulation with one-layer protection. Generally, the formulation with partial protection exhibits the best performance, yielding a decent profit slightly lower than the best case, an acceptable violation probability that is at least half as low as the worst case, and a stable violation degree observed in Figure 2.5(d).

2.5.5 Sensitivity Analysis

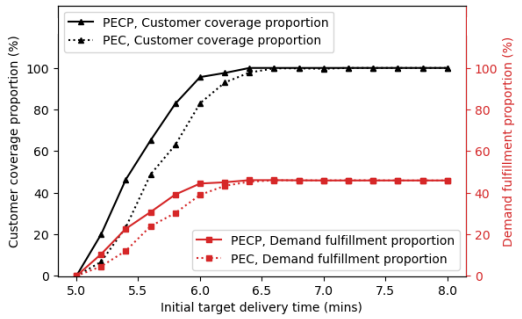
In this section, we examine the influence of the initial target delivery time, competitor delivery time, setup cost, and number of layers on the results. We also present the efficient frontiers concerning profitability and violation probability for both period and daily service levels under various levels of service level protection.

2.5.5.1 The impact of the initial target delivery time.

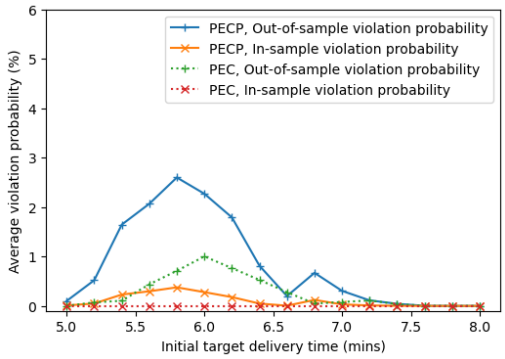
Figure 2.7 shows the changes in profit, number of open micro-depots, customer coverage proportion, demand fulfillment proportion, violation probability, and violation degree as the initial target delivery time changes. A higher initial target delivery time implies less restriction on service levels, resulting in increased profit and greater demand fulfillment. This leads to a trade-off between service levels and fulfillment. Compared to the period service level (PEC), the daily service level (PECP) always yields a higher profit with higher demand fulfillment and coverage proportion (see Figure 2.7(a) and (b)). This fact is on account of two reasons: (1) Compared to PEC, PECP considers the weighted-average



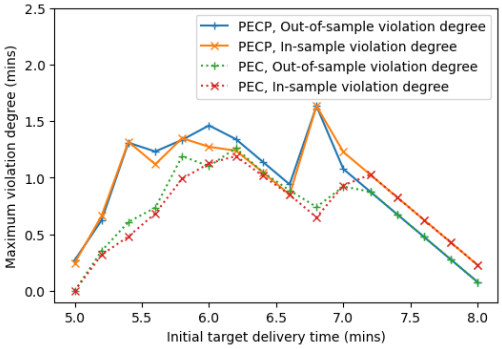
(a) Profit & Number of open micro-depots



(b) Customer coverage & Demand fulfillment proportion



(c) Average violation probability



(d) Maximum violation degree

Figure 2.7 – The impact of the initial target delivery time on PEC and PECP

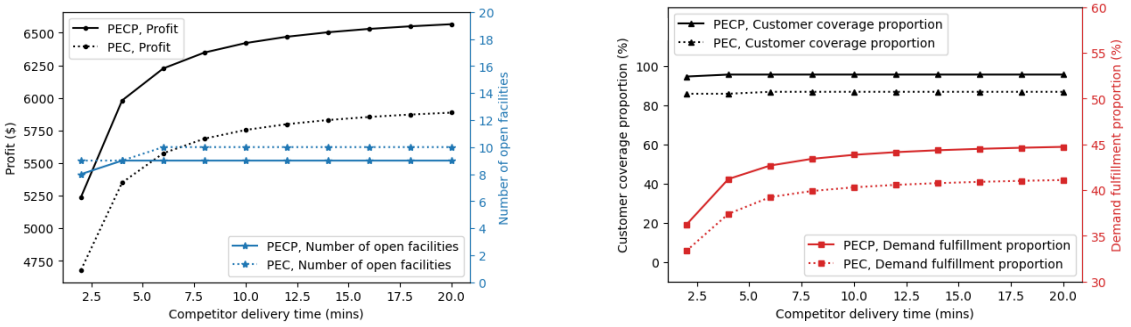
performance among all periods instead of the equivalent performance for each period, leading to a less restricted requirement on the delivery time. (2) Since customers have a higher probability of placing orders at the dinner time and lunch time, given the allowed daily violation, more allowance will be put on these two periods to cover more demand and to yield a higher profit in PECP. The out-of-sample violation probability is at most 2.6% and the violation degree is at most 1.6, which should be acceptable in practice (see Figure 2.7(c) and (d)). More detailed results related to the initial target delivery time in each period are shown in Appendix E.

2.5.5.2 The impact of the competitor delivery time.

Figure 2.8 shows how the profit, number of open micro-depots, customer coverage proportion, and demand fulfillment proportion change as the competitor delivery time changes. As the competitor delivery time increases, the profit of ultra-fast delivery (with the ini-

tial target being 6 minutes) increases with an increasing captured demand. The value is overall stable when the competitor delivery time exceeds 10 minutes. The coverage proportion and the number of open micro-depots keep consistent, which means the allocation decisions remain unchanged no matter how the competitor service level changes. In this case, both the violation probability and degree also remain steady.

Insight 3 *The competitor delivery time does not affect the operations of allocating micro-depots to serve customers, but only impact the demand volume captured by the ultra-fast delivery company. The slower the competitor delivery, the higher the demand captured by the ultra-fast delivery.*

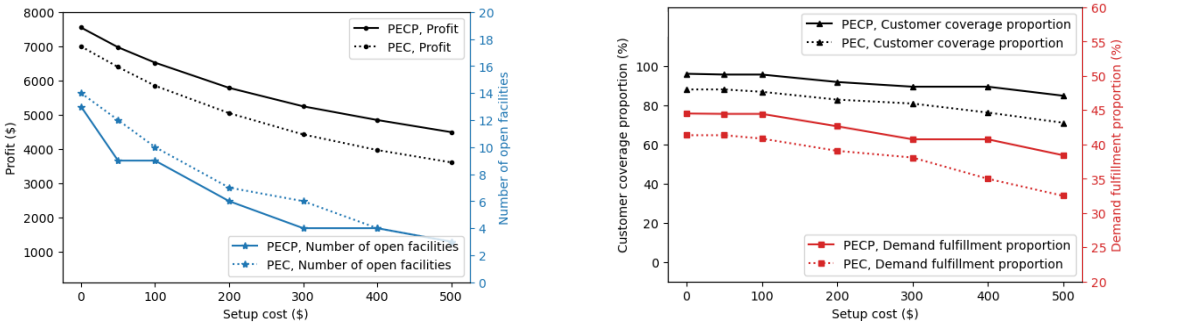


(a) Profit & Number of open micro-depots (b) Customer coverage & Demand fulfillment proportion

Figure 2.8 – The impact of the competitor delivery time on PEC and PECP

2.5.5.3 The impact of the setup cost.

Figure 2.9 shows the changes in profit, number of open micro-depots, customer coverage proportion, demand fulfillment proportion, violation probability, and violation degree as the setup cost varies. The higher the setup cost, the fewer the open micro-depots. In this case, the profit decreases with decreasing demand fulfillment and customer coverage proportions. The violation probability and degree remain overall stable.



(a) Profit & Number of open micro-depots (b) Customer coverage & Demand fulfillment proportion

Figure 2.9 – The impact of the setup cost on PEC and PECP

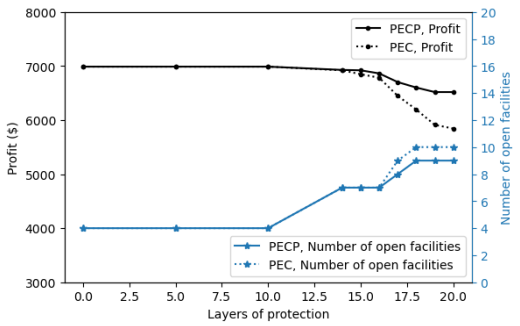
2.5.5.4 The impact of the layers of protection.

Figure 2.10 demonstrates the changes in profit, number of open micro-depots, customer coverage proportion, demand fulfillment proportion, violation probability, and violation degree with variations in the layers of protection. The more the layers of protection, the more reliable the ultra-fast delivery service. When the number of layers increases, the profit first remains unchanged and then decreases, due to a lower captured demand and a lower coverage proportion (see Figure 2.10 (a) and (b)). Both the violation probability and degree decrease (see Figure 2.10 (c) and (d)).

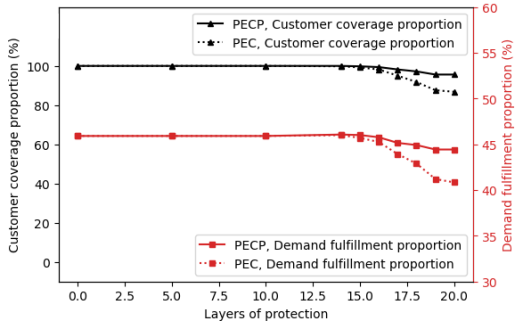
Insight 4 ***Value of the daily service level****: Regardless of changes in the initial target delivery time, competitor delivery time, setup cost, or layers of protection, the daily service level consistently outperforms the period service level in terms of higher profit, greater coverage, and milder violations.*

2.5.6 Efficient Frontier of Four Regions for Varying Service Guarantees

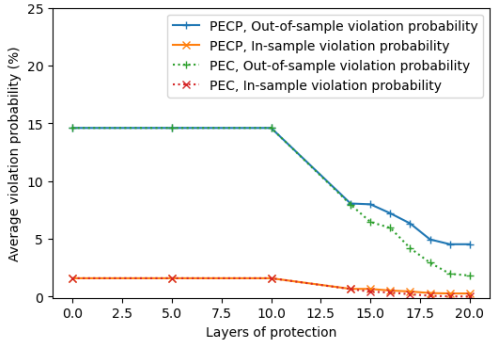
Inevitably, there is trade-off between the profit and the service level. The more the protection on the service level, the lower the profit. The trade-off changes for different regions with varying customer densities.



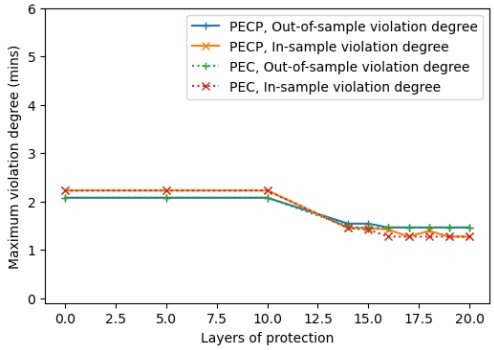
(a) Profit & Number of open micro-depots



(b) Customer coverage & demand fulfillment proportion



(c) Average violation probability



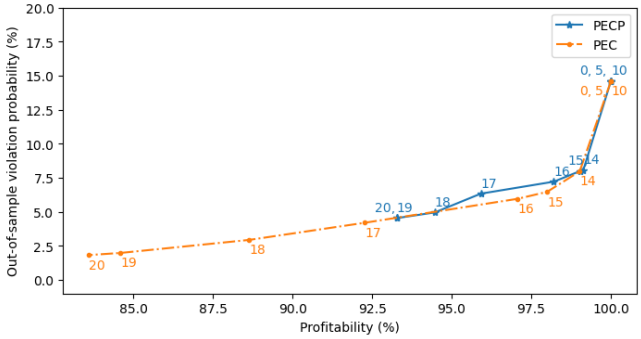
(d) Maximum violation degree

Figure 2.10 – The impact of protection layers

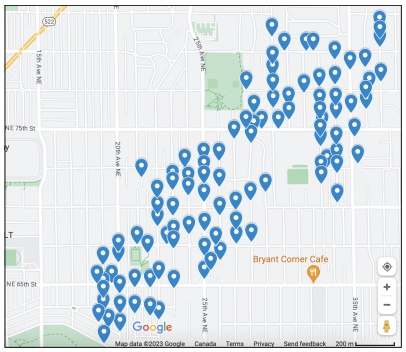
In Figure 2.11, we display customer distributions in four regions and plot their profitability and out-of-sample violation probability under varying layers of service level protection. Connecting these points forms an efficient frontier of solutions for Los Angeles (LA), Seattle, Tacoma, and Orange, respectively. According to the density of customer locations per square kilometer, we classify LA (33 customers/km^2) and Seattle (42 customers/km^2) as urban areas, while we consider Tacoma (18 customers/km^2) and Orange (17 customers/km^2) as rural areas.

Without any protection, each region achieves its 100% profitability by serving all customers, and the violation probability of serving customers on time for rural areas is higher than that of urban areas. For all cases, the steepest slope between points is that between the 10-layer and 15-layer points. By comparing the slope between these two points of different regions, we find that the slope of urban areas is always steeper than that of rural areas. That is, the violation probability is almost halved by only sacrificing

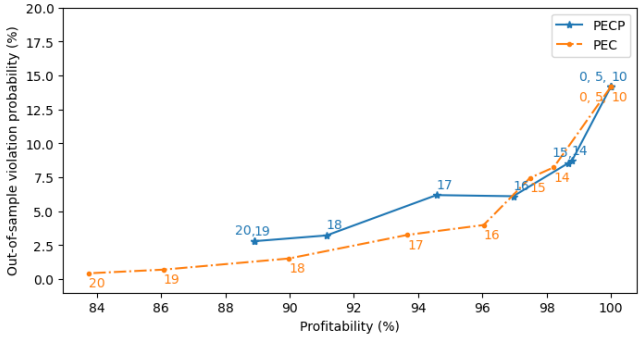
(a) LA (urban area, 100 customers, 33 customers/km^2)



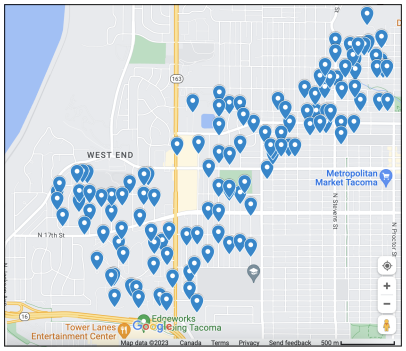
(b) EF of LA



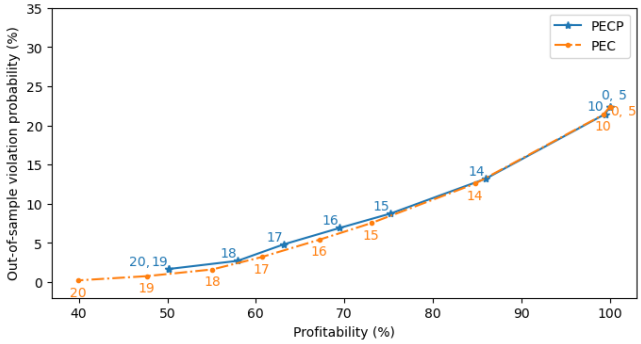
(c) Seattle (urban area, 85 customers, 42 customers/km^2)



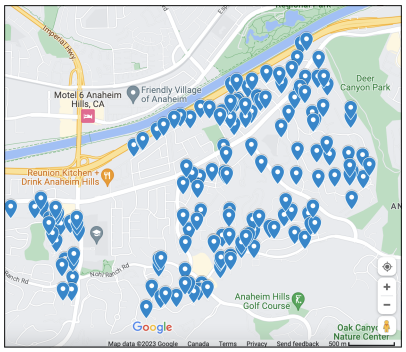
(d) EF of Seattle



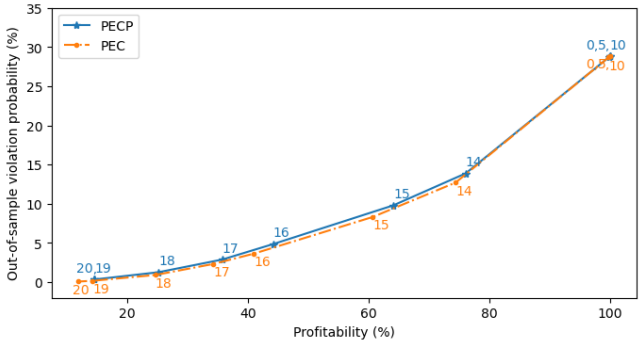
(e) Tacoma (rural area, 110 customers, 18 customers/km^2)



(f) EF of Orange



(g) Orange (rural area, 135 customers, 17 customers/km^2)



(h) EF of Orange

Notes. The colored numbers next to points denote the number of layers of protection.

Figure 2.11 – Customer distributions and efficient frontiers (EF) under varying service guarantees

$1-2\%$ profitability for urban areas, but by sacrificing $13-25\%$ profitability for rural areas.

Insight 5 *Compared to dense urban areas, maintaining a high service level of on-time delivery is more challenging in rural areas, where customers are more dispersed, making it harder to sustain profitable and reliable fast deliveries. This is due to the longer distances between delivery locations, necessitating the setup of more micro-depots in rural regions.*

Insight 6 ***Value of the multi-layer partial protection****: Providing full protection with the lowest profitability is too conservative, while offering no-layer protection with the highest probability of violating the promised service level is too risky. A multi-layered partial protection strategy (e.g., using 15 layers) can strike a better balance between the profitability and reliability.*

In addition, the partial protection on the delivery time is practical in real-life scenarios, as delivery companies are not obligated to guarantee on-time delivery at all levels. Taking the urban area in Los Angeles with a customer density of 33 customers/km^2 (see Figure 2.11(a)) as an example, promising delivery within an average of 15 minutes might be too lenient, while guaranteeing delivery within exactly 15 minutes could be too stringent. A stepwise approach to delivery promises, such as ensuring delivery to 99% of customers within 43 minutes, 75% within 11 minutes, and 40% within 6 minutes, proves to be a more effective strategy, regardless of when customers place their orders or the prevailing traffic conditions at that time. Therefore, implementing an optimized service level with partial protection could be a viable strategy for ultra-fast delivery companies to operate a profitable business and maintain a good service level without over-committing or under-delivering. In real-world business, we can customize delivery strategies for different customer groups, each with varying service levels. For example, customers can choose *Premium delivery* with full protections, which offers high reliability and guarantees high compensation for delays. The lower profitability from this high-reliability service can be offset by membership fees or higher delivery fee. *Standard delivery* offers

a balanced trade-off with partial protections, providing medium reliability and leading to decent profitability, catering to customers who value both speed and cost efficiency. Finally, *Economy delivery* without protection targets customers who are less sensitive to delivery time, allowing for a wider service area and lower costs. This service may offer fewer guarantees and longer delivery times but ensures affordability and access to delivery services for those who prioritize savings over speed.

2.6 Conclusion

The ultra-fast delivery service industry has emerged suddenly and expanded rapidly, but it also scales down quickly, often due to business failures or bankruptcies. This prompts us to consider its profitability while maintaining on-time and fast deliveries. To find an effective strategy for operating ultra-fast delivery services, we model and solve a network design problem using probabilistic envelope constrained programs under uncertainty in travel time distribution and period probability. We investigate both period and daily service levels of ultra-fast delivery, considering various layers of service level protection. While the period service level emphasizes equal service across periods, the daily service level prioritizes high-order frequency periods and guarantees a certain service level for the entire day. The probabilistic envelope constrained programs are computationally challenging when the distribution of travel time and the probability of customers placing orders in different time periods are not explicitly known. To address this, we derive equivalent linear constrained programs with an infinite number of constraints and then propose outer and inner approximations with finite linear constraints. We conduct a numerical study using a real-world dataset provided by Amazon and obtained through the Google API.

The results reveal that the outer and inner approximations converge rapidly as the number of steps increases. Additionally, the approximations becomes tight when the number of steps surpasses that of the training samples. Notably, the approximation using 20 steps demonstrates good performance in terms of both efficiency and accuracy. By

comparing the out-of-sample performance, we observe that the robust formulation can yield a lower probability of violating the target delivery time, and a reduced degree of exceeding the bound in case of violation. However, this comes at the expense of a lower profit. When we compare the performance of period and daily service levels under different layers of protection and investigate the impact of various factors on the results, we obtain the following managerial insights: (1) The daily service level has an overall better performance than the period service level with higher profitability, higher coverage, and mild violation. (2) Full protection provides low profitability and is overly conservative. On the other hand, offering either one-layer or no-layer protection with a high probability of violating the promised service level is overly risky. Implementing multi-layered protection by optimizing the service level guarantee could be a good strategy for an ultra-fast delivery company to run a profitable and reliable business. (3) The competitor delivery time may not affect the allocation operations, but only impact the demand volume captured by the ultra-fast delivery company. (4) Compared to urban areas, maintaining a high service level is more challenging in rural areas where customers are more dispersed.

Our work has some limitations that could be addressed in future research. Specifically, we assume that an unlimited number of drivers are available and that each customer can be served instantly upon placing an order. This assumption can be relaxed to account for routing decisions with a limited number of available drivers. Additionally, real-world scenarios often involve batch processing, where a single driver serves multiple customers located close to each other and who place orders within a short time frame. To address this, it would be necessary to determine the optimal batch size, the composition of orders within each batch, and the assignment of batches to drivers. Furthermore, the variation in products and customer preferences regarding delivery times across different customer types can be incorporated to develop more complex models and offer insights from a marketing perspective. Lastly, other methods, such as queuing models, can account for order preparation and delivery times from a more practical standpoint, while reinforcement learning can enable real-time operational planning for ultra-fast delivery.

2.7 Appendix

Appendix A: Summary of Notations

The notation is presented in Table 3.5.

Table 2.3 – Notations

Index	**Description**
$\mathscr{I}$	set of customer locations
$\mathscr{J}$	set of potential micro-depot locations
$\mathscr{T}$	set of time periods
$\mathscr{K}$	set of steps in $\beta(v)$ step functions
$\mathscr{X}$	set of allocation decisions
Parameters	**Description**
o_j	setup cost of micro-depot j
c	delivery cost per unit of distance
r	average revenue per order
$\bar{d}_{it}$	nominal demand at location i in period t
l_{ij}	distance between customer i and micro-depot j
$\tilde{s}_{ijt}$	uncertain travel time from micro-depot j to customer i in period t
$\tilde{\tau}_{ijt}$	uncertain delivery time from micro-depot j to customer i in period t
$\tilde{\delta}_{ijt}$	random part of uncertain delivery time from micro-depot j to customer i in period t, i.e., $\tilde{\delta}_{ijt} = \tilde{\tau}_{ijt} - \hat{\tau}_{ijt}$
Σ	covariance matrix of $\tilde{\delta}$
τ^u_{it}	delivery time from the assigned micro-depot to customer i in period t
τ^c_{it}	delivery time of the best competitor to serve customer i in period t
a_{ijt}	order preparation time for customer i served by micro-depot j in period t
h	hiring cost of one driver per period
m	average units of demand served by each driver in each period
$\bar{\tau}$	target delivery time
v	maximum violation
β	probability of meeting the target delivery time
q_{it}	probability of customer i placing an order in period t
Σ_q	covariance matrix of the observations of the period probability q
Γ	radius of the uncertainty set of the period probability q
Decisions	**Description**
x_{ijt}	binary variable taking value 1 if customer i is covered by micro-depot j in period t, and 0 otherwise
y_j	binary variable taking value 1 if micro-depot j is open, and 0 otherwise
d_{ijt}	captured demand at location i served by micro-depot j in period t
z_t	number of drivers needed in period t

Appendix B: Detailed Proofs of Propositions

B.1: Proof of Proposition 4

Proof. We rewrite the PEC (2.2) as

$$\inf_{v\geq 0}\mathbb{P}_{\tilde{\tau}}\left\{\sum_{j}\tilde{\tau}_{ijt}x_{ijt}\leq\bar{\tau}+v\right\}-\beta(v)\geq 0,\forall i,t. \qquad (A)$$

Since $x_{ijt}\in\{0,1\}$ and $\sum_j x_{ijt}\leq 1$, the above equation is equivalent to

$$x_{ijt}\leq\mathbb{I}\left\{\inf_{v\geq 0}\mathbb{P}_{\tilde{\tau}}\left\{\tilde{\tau}_{ijt}\leq\bar{\tau}+v\right\}-\beta(v)\geq 0\right\},\forall i,j,t, \qquad (B)$$

where $\mathbb{I}\{\cdot\}$ is the indicator function. To show that $(A)\Leftrightarrow(B)$, we investigate two cases:
(1) When $\sum_j x_{ijt}=0$, we have $x_{ijt}=0$. In this case, the left-hand side of equation (A) is equal to $1-\beta(v)$ since $\{0\leq\bar{\tau}+v\}$ is always satisfied with probability 1. Thus, the equation (A) being $1-\beta(v)\geq 0$ is always feasible. Additionally, the equation (B) is also feasible with the left hand side being equal to 0.
(2) When $\sum_j x_{ijt}=1$, let $x_{ij't}=1$ and $x_{ijt}=0$ when $j\neq j'$. In this case, we have

$$(B)\quad\Leftrightarrow\quad\inf_{v\geq 0}\mathbb{P}_{\tilde{\tau}}\left\{\tilde{\tau}_{ij't}\leq\bar{\tau}+v\right\}-\beta(v)\geq 0,\forall i,t\quad\Leftrightarrow\quad(A).$$

Our next step is to assume that $\tilde{\tau}$ follows a continuous distribution. We define $\Psi_{\tilde{\tau}_{ijt}}$ as the cumulative probability function of $\tilde{\tau}_{ijt}$, and $\Psi^{-1}_{\tilde{\tau}_{ijt}}(\beta)$ as its quantile at probability β. We have

$$x_{ijt}\leq\mathbb{I}\left\{\sup_{v\geq 0}\Psi^{-1}_{\tilde{\tau}_{ijt}}(\beta(v))-\bar{\tau}-v\leq 0\right\},\forall i,j,t.$$

∎

B.2: Proof of Proposition 5

Proof. To simplify the robust PEC (2.8) even more, we can rewrite it as

$$\mathbb{I}\left\{\inf_{v\geq 0,\tilde{\delta}_{it}\sim(0,\Sigma_{it})}\mathbb{P}\left\{\left(\hat{\tau}_{it}+\tilde{\delta}_{it}\right)^{T}x_{it}\leq\bar{\tau}+v\right\}-\beta(v)\geq 0\right\}\geq 1,\forall i,t,$$

where $\mathbb{I}\{\cdot\}$ is the indicator function. Exploiting that $x_{ijt} \in \{0,1\}$ and $\sum_j x_{ijt} \leq 1$, we get

$$\sum_j \mathbb{I}\left\{\inf_{v\geq 0, \tilde{\delta}_{ijt}\sim(0,\sigma_{ijt}^2)} \mathbb{P}\left\{\hat{\tau}_{ijt} + \tilde{\delta}_{ijt} \leq \bar{\tau} + v\right\} - \beta(v) \geq 0\right\} x_{ijt} \geq \sum_j x_{ijt}, \forall i,t,$$

which is equivalent to

$$x_{ijt} \leq \mathbb{I}\left\{\inf_{v\geq 0, \tilde{\delta}_{ijt}\sim(0,\sigma_{ijt}^2)} \mathbb{P}\left\{\hat{\tau}_{ijt} + \tilde{\delta}_{ijt} \leq \bar{\tau} + v\right\} - \beta(v) \geq 0\right\}, \forall i,j,t.$$

Exploiting the reformulation (2.10) presented in Lemma 1, for each i,j,t, instead of verifying

$$\inf_{\tilde{\delta}_{ijt}\sim(0,\sigma_{ijt}^2)} \mathbb{P}\left\{\hat{\tau}_{ijt} + \tilde{\delta}_{ijt} \leq \bar{\tau} + v\right\} - \beta(v) \geq 0, \forall v \geq 0,$$

one can simply verify whether

$$\sup_{v\geq 0} \hat{\tau}_{ijt} + \sqrt{\frac{\beta(v)}{1-\beta(v)}}\sigma_{ijt} - \bar{\tau} - v \leq 0.$$

Hence, the robust PEC is equivalent to

$$x_{ijt} \leq \mathbb{I}\left\{\sup_{v\geq 0} \hat{\tau}_{ijt} + \sqrt{\frac{\beta(v)}{1-\beta(v)}}\sigma_{ijt} - \bar{\tau} - v \leq 0\right\}, \forall i,j,t,$$

which is linear in x_{ijt}, leading to a linear program.

In the case that $\beta(v) := \frac{1}{\frac{\gamma}{v+\alpha}+1}$, the robust PEC is equivalent to $x_{ijt} \leq \mathbb{I}\left\{\hat{\tau}_{ijt} + \alpha + \frac{\sigma_{ijt}^2}{4\gamma} - \bar{\tau} \leq 0\right\}, \forall i,j,t$. This is because we can optimize v out of the equation and derive the optimal $v^* = \frac{\sigma_{ijt}^2}{4\gamma} - \alpha$. This optimal v^* exists and is unique since $F(v) = \hat{\tau}_{ijt} + \sqrt{\frac{\beta(v)}{1-\beta(v)}}\sigma_{ijt} - \bar{\tau} - v$ is concave with its second derivative (i.e., $\frac{-1}{4\gamma}(\frac{v+\alpha}{\gamma})^{-\frac{3}{2}}$) being negative. ∎

B.3: Proof of Proposition 6

Proof. Suppose that there is a finite number of periods $t \in \mathscr{T}$. For any customer i in each period t such that $\mathbb{P}\left(\sum_j x_{ij\tilde{t}} = 1\right) > 0$, the PECP (2.12) can be reformulated as

$$\mathbb{P}_{\tilde{\tau},\tilde{t}}\left(\sum_j \tilde{\tau}_{ij\tilde{t}} x_{ij\tilde{t}} \leq \bar{\tau} + v \middle| \sum_j x_{ij\tilde{t}} = 1\right) \geq \beta(v), \forall i, \forall v \geq 0 \tag{2.22a}$$

$$\equiv \quad \frac{\mathbb{P}_{\tilde{\tau},\tilde{t}}\left(\sum_j \tilde{\tau}_{ij\tilde{t}} x_{ij\tilde{t}} \le \bar{\tau}+v \;\&\; \sum_j x_{ij\tilde{t}}=1\right)}{\mathbb{P}_{\tilde{t}}\left(\sum_j x_{ij\tilde{t}}=1\right)} \ge \beta(v), \forall i, \forall v \ge 0 \tag{2.22b}$$

$$\equiv \quad \frac{\sum_t q_{it}\mathbb{P}_{\tilde{\tau}}\left(\sum_j \tilde{\tau}_{ijt} x_{ijt} \le \bar{\tau}+v\right)\mathbb{P}\left(\sum_j x_{ijt}=1\right)}{\sum_t q_{it}\mathbb{P}\left(\sum_j x_{ijt}=1\right)} \ge \beta(v), \forall i, \forall v \ge 0 \tag{2.22c}$$

$$\equiv \quad \sum_t q_{it}\left[\mathbb{P}_{\tilde{\tau}}\left(\sum_j \tilde{\tau}_{ijt} x_{ijt} \le \bar{\tau}+v\right)\mathbb{I}\left(\sum_j x_{ijt}=1\right)\right] \ge \beta(v)\sum_t q_{it}\mathbb{I}\left(\sum_j x_{ijt}=1\right), \quad \forall i, \forall v \ge 0 \tag{2.22d}$$

$$\equiv \quad \sum_t q_{it}\left[\left(\sum_j x_{ijt}\right)\mathbb{P}_{\tilde{\tau}}\left(\sum_j \tilde{\tau}_{ijt} x_{ijt} \le \bar{\tau}+v\right)\right] \ge \beta(v)\sum_t q_{it}\sum_j x_{ijt}, \forall i, \forall v \ge 0 \tag{2.22e}$$

$$\equiv \quad \sum_t q_{it}\left[\sum_j \mathbb{P}_{\tilde{\tau}}\left\{\tilde{\tau}_{ijt} \le \bar{\tau}+v\right\} x_{ijt}\right] \ge \beta(v)\sum_t\sum_j q_{it} x_{ijt}, \forall i, \forall v \ge 0, \tag{2.22f}$$

$$\equiv \quad \sum_t q_{it}\left[\sum_j [\Psi_{\tilde{\tau}}(\bar{\tau}+v)-\beta(v)] x_{ijt}\right] \ge 0, \forall i, \forall v \ge 0. \tag{2.22g}$$

In the case that $\mathbb{P}\left(\sum_j x_{ij\tilde{t}}=1\right)=0$, the constraint is redundant since it is always satisfied.

∎

B.4: Proof of Proposition 7

Proof. According to the strong duality, we obtain the robust counterpart of (2.15) under the uncertainty set $\mathscr{Q}_i = \left\{ q_i \in \mathbb{R}^{|\mathscr{T}|} \mid q_i^T e = 1,\ 0 \le q_i \le 1,\ \left\| \Sigma_{q_i}^{-\frac{1}{2}}(q_i - \hat{q}_i) \right\|_1 \le \Gamma \right\}$:

$$\begin{array}{lll}
 & \inf_{q_i \in \mathscr{Q}_i} & \sum_t q_{it}\left(\sum_j \left[\Upsilon_{ijt}(v)-\beta(v)\right] x_{ijt}\right) \ge 0, \forall i, \forall v \ge 0 \\
\equiv & \sup_{q_i \in \mathscr{Q}_i} & \sum_t q_{it}\left(\beta(v) x_{it}^T I - x_{it}^T \Upsilon_{it}(v)\right) \le 0, \forall i, \forall v \ge 0 \\
\equiv & \sup_q & \delta\left(\sum_t e_t x_{it}^T \left(\beta(v) I - \Upsilon_{it}(v)\right) \;\middle|\; \mathscr{Q}_i\right) \le 0, \forall i, \forall v \ge 0 \\
\equiv & \inf_{u_1,u_2,\theta_1} & \hat{q}_i^T u_{1i} + \Gamma \left\| \Sigma_{q_i}^{\frac{1}{2}} u_{1i} \right\|_\infty + \theta_{2i} \le 0, \forall i, \forall v \ge 0 \\
 & \text{s.t.} & u_{1i} + u_{2i} = \sum_t e_t x_{it}^T \left(\beta(v) I - \Upsilon_{it}(v)\right), \forall i \\
 & & \theta_{2i} \ge u_{2it}, \forall i, t \\
\equiv & \inf_{u_1,\theta_1,\theta_2} & \hat{q}_i^T u_{1i} + \Gamma \theta_{1i} + \theta_{2i} \le 0, \forall i, \forall v \ge 0 \\
 & \text{s.t.} & u_{1it} + \theta_{2i} \ge \beta(v) x_{it}^T I - x_{it}^T \Upsilon_{it}(v), \forall i, t \\
 & & \theta_{1i} \ge u_{1i}^T [\Sigma_{q_i}^{\frac{1}{2}}]_t, \forall i, t \\
 & & \theta_{1i} \ge -u_{1i}^T [\Sigma_{q_i}^{\frac{1}{2}}]_t, \forall i, t,
\end{array}$$

where $e_t \in \mathbb{R}^{|\mathscr{T}|}$ is the t^{th} column of the identity matrix, $\delta(v|\mathscr{Q}_i) = \sup_{q_i \in \mathscr{Q}_i} q_i^T v$ is the support function of $\mathscr{Q}_i$, and $[\Sigma_{q_i}^{\frac{1}{2}}]_t$ is the t^{th} column of the matrix $\Sigma_{q_i}^{\frac{1}{2}}$. Note that u_1, θ_1, θ_2 are dependent on v. Additionally, $\Upsilon_{ijt}(v) = \inf_{\tilde{\delta}_{ijt} \sim (0,\sigma_{ijt}^2)} \Psi_{\tilde{\delta}_{ijt}} \left\{ \bar{\tau} + v - \hat{\tau}_{ijt} \right\}$, and can be reformulated as:

$$\Upsilon_{ijt}(v) = \frac{(\bar{\tau} + v - \hat{\tau}_{ijt})_+^2}{(\bar{\tau} + v - \hat{\tau}_{ijt})_+^2 + \sigma_{ijt}^2},$$

where $(y)_+ := \max(0, y)$. This is because

$$\begin{aligned}
\Upsilon_{ijt}(v) &= \inf_{\tilde{\delta}_{ijt} \sim (0,\sigma_{ijt}^2)} \mathbb{P}\left\{ \hat{\tau}_{ijt} + \tilde{\delta}_{ijt} \leq \bar{\tau} + v \right\} \\
&= \sup \left[\lambda : \inf_{\tilde{\delta}_{ijt} \sim (0,\sigma_{ijt}^2)} \mathbb{P}\left\{ \hat{\tau}_{ijt} + \tilde{\delta}_{ijt} \leq \bar{\tau} + v \right\} \geq \lambda \right] \\
&= \sup[\lambda : \hat{\tau}_{ijt} + \sigma_{ijt} \sqrt{\lambda/(1-\lambda)} \leq \bar{\tau} + v] \\
&= \sup \left[\lambda : \lambda \leq \begin{cases} \frac{(\bar{\tau}+v-\hat{\tau}_{ijt})^2}{(\bar{\tau}+v-\hat{\tau}_{ijt})^2 + \sigma_{ijt}^2} & \text{if } \bar{\tau} + v - \hat{\tau}_{ijt} \geq 0 \\ 0 & \text{otherwise} \end{cases} \right]. \\
&= \frac{(\bar{\tau} + v - \hat{\tau}_{ijt})_+^2}{(\bar{\tau} + v - \hat{\tau}_{ijt})_+^2 + \sigma_{ijt}^2}
\end{aligned}$$

∎

Appendix C: Linear Program Representation of Outer and Inner Approximations

The feasible sets of x, including $\mathscr{X}_{PEC}$, $\mathscr{X}_{R-PEC}$, $\mathscr{X}_{PECP}$, and $\mathscr{X}_{R-PECP}$, can be reformulated into a finite set of linear constraints using their respective outer and inner approximations. This section covers the presentation of these approximations, with the exception of the approximations for $\mathscr{X}_{PEC}$, which are discussed in the main text.

C.1: Outer and Inner Approximations of $\mathscr{X}_{R-PEC}$

Corollary 2 *When $\beta(v)$ is approximated by its outer and inner step functions* (2.5)*, the approximated reformulation of $\mathscr{X}_{R-PEC}(v)$ is*

$$\mathscr{X}_{PEC}^{outer}\left(\{v^k\}_{k\in\mathscr{K}}\right) \subseteq \mathscr{X}_{PEC}(v) \subseteq \mathscr{X}_{PEC}^{inner}\left(\{v^k\}_{k\in\mathscr{K}}\right)$$

with

$$\mathscr{X}_{R-PEC}^{inner}\left(\{v^k\}_{k\in\mathscr{K}}\right) := \left\{x\in\mathbb{R}^{|\mathscr{I}|\times|\mathscr{J}|\times|\mathscr{T}|}\middle| x_{ijt}\leq\Theta_{ijt}^{inner},\forall i,j,t\right\}, \tag{2.23}$$

$$\mathscr{X}_{R-PEC}^{outer}\left(\{v^k\}_{k\in\mathscr{K}}\right) := \left\{x\in\mathbb{R}^{|\mathscr{I}|\times|\mathscr{J}|\times|\mathscr{T}|}\middle| x_{ijt}\leq\Theta_{ijt}^{outer},\forall i,j,t\right\}, \tag{2.24}$$

where $\Theta_{ijt}^{inner} := \min_k \mathbb{I}\left\{\hat{\tau}_{ijt}+\sqrt{\frac{\beta(v^k)}{1-\beta(v^k)}}\sigma_{ijt}-\bar{\tau}-v^k\leq 0\right\}$ *and*
$\Theta_{ijt}^{outer} := \min_k \mathbb{I}\left\{\hat{\tau}_{ijt}+\sqrt{\frac{\beta(v^{k+1})}{1-\beta(v^{k+1})}}\sigma_{ijt}-\bar{\tau}-v^{k+1}\leq 0\right\}$.

C.2: Outer and Inner Approximations of $\mathscr{X}_{PECP}$

Corollary 3 *When $\beta(v)$ is approximated by its outer and inner step functions* (2.5)*, the approximated reformulation of $\mathscr{X}_{PECP}(v)$ is*

$$\mathscr{X}_{PECP}^{outer}\left(\{v^k\}_{k\in\mathscr{K}}\right) \subseteq \mathscr{X}_{PECP}(v) \subseteq \mathscr{X}_{PECP}^{inner}\left(\{v^k\}_{k\in\mathscr{K}}\right)$$

with

$$\mathscr{X}_{PECP}^{inner}\left(\{v^k\}_{k\in\mathscr{K}}\right) := \left\{x\in\mathbb{R}^{|\mathscr{I}|\times|\mathscr{J}|\times|\mathscr{T}|}\middle|\sum_t q_{it}\left(\sum_j\left[\Psi_{\tilde{\tau}}(\bar{\tau}+v^k)-\beta(v^k)\right]x_{ijt}\right)\geq 0,\forall i,k\right\} \tag{2.25}$$

$$\mathscr{X}_{PECP}^{outer}\left(\{v^k\}_{k\in\mathscr{K}}\right) := \left\{x\in\mathbb{R}^{|\mathscr{I}|\times|\mathscr{J}|\times|\mathscr{T}|}\middle|\sum_t q_{it}\left(\sum_j\left[\Psi_{\tilde{\tau}}(\bar{\tau}+v^{k+1})-\beta(v^{k+1})\right]x_{ijt}\right)\geq 0,\forall i,k\right\} \tag{2.26}$$

C.3: Outer and Inner Approximations of $\mathscr{X}_{R-PECP}$

Corollary 4 *When $\beta(v)$ is approximated by its outer and inner step functions* (2.5)*, the approximated reformulation of $\mathscr{X}_{R-PECP}(v)$ is*

$$\mathscr{X}^{outer}_{R-PECP}\left(\{v^k\}_{k\in\mathscr{K}}\right) \subseteq \mathscr{X}_{R-PECP}(v) \subseteq \mathscr{X}^{inner}_{R-PECP}\left(\{v^k\}_{k\in\mathscr{K}}\right)$$

with

$$\mathscr{X}^{inner}_{R-PECP}\left(\{v^k\}_{k\in\mathscr{K}}\right) := \left\{ x\in\mathbb{R}^{|\mathscr{I}|\times|\mathscr{J}|\times|\mathscr{T}|} \;\middle|\; \begin{array}{l} \exists \left\{u_1^k,\theta_1^k,\theta_2^k\right\}_{k=1}^{|\mathscr{K}|} \\ \hat{q}_i^T u_{1i}^k + \Gamma\theta_{1i}^k + \theta_{2i}^k \le 0, \forall i,k \\ u_{1it}^k + \theta_{2i}^k \ge \beta(v^k) x_{it}^T I - x_{it}^T \Upsilon_{it}(v^k), \forall i,t,k \\ \theta_{1i}^k \ge (u_{1i}^k)^T [\Sigma^{\frac{1}{2}}_{q_i}]_t, \forall i,t,k \\ \theta_{1i}^k \ge -(u_{1i}^k)^T [\Sigma^{\frac{1}{2}}_{q_i}]_t, \forall i,t,k \end{array} \right\} \quad (2.27)$$

$$\mathscr{X}^{outer}_{R-PECP}\left(\{v^k\}_{k\in\mathscr{K}}\right) := \left\{ x\in\mathbb{R}^{|\mathscr{I}|\times|\mathscr{J}|\times|\mathscr{T}|} \;\middle|\; \begin{array}{l} \exists \left\{u_1^k,\theta_1^k,\theta_2^k\right\}_{k=1}^{|\mathscr{K}|} \\ \hat{q}_i^T u_{1i}^k + \Gamma\theta_{1i}^k + \theta_{2i}^k \le 0, \forall i,k \\ u_{1it}^k + \theta_{2i}^k \ge \beta(v^{k+1}) x_{it}^T I - x_{it}^T \Upsilon_{it}(v^{k+1}), \forall i,t,k \\ \theta_{1i}^k \ge (u_{1i}^k)^T [\Sigma^{\frac{1}{2}}_{q_i}]_t, \forall i,t,k \\ \theta_{1i}^k \ge -(u_{1i}^k)^T [\Sigma^{\frac{1}{2}}_{q_i}]_t, \forall i,t,k \end{array} \right\} \quad (2.28)$$

Appendix D: Linear Reformulation of Stochastic Program

The probabilistic envelope constrained program can be reformulated into linear programs with Corollary 1, 2, 3, and 4 for different scenarios. In this section, we present linear programs for each scenario, except the one presented in main text (see Section 2.4.4 and 2.4.5).

D.1: Linear Reformulation of Stochastic Program with Proposition 4

When the travel time distribution is explicitly known, the probabilistic envelope constrained program SP_1 and SP_2 can be reformulated as

$$
\begin{aligned}
(\mathrm{SP}_1^R) \max_{x,y,d,z,u,\theta} \quad & \sum_i \sum_j \sum_t \left(r_i - cl_{ij}\right) \hat{d}_{ijt} - \sum_j \left(o_j + cl_{0j}\right) y_j - \sum_t h\hat{z}_t \\
\text{s.t.} \quad & (2.1\text{b}) - (2.1\text{d}), (2.1\text{f}) - (2.1\text{g}), \\
& x_{ijt} \leq \mathbb{I}\left\{ \max_k \Psi^{-1}_{\tilde{\tau}_{ijt}}(\beta(v^{k+\varepsilon})) - \bar{\tau} - v^k \leq 0 \right\}, \forall i, j, t.
\end{aligned}
$$

$$
\begin{aligned}
(\mathrm{SP}_2^R) \max_{x,y,d,z,u,\theta} \quad & \sum_i \sum_j \sum_t \left(r_i - cl_{ij}\right) \hat{d}_{ijt} - \sum_j \left(o_j + cl_{0j}\right) y_j - \sum_t h\hat{z}_t \\
\text{s.t.} \quad & (2.1\text{b}) - (2.1\text{d}), (2.1\text{f}) - (2.1\text{g}) \\
& x_{ijt} \leq \mathbb{I}\left\{ \Psi^{-1}_{\tilde{\tau}_{ijt}}(\beta(v^{k+\varepsilon})) - \bar{\tau} - v^k \leq 0 \right\}, \forall i, j, t, k \in [|\mathscr{K}| + 1 - n, |\mathscr{K}|].
\end{aligned}
$$

Note that $\varepsilon = 0$ for relaxation and $\varepsilon = 1$ for restriction.

D.2: Linear Reformulation of Stochastic Program with Proposition 5

When the travel time distribution is unknown, the SP_1 and SP_2 can be reformulated as

$$
\begin{aligned}
(\mathrm{SP}_1^R) \max_{x,y,d,z,u,\theta} \quad & \sum_i \sum_j \sum_t \left(r_i - cl_{ij}\right) \hat{d}_{ijt} - \sum_j \left(o_j + cl_{0j}\right) y_j - \sum_t h\hat{z}_t \\
\text{s.t.} \quad & (2.1\text{b}) - (2.1\text{d}), (2.1\text{f}) - (2.1\text{g}) \\
& x_{ijt} \leq \mathbb{I}\left\{ \max_k \hat{\tau}_{ijt} + \sqrt{\frac{\beta(v^{k+\varepsilon})}{1 - \beta(v^k)}} \sigma_{ijt} - \bar{\tau} - v^k \leq 0 \right\}, \forall i, j, t.
\end{aligned}
$$

$$
\begin{aligned}
(\mathrm{SP}_2^R) \max_{x,y,d,z,u,\theta} \quad & \sum_i \sum_j \sum_t \left(r_i - cl_{ij}\right) \hat{d}_{ijt} - \sum_j \left(o_j + cl_{0j}\right) y_j - \sum_t h\hat{z}_t \\
\text{s.t.} \quad & (2.1\text{b}) - (2.1\text{d}), (2.1\text{f}) - (2.1\text{g}) \\
& x_{ijt} \leq \mathbb{I}\left\{ \hat{\tau}_{ijt} + \sqrt{\frac{\beta(v^{k+\varepsilon})}{1 - \beta(v^k)}} \sigma_{ijt} - \bar{\tau} - v^k \leq 0 \right\}, \\
& \qquad \forall i, j, t, k \in [|\mathscr{K}| + 1 - n, |\mathscr{K}|].
\end{aligned}
$$

Note that $\varepsilon = 0$ for relaxation and $\varepsilon = 1$ for restriction.

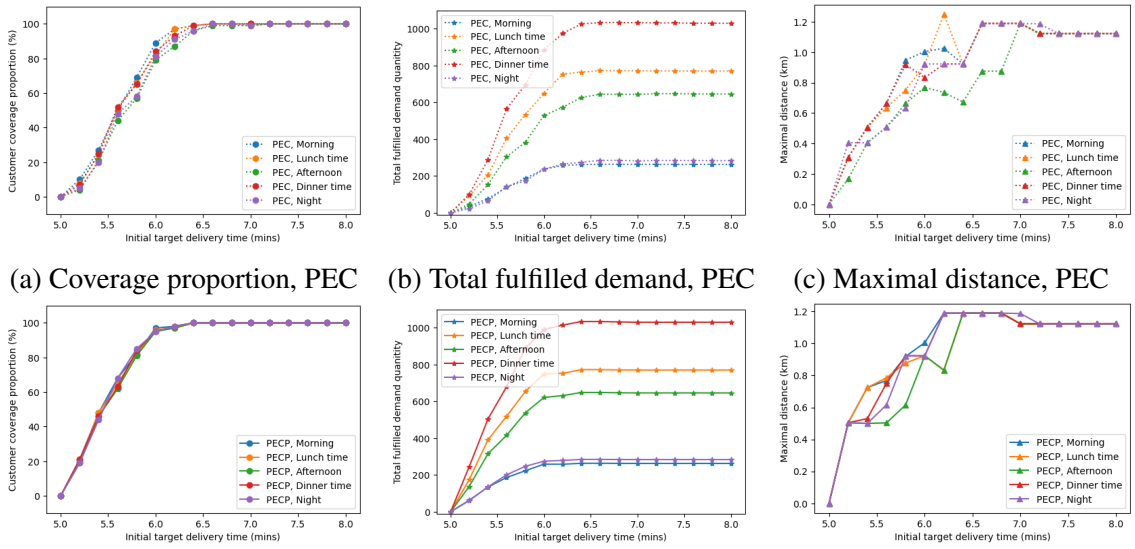
D.3: Linear Reformulation of Stochastic Program with Proposition 6

When the travel time distribution is explicitly known but the period probability distribution is unknown, the SP_1 and SP_2 can be reformulated as

$$
\begin{aligned}
(\mathrm{SP}_1^R) \max_{x,y,d,z,u,\theta} \quad & \sum_i \sum_j \sum_t \left(r_i - cl_{ij}\right) \hat{d}_{ijt} - \sum_j \left(o_j + cl_{0j}\right) y_j - \sum_t h\hat{z}_t \\
\text{s.t.} \quad & (2.1\text{b}) - (2.1\text{d}), (2.1\text{f}) - (2.1\text{g}) \\
& \sum_t q_{it} \left(\sum_j \left[\Psi(\bar{\tau} + v^k - \hat{\tau}_{ijt}) - \beta(v^{k+\varepsilon}) \right] x_{ijt} \right) \geq 0, \forall i,k.
\end{aligned}
$$

$$
\begin{aligned}
(\mathrm{SP}_2^R) \max_{x,y,d,z,u,\theta} \quad & \sum_i \sum_j \sum_t \left(r_i - cl_{ij}\right) \hat{d}_{ijt} - \sum_j \left(o_j + cl_{0j}\right) y_j - \sum_t h\hat{z}_t \\
\text{s.t.} \quad & (2.1\text{b}) - (2.1\text{d}), (2.1\text{f}) - (2.1\text{g}) \\
& \sum_t q_{it} \left(\sum_j \left[\Psi(\bar{\tau} + v^k - \hat{\tau}_{ijt}) - \beta(v^{k+\varepsilon}) \right] x_{ijt} \right) \geq 0, \\
& \qquad \forall i,k \in [|\mathscr{K}| + 1 - n, |\mathscr{K}|].
\end{aligned}
$$

Appendix E: The Detailed Impact of Target Delivery Time



(a) Coverage proportion, PEC (b) Total fulfilled demand, PEC (c) Maximal distance, PEC

(d) Coverage proportion, PECP (e) Total fulfilled demand, PECP (f) Maximal distance, PECP

Figure 2.12 – The impact of initial target delivery time on PEC and PECP

Figure 2.12 illustrates how the initial target delivery time affects results in each period. Across different time periods, the coverage proportion changes in similar trends, with captured demand being proportional to the nominal demand in each period. Additionally, there is a small variation in the maximal distance to travel from micro-depots to customers.

References

Aboolian, Robert, Tingting Cui, and Zuo-Jun Max Shen (2013). "An efficient approach for solving reliable facility location models". In: *INFORMS Journal on Computing* 25.4, pp. 720–729.

Aikens, Charles H (1985). "Facility location models for distribution planning". In: *European Journal of Operational Research* 22.3, pp. 263–279.

Armbruster, Benjamin and Erick Delage (2015). "Decision making under uncertainty when preference information is incomplete". In: *Management Science* 61.1, pp. 111–128.

Ben-Akiva, Moshe and Michel Bierlaire (1999). "Discrete choice methods and their applications to short term travel decisions". In: *Handbook of Transportation Science*. Boston, MA: Springer, pp. 5–33.

Calafiore, Giuseppe Carlo and L El Ghaoui (2006). "On distributionally robust chance-constrained linear programs". In: *Journal of Optimization Theory and Applications* 130.1, pp. 1–22.

Cao, Junyu and Wei Qi (2023). "Stall economy: The value of mobility in retail on wheels". In: *Operations Research* 71.2, pp. 708–726.

Capasso Da Silva, Denise, David A King, and Shea Lemar (2019). "Accessibility in practice: 20-minute city as a sustainability planning goal". In: *Sustainability* 12.1, p. 129.

Chandler, Adam (2022). *America's Need for Speed Never Ends Well*. Last accessed on Aug 01, 2023. URL: https://www.theatlantic.com/technology/archive/2022/05/fast-15-minute-delivery-apps-amazon/661145/.

Chen, Manlu, Ming Hu, and Jianfu Wang (2022). "Food delivery service and restaurant: Friend or foe?" In: *Management Science* 68.9, pp. 6539–6551.

Chen, Ye et al. (2022). "Data-driven robust resource allocation with monotonic cost functions". In: *Operations Research* 70.1, pp. 73–94.

Cheng, Chun, Yossiri Adulyasak, and Louis-Martin Rousseau (2021). "Robust facility location under disruptions". In: *INFORMS Journal on Optimization* 3.3, pp. 298–314.

Dai, Hanjun et al. (2023). “Learning to Optimize with Stochastic Dominance Constraints”. In: *International Conference on Artificial Intelligence and Statistics*. PMLR, pp. 8991–9009.

Dentcheva, Darinka and Andrzej Ruszczyński (2004). “Semi-infinite probabilistic optimization: first-order stochastic dominance constrain”. In: *Optimization* 53.5-6, pp. 583–601.

Deshpande, Vinayak and Pradeep K Pendem (2023). “Logistics performance, ratings, and its impact on customer purchasing behavior and sales in e-commerce platforms”. In: *Manufacturing & Service Operations Management* 25.3, pp. 827–845.

Fatehi, Soraya and Michael R Wagner (2022). “Crowdsourcing last-mile deliveries”. In: *Manufacturing & Service Operations Management* 24.2, pp. 791–809.

Feldman, Pnina, Andrew E Frazelle, and Robert Swinney (2023). “Managing relationships between restaurants and food delivery platforms: Conflict, contracts, and coordination”. In: *Management Science* 69.2, pp. 812–823.

Getir (2022). *Getir: groceries in minutes*. Last accessed on Nov 01, 2023. URL: https://getir.com/us/.

Ghosal, Shubhechyya and Wolfram Wiesemann (2020). “The distributionally robust chance-constrained vehicle routing problem”. In: *Operations Research* 68.3, pp. 716–732.

Goodfood (2022). *Goodfood is in financial trouble and gives up fast delivery (In French)*. Last accessed on Aug 01, 2023. URL: https://www.lapresse.ca/affaires/entreprises/2022-10-14/goodfood-a-des-ennuis-financiers-et-laisse-tomber-la-livraison-rapide.php.

Gorillas (2022). *Grocery app Gorillas drops 10-minute delivery pledge, adds store pick-up option*. Last accessed on Aug 01, 2023. URL: https://nypost.com/2022/02/25/grocery-app-gorillas-drops-10-minute-delivery-pledge-adds-store-pick-up-option/.

Hanasusanto, Grani A et al. (2015). “A distributionally robust perspective on uncertainty quantification and chance constrained programming”. In: *Mathematical Programming* 151, pp. 35–62.

Hildebrandt, Florentin D and Marlin W Ulmer (2022). "Supervised learning for arrival time estimations in restaurant meal delivery". In: *Transportation Science* 56.4, pp. 1058–1084.

Kavuk, Eray Mert et al. (2022). "Order dispatching for an ultra-fast delivery service via deep reinforcement learning". In: *Applied Intelligence*, pp. 1–26.

Laporte, Gilbert, François V Louveaux, and Luc van Hamme (1994). "Exact solution to a location problem with stochastic demands". In: *Transportation Science* 28.2, pp. 95–103.

Li, Yongzhen et al. (2022). "A general model and efficient algorithms for reliable facility location problem under uncertain disruptions". In: *INFORMS Journal on Computing* 34.1, pp. 407–426.

Liu, Sheng, Long He, and Zuo-Jun Max Shen (2021). "On-time last-mile delivery: Order assignment with travel-time predictors". In: *Management Science* 67.7, pp. 4095–4119.

Liu, Sheng and Zhixing Luo (2023). "On-Demand Delivery from Stores: Dynamic Dispatching and Routing with Random Demand". In: *Manufacturing & Service Operations Management* 25.2, pp. 595–612.

Liu, Tianqi et al. (2022). "Robust stochastic facility location: sensitivity analysis and exact solution". In: *INFORMS Journal on Computing* 34.5, pp. 2776–2803.

Luedtke, James (2008). "New formulations for optimization under stochastic dominance constraints". In: *SIAM Journal on Optimization* 19.3, pp. 1433–1450.

Mak, Ho-Yin (2022). "Enabling smarter cities with operations management". In: *Manufacturing & Service Operations Management* 24.1, pp. 24–39.

Merchan, Daniel et al. (2021). *Amazon Last Mile Routing Research Challenge Dataset*. Accessed January 6, 2022. Seattle: Amazon.com. URL: `https://registry.opendata.aws/amazon-last-mile-challenges`.

Moreno, Carlos et al. (2021). "Introducing the "15-Minute City": Sustainability, resilience and place identity in future post-pandemic cities". In: *Smart Cities* 4.1, pp. 93–111.

Mousavi, Kianoush, Merve Bodur, and Matthew J Roorda (2022). "Stochastic last-mile delivery with crowd-shipping and mobile depots". In: *Transportation Science* 56.3, pp. 612–630.

Peng, Chun, Erick Delage, and Jinlin Li (2020). "Probabilistic envelope constrained multiperiod stochastic emergency medical services location model and decomposition scheme". In: *Transportation Science* 54.6, pp. 1471–1494.

Perakis, Georgia and Guillaume Roels (2006). "An analytical model for traffic delays and the dynamic user equilibrium problem". In: *Operations Research* 54.6, pp. 1151–1171.

Postek, Krzysztof et al. (2018). "Robust optimization with ambiguous stochastic constraints under mean and dispersion information". In: *Operations Research* 66.3, pp. 814–833.

Reed, Sara, Ann Melissa Campbell, and Barrett W Thomas (2022). "The value of autonomous vehicles for last-mile deliveries in urban environments". In: *Management Science* 68.1, pp. 280–299.

Repko, Melissa (2021). *Ultrafast grocery delivery has exploded in New York City. Your town could be next*. Last accessed on Aug 01, 2023. URL: https://www.cnbc.com/2021/10/21/gopuff-gorillas-and-others-flood-new-york-with-instant-delivery-options.html.

Salari, Nooshin, Sheng Liu, and Zuo-Jun Max Shen (2022). "Real-time delivery time forecasting and promising in online retailing: when will your package arrive?" In: *Manufacturing & Service Operations Management* 24.3, pp. 1421–1436.

Senzamici, Peter (2024). *Getir, grocery app that bought FreshDirect, owes millions of dollars in NYC back rent: lawsuits*. Last accessed on Feb 01, 2025. URL: https://nypost.com/2024/08/13/us-news/grocery-app-that-bought-freshdirect-owes-millions-of-dollars-in-nyc-back-rent-lawsuits/?utm_source=chatgpt.com.

Shehadeh, Karmel S (2023). "Distributionally robust optimization approaches for a stochastic mobile facility fleet sizing, routing, and scheduling problem". In: *Transportation Science* 57.1, pp. 197–229.

Shen, Zuo-Jun Max, Roger Lezhou Zhan, and Jiawei Zhang (2011). "The reliable facility location problem: Formulations, heuristics, and approximation algorithms". In: *INFORMS Journal on Computing* 23.3, pp. 470–482.

Snyder, Lawrence V (2006). "Facility location under uncertainty: a review". In: *IIE Transactions* 38.7, pp. 547–564.

Statista (2023). *Market insights into quick commerce of online food and grocery delivery (Worldwide)*. Last accessed on Dec 15, 2023. URL: https://www.statista.com/outlook/dmo/online-food-delivery/grocery-delivery/quick-commerce/worldwide.

Talluri, Kalyan T, Garrett Van Ryzin, and Garrett Van Ryzin (2004). *The Theory and Practice of Revenue Management*. Vol. Vol 3. Boston, MA: Springer.

Verter, Vedat (2011). "Uncapacitated and capacitated facility location problems". In: *Foundations of Location Analysis*. New York, NY: Springer, pp. 25–37.

Wang, Ruxian (2021). "Consumer choice and market expansion: Modeling, optimization, and estimation". In: *Operations Research* 69.4, pp. 1044–1056.

Xu, Huan, Constantine Caramanis, and Shie Mannor (2012). "Optimization under probabilistic envelope constraints". In: *Operations Research* 60.3, pp. 682–699.

Zhang, Wenchang et al. (2022). "Reducing Traffic Incidents in Meal Delivery: Penalize the Platform or its Independent Drivers?" In: *Kelley School of Business Research Paper*, No. 2022–09, Available at SSRN: https://ssrn.com/abstract=4231746.

Chapter 3

Learning-to-optimize for Consolidation and Transshipment in Multi-store Order Delivery

Abstract

This study investigates multi-store order delivery services that allow customers to place orders from multiple stores for home delivery. We first consider the case of separated-order delivery, where orders from different stores are processed and delivered separately without consolidation. To enhance customer convenience and operational efficiency, we then introduce a consolidated-order delivery option. This option enables customers to place a single order across multiple stores and receive all items in one combined delivery. While consolidated-order delivery increases convenience, assigning a single driver to pick up and deliver items from multiple stores can increase delivery times due to the additional routing for visiting multiple stores. To address this shortcoming, we propose a consolidated-order delivery system with transshipment, which allows drivers to coordinate and transfer orders at transshipment nodes. We develop a mixed-integer linear program (MILP) for the multi-store order problem that can model different delivery systems,

including separated-order delivery and consolidated-order delivery with or without transshipment. This formulation presents significant computational challenges due to the large number of constraints and variables arising from the routing decisions and time variables. To overcome these challenges, we adopt a learning-to-optimize approach that integrates machine learning and optimization. Four methods are implemented for learning allocation decisions, including Nearest Driver Allocation, Driver Assignment Neural Network (DANN), Driver Classification Neural Network (DCNN), and Graph-based Neural Network (GNN). Our experimental study reveals that GNN consistently performs the best in terms of accuracy, optimality gap, efficiency, and scalability to larger problem instances beyond the training set. The DCNN and DANN, on the other hand, are effective with a sufficiently large training set and they perform well particularly when the instance scale in the testing set matches those in the training set. We conduct experiments across four U.S. regions using the learning-to-optimize method in a realistic setting with dynamic customer arrivals. We find that the consolidated-order delivery with transshipment, coupled with short-duration waiting strategy, consistently delivers superior performance, yielding shorter order completion times and reduced driver travel times through effective spatial and temporal consolidation. The optimal waiting strategy varies depending on customer arrival rates and driver availability relative to customer demand.

3.1 Introduction

Online meal and grocery delivery services emerged in the early 2000s and experienced substantial growth soon thereafter. This expansion accelerated between 2010 and 2020, with a dramatic surge in 2020 due to the global pandemic. By 2024, the online delivery market has reached a volume of USD 1.22 trillion, with projections indicating further growth to USD 1.79 trillion by 2028 (Statista 2024).

To access a wider variety of products or enjoy the convenience of home delivery, customers increasingly rely on online delivery services for items from multiple types of stores, such as meals from restaurants, fresh produce from grocery stores, or emer-

gency medicine from pharmacies. Companies such as UberEats and DoorDash provide a *multi-store order service* that caters to customers who prefer the convenience of home delivery over visiting multiple stores in person, especially when they are pressed for time. These platforms conventionally adopt *separated-order delivery* (SOD), a system where orders from different stores are treated as distinct transactions and delivered individually. Customers place separate orders from each store, incurring separate delivery fees for each, while drivers handle pickups and deliveries for each order independently. Although this model provides convenient access to products from multiple stores, it can be inefficient and costly due to multiple delivery fees and minimum order requirements for each store. Recently, companies such as DoubleDash (DoorDash), Instacart, and Epipresto have started offering what we term *consolidated-order delivery* (COD) services. These services allow customers to order from multiple stores in a single transaction, receiving all items as a combined delivery, typically without an added delivery fee. For instance, DoubleDash (2023) and Instacart (2022) allow customers to add items from nearby retailers to their original order without an additional delivery charge, ensuring that all items are delivered together by the same driver. Similarly, Epipresto (2023) enables customers to shop from multiple stores at in-store prices with a fixed delivery fee, regardless of distance. This consolidated-order model offers multiple benefits: customers save on delivery fees, enjoy the convenience of placing one order for all needed products, and have a single delivery experience. Meanwhile, companies may benefit from a decreased need for drivers, reduced delivery costs, and increased sales. Despite these benefits, consolidated-order delivery can introduce new challenges to delivery efficiency. A single driver responsible for picking up and delivering items from multiple stores may face extended delivery times, particularly when fulfilling orders from far apart store locations to multiple customers. To overcome these limitations and optimize the delivery process, we propose a model that divides multi-store orders among several drivers, who can then transfer orders through transshipment. In the *consolidated-order delivery with transshipment* (CODT) system, a customer's multi-store order is distributed among multiple drivers, each responsible for pickups from different stores. These orders are then transferred at selected transship-

ment nodes, potentially assigned to a different driver, and delivered as one package to the customer's doorstep. This collaborative approach offers the combined benefits of consolidation with enhanced efficiency, reducing overall delivery time and improving route management for drivers handling multi-store, multi-customer orders.

To quantify the value of enabling order consolidation and transshipment for fulfilling multi-store orders in a combined delivery, we develop a mixed integer linear program (MILP) that models three delivery systems: separated-order delivery, consolidated-order delivery without transshipment, and consolidated-order delivery with transshipment. The MILP for multi-store order services presents substantial computational challenges. From the drivers' perspective, this problem resembles a complex vehicle routing problem involving multiple store and customer locations. Conversely, the journey of each item can be viewed as a shortest path problem within a network defined by the drivers' delivery routes, starting at the store and ending at the customer. This requires precise tracking of each item and driver's arrival and departure times to meet time windows and ensure timely deliveries. To mitigate the significant solution times for solving the MILP directly, we adopt a learning-to-optimize approach. This method replaces the traditional optimization process with a machine learning-based approximation (i.e., an optimization proxy). The learning model estimates allocation decisions, identifying which drivers should pick up items from stores and which should deliver them to customers. These estimated allocations are subsequently refined through an MILP with a reduced search space, significantly enhancing computational efficiency and enabling practical implementation. We evaluate three delivery systems in a simulated dynamic setting, where both customer arrivals and the optimization process occur in real-time, making time and space consolidation crucial. The system can optimize deliveries immediately as each customer arrives or wait to batch multiple customers for joint optimization. This analysis helps identify the most efficient delivery system under various waiting strategies.

The four main contributions of this paper are as follows:

- We investigate three delivery systems for multi-store order services, where cus-

tomers place orders across multiple stores for home delivery. Beyond separated-order delivery, in which each order from each store is handled separately, we propose consolidated-order delivery, with or without transshipment. Consolidated-order delivery enables customers to order from multiple stores in a single transaction and receive all items as a combined delivery, while transshipment further allows for driver coordination and order transshipment to enhance delivery efficiency.

- We formulate a mixed-integer linear program to model the multi-store order problem (MOP) across all three delivery systems. To address the computationally demanding problem of optimizing both routing and transfer decisions, we adopt a learning-to-optimize approach that integrates machine learning with optimization. In this framework, the estimated allocation decisions act as a lower bound within the optimization process. This process not only refines the allocation decisions but also optimizes additional decisions that are not directly learned. By offloading a portion of the computational workload to the offline phase, this approach enables the near real-time generation of high-quality solutions.

- We implement four methods to obtain the estimated allocation plans efficiently. Nearest Driver Allocation (NDA) identifies the closest driver for each customer. Driver Assignment Neural Network (DANN) uses binary predictions to assign customers to drivers, treating each driver independently. Driver Classification Neural Network (DCNN) frames the allocation of customers to drivers as a classification problem, with each driver representing a unique class. Finally, Graph-based Neural Network (GNN) models each instance as a graph and treats customer-to-driver allocation as edge labels to be learned.

- We conduct numerical experiments across various U.S. regions with varying customer arrival rates, comparing the performance of our three delivery systems and four learning methods. The following insights are derived:

  - By applying a learning-to-optimize approach, we enhance the solution effi-

ciency of the multi-store order problem and obtain high-quality solutions. This approach involves learning allocation decisions for assigning drivers to visit nodes based on historically optimal samples, which narrows the feasible solution space and facilitates effective decision-making within this refined search area.

- The GNN outperforms other methods, achieving a better balance between optimality gap and solution time while also scaling well to larger instances beyond the training set, due to its ability to facilitate information exchange. However, it requires significant effort to collect sufficient historical samples, as each instance corresponds to a single graph. The NDA is simple to implement, requiring no training or historical samples, but struggles with clustered customer locations. Both the DCNN and DANN perform well when the test instance scale matches the training data but face challenges adapting to larger-scale instances.

- We identify a balance between batching efficiency and individual order fulfillment. As more orders are batched for joint optimization, delivery times rise due to longer routes, but wait times initially drop significantly because batching makes better use of drivers. However, beyond a certain point, extending the re-optimization interval leads to increased wait times as customers experience longer delays before being served. Consequently, the total order completion time, which includes both wait and delivery times, initially decreases, reaching a minimum before increasing again. This indicates the existence of an optimal waiting strategy, influenced by factors such as customer arrival rates and the ratio of drivers to customers.

- Consolidated-order delivery shows increasing efficiency over separated-order delivery as customer scale grows, since consolidating orders from multiple stores effectively reduces both delivery and driver travel times. The COD with transshipment and a short-duration waiting strategy consistently achieves the

best performance in delivery and travel times, regardless of customer or driver scales.

The rest of the paper is organized as follows. We review related work in Section 3.2. Next, we define the problem setting, describe our three delivery systems, and present the models for these systems in Section 3.3. We develop the learning-to-optimize method in Section 3.4 and report the results of numerical studies comparing the performance of our proposed models and their solution quality in Section 3.5. Finally, we conclude with managerial insights in Section 3.6.

3.2 Literature Review

In this section, we review the main studies relevant to our research from three points of view: last-mile delivery, pickup and delivery problem, and learning-based optimization.

3.2.1 Last-mile Delivery

Last-mile delivery, which refers to the delivery from distribution points or transshipment nodes to customer locations, is closely related to online food and grocery delivery. Savelsbergh and Van Woensel (2016) review and discuss various current and anticipated challenges and opportunities in last-mile logistics. They claim that customers are increasingly demanding in terms of price, quality, time, and sustainability. They also highlight the importance of batching in city logistics, noting that delaying dispatch can allow for additional orders, which may lead to more cost-effective delivery routes. However, this approach also increases the risk of delayed order fulfillment.

In the field of on-time last-mile delivery for food and groceries, most research focuses on either improving delivery times to enhance service quality or reducing travel costs to improve efficiency. This is achieved through demonstrating the importance of on-demand delivery using empirical evidence, optimizing the assignment of drivers to customers or driver routing for dispatching items, or proposing novel business models and applying

learning techniques to improve the overall efficiency of the delivery system. From an empirical perspective, Mao et al. (2022) develop data analytics models to improve the delivery performance of a meal delivery platform, emphasizing the need for the platform to provide superior delivery performance without increasing delivery fees. Similarly, Li and G. Wang (2024) highlight that restaurants increasingly rely on on-demand delivery platforms such as DoorDash and Uber Eats to reach customers and fulfill takeout orders, providing empirical evidence of the overall benefits for restaurants using these platforms. From an optimization perspective, Liu, He, and Max Shen (2021) investigate the impact of delivery data on the on-time performance of food delivery services, developing an order assignment problem with travel-time predictors. Carlsson et al. (2024) characterize the optimal region partitioning policy to minimize the expected delivery time of customer orders in a stochastic and dynamic setting, assigning each driver to a specific subregion to ensure drivers are dispatched only within their territories. In terms of novel business models, Cao and W. Qi (2023) propose selling groceries in public spaces using self-driving mini grocery stores to enhance mobility, proximity, and flexibility of grocery delivery by avoiding the last 100 meters. Additionally, Raghavan and Zhang (2024) explore coordinated logistics by evaluating the value of using aides who can assist drivers in last-mile delivery. The aide can either help the driver prepare and deliver packages, thus reducing service time at a given stop, or work independently at a location, allowing the driver to deliver packages at other sites before returning. From the learning perspective, Hildebrandt and Ulmer (2022) present offline and online-offline estimation approaches for estimating arrival times, finding that accurate arrival predictions not only improve service perception but also enhance the overall delivery system by guiding customer selections, resulting in faster deliveries. Auad, Erera, and Savelsbergh (2024) study the dynamic management of courier capacity in a meal delivery system, proposing a deep reinforcement learning approach to balance the costs of adding couriers and the degradation of service quality due to insufficient delivery capacity.

We also aim to provide efficient on-demand and on-time delivery for food and groceries. Our goal is to ensure fast delivery times for customers while also minimizing total

travel times. In our proposed setting, the platform allows customers to place orders from multiple stores and receive all items in a single delivery, enhancing convenience for customers. Additionally, drivers can manage parts of the delivery process and coordinate at intermediate points to transfer orders, with any driver able to complete deliveries from these points. However, this setup may lead to a complex pickup and delivery problem, potentially causing inefficiencies in the delivery system. Therefore, we will discuss the problem formulation and explore how to address it effectively.

3.2.2 Pickup and Delivery Problem

The Pickup and Delivery Problem (PDP) is a classical combinatorial optimization problem concerned with optimizing vehicle routes for picking up and delivering goods or passengers at various locations. Its primary goal is the efficient movement of vehicles to minimize transportation costs. The PDP is an extension of the Vehicle Routing Problem (VRP), first introduced by Dantzig and Ramser (1959).

There are several variants of the PDP, including single-vehicle PDP, multiple-vehicle PDP, dynamic PDP, many-origin-to-many-destination PDP, PDP with time windows, PDP with split delivery, and PDP with transshipment, among others. Berbeglia, Cordeau, and Laporte (2010) provide a general framework for dynamic PDPs, where objects or people need to be collected and delivered in real-time, with requests revealed over time. They also explain how to design waiting strategies and assess their performance. Additionally, Koç, Laporte, and Tükenmez (2020) conduct a detailed survey of vehicle routing problems with simultaneous pickup and delivery, where goods must be transported from various origins to different destinations, satisfying both delivery and pickup demands concurrently.

In the context of the many-to-many PDP with transshipment that we are interested in, the pickup loads and delivery processes can be split among different drivers, provided that drivers meet at transshipment nodes to exchange items. Specifically, one vehicle collects the load at the pickup location, drops it at a transshipment point, and then an-

other vehicle transports the load to the delivery location. In terms of empirical studies, Mitrović-Minić and Laporte (2006) study the pickup and delivery problem with time windows and transshipment, presenting an empirical study on the usefulness of transshipment points. They propose a heuristic for the static problem setting and evaluate the benefits of transshipment. Nowak, Ergun, and White III (2009) demonstrate that splitting loads to complete certain deliveries in multiple trips, rather than in a single trip, is advantageous for the PDP. They characterize the real-world environments where split loads are most beneficial through empirical analysis. In terms of optimization, Rieck, Ehrenberg, and Zimmermann (2014) consider the many-to-many location-routing problem with inter-hub transport and multi-commodity pickup-and-delivery. To tackle medium- and large-scale instances, they develop a fix-and-optimize heuristic and a genetic algorithm that construct promising solutions within an appropriate time limit. While this problem is similar to our study, it does not account for the importance of time windows, which further complicates the setting. Rais, Alvelos, and Carvalho (2014) explore the pickup and delivery problem with time windows and transshipments from an optimization perspective. However, due to the complexity of the problem, they only consider instances with 10 or 14 nodes, with five order requests and one transfer station. Lyu and Yu (2023) revise the formulations proposed by Rais, Alvelos, and Carvalho (2014), successfully solving instances with 25 requests and two transfer stations to optimality within one hour. Both studies consider the flexibility of drivers picking up items from different locations, exchanging them at transshipment nodes, and delivering them to customers. However, they do not address the possibility that orders originated from multiple pickup points can be combined together and delivered as a single package, nor do they recognize that any node visited by drivers could serve as a transshipment point for exchanging items, provided time restrictions are respected. Ulmer et al. (2021) consider a stochastic dynamic pickup and delivery problem, model it as a route-based Markov decision process, and propose an anticipatory customer assignment policy that significantly enhances service. However, their work focuses on single-courier direct deliveries without transshipment, making it less applicable to multi-driver coordination and exchange-based delivery networks like ours. Su

et al. (2023) tackle a pickup and delivery problem with crowdsourced bids and transshipment in last-mile delivery, where requests are fulfilled using either the company's fleet or crowdshippers with a small compensation via transshipment facilities. Although it optimizes the company's vehicle routes and bid selection, it does not account for the optimal routing of crowdshippers and imposes several restrictions on service requests.

The problem we investigate is a many-origin-to-many-destination pickup and delivery problem with time windows, allowing coordination and transshipment among drivers to fulfill orders placed from multiple stores. To accurately model this complex setting, we will consider the allocation decisions of orders to drivers, the transshipment nodes where drivers meet and exchange items, the routing decisions for each driver and each order, and the time variables associated with driver and order arrivals and departures.

3.2.3 Machine Learning-based Optimization

Machine learning (ML) has emerged as a transformative tool in operations research (OR) and operations management, enabling innovative approaches to solve complex decision-making problems. The integration of ML in OR can broadly be categorized into two areas: using ML to design and enhance optimization models, and employing optimization proxies to accelerate or approximate solutions.

In the first category, ML is widely used to design optimization models, improve data quality, and enhance solution procedures. Sadana et al. (2024) provide a comprehensive review of contextual stochastic optimization, unifying diverse methodologies for decision-making under uncertainty. They categorize key approaches into decision rule optimization, sequential learning and optimization, and integrated learning and optimization. Maragno et al. (2023) propose a pipeline where constraints and objectives are learned directly from data and embedded into optimization models. Similarly, S. Wang, Delage, and Coelho (2024) enhance inputs to vehicle routing problems using techniques like k-nearest neighbor and kernel density estimation. In practical applications, Liu, He, and Max Shen (2021) integrate ML-based travel-time predictors into order-assignment

optimization for food delivery, improving on-time performance by refining parameter estimates. Additionally, Hong, Huang, and Lam (2021) propose a statistical framework for robust optimization that uses machine learning to provide more accurate parameter estimates, enhancing optimization outcomes.

In the second category, optimization proxies use ML to approximate solutions, accelerate computational processes, or enable inverse optimization. Van Hentenryck (2021) distinguishes between two main approaches: end-to-end learning, which uses proxies to directly approximate optimal solutions, and learning-to-optimize, which accelerates existing optimization algorithms for enhanced efficiency. Kotary, Fioretto, et al. (2021) review hybrid methods that integrate combinatorial solvers with ML to provide fast, approximate solutions and support logical inference. Bogyrbayeva et al. (2024) provide a systematic overview of machine learning methods applied to solve vehicle routing problems. Several studies highlight the potential of learning-to-optimize for speeding up solution generation. For example, Ojha et al. (2023) develop ML proxies to predict loads and ensure constraint compliance in dynamic load planning, significantly improving decision-making speed. Similarly, Julien, Postek, and Birbil (2024) introduce a machine-learning-based node selection strategy that accelerates robust optimization by identifying high-quality solutions faster. In two-stage stochastic programming, Larsen, Lachapelle, et al. (2022) and Larsen, Frejinger, et al. (2024) use supervised learning to approximate computationally intensive second-stage solutions, reducing solution time while maintaining accuracy in repeated problem-solving tasks. Kotary, Di Vito, et al. (2024) propose a learning-to-optimize-from-features framework, aligning traditional methods with the predict-then-optimize paradigm. Additionally, M. Qi et al. (2023) demonstrate the impact of end-to-end learning by developing a multi-period inventory replenishment model that directly outputs optimal replenishment decisions from input features. For inverse optimization, Özarık, Costa, and Florio (2024) leverage ML to estimate key parameters for last-mile delivery route optimization, illustrating how predictive modeling can inform complex decision-making.

In this work, we apply the learning-to-optimize method to efficiently address our com-

putationally challenging problem. Initially, we use machine learning techniques, including neural networks and graph-based neural networks, to estimate allocation decisions. These estimations then serve as inputs to an optimization process, which refines the allocation decisions and makes additional decisions, such as optimal routing. Unlike conventional learning-to-optimize methods, our approach maintains flexibility by treating the estimated decisions as lower bounds rather than fixed inputs. Compared to directly learning routing decisions, this method has low-dimensional outputs, enhancing both accuracy and efficiency. Rather than outputting fixed routing decisions, it allows for the exploration of routing solutions within a refined search space during the optimization process. Additionally, our work focuses on both efficiency and high-quality solutions, while also evaluating the scalability of different learning algorithms across various cases.

3.3 Multi-store Order Delivery Problem

In this section, we first introduce our three delivery systems of interest and present the problem definition in Section 3.3.1. We then formulate the mathematical model for the multi-store order delivery problem under various delivery systems in Section 3.3.2, followed by the dynamic problem description incorporating waiting strategies in Section 3.3.3.

3.3.1 Delivery System Description and Problem Definition

Due to the variety of store types, product availability, and special pricing, customers often purchase products from multiple stores, such as meals from restaurants, fresh produce from grocery stores, or emergency medicine from pharmacies. To avoid the inconvenience of visiting multiple stores, customers increasingly rely on online ordering and delivery platforms, creating the necessity for multi-store order delivery services. We consider three delivery systems for managing multi-store order deliveries:

(1) *Separated-order delivery (SOD):* Customers place separate orders from different

stores, pay a delivery fee for each order, and receive individual deliveries. Orders from each store are handled independently, with drivers managing both the pickup and delivery processes (Figure 3.1(a)). This system serves as a baseline for comparison in the absence of consolidation.

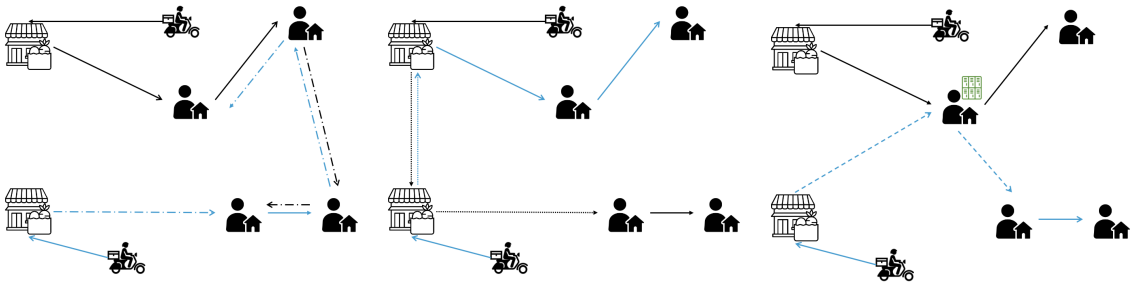
(2) *Consolidated-order delivery (COD):* Customers place a combined order from multiple stores in a single transaction and receive a combined delivery, typically without additional delivery fees. A single driver collects items from several stores and delivers them to the customer (Figure 3.1(b)). This system consolidates orders from different stores and serves as a baseline in the absence of transshipment.

(3) *Consolidated-order delivery with transshipment (CODT):* Similar to COD, customers place a consolidated order from multiple stores and receive a combined delivery. However, in this system, different drivers can pick up items from different stores and can transfer them at selected transshipment nodes. A single driver then completes the final delivery to the customer's doorstep (Figure 3.1(c)). Transshipment can occur at various locations, such as stores or customer locations, where drivers can drop off or pick up items and items can be temporarily held without the presence of drivers. In some cases, customers acting as transshipment nodes may receive multiple deliveries, while others receive a single delivery.

We define the problem that accommodates these three delivery systems as follows:

Definition 7 *The Multi-store Order Problem (MOP) is an optimization problem addressing the fulfillment of customer orders placed across multiple stores using a limited number of drivers. The objective is to jointly minimize a weighted sum of the latest delivery time for serving each order and the total driver travel time for fulfilling all requests. This is achieved by addressing key decisions, including allocating drivers for pickup and delivery, planning driver routes with time and capacity restrictions, and selecting transshipment nodes for transferring items among drivers. In addition to handling orders from different stores separately, MOP also incorporates consolidation, allowing orders from multiple stores to be grouped and handled together, and transshipment, permitting partial*

order transfers among drivers at specified nodes.



(a) Separated-order delivery (b) Consolidated-order delivery (c) COD with transshipment

Notes. This figure illustrates an instance with two drivers, two stores, and four customers. The same set of customers is served by the same set of drivers across all three systems. The route of one driver is represented by a black line, while the route of another driver is shown in blue. The solid lines are the same for each system, while the dashed lines highlight the differences.

Figure 3.1 – Delivery systems for multi-store order services

3.3.2 Mathematical Model

We formulate the multi-store order problem for all three delivery systems using a mixed-integer linear programming model. This MILP is defined on an undirected network $\mathscr{G} = (\mathscr{N}, \mathscr{A})$, where $\mathscr{N}$ represents the set of nodes and $\mathscr{A}$ denotes the set of arcs. Customers are denoted by $i \in \mathscr{I}$, stores by $j \in \mathscr{J}$, and drivers by $k \in \mathscr{K}$. The origin node of driver k, representing their starting location, is denoted as o_k, and the set $\mathscr{O}$ comprises the origin nodes of all drivers. The node set $\mathscr{N}$ includes all customer, store, and driver starting locations, i.e., $\mathscr{N} = \mathscr{I} \cup \mathscr{J} \cup \mathscr{O}$. The binary parameter e_{ij} indicates whether customer i orders from store j, and the item ordered by customer i from store j is indexed by ij. The size of item ij is p_{ij}, and the capacity of driver k is q^k. The travel time between nodes n and n' is denoted by $t_{nn'}$, and it satisfies the triangle inequality: $t_{nn'} \leq t_{nn''} + t_{n''n'}, \forall n, n', n''$. The time window for customer i is given by $[\alpha_i, \beta_i]$. The decision variables are as follows. Let z_n^k be 1 if and only if driver k visits node n, $x_{nn'}^k$ be 1 if and only if driver k travels from node n to node n', $y_{nn'}^{ij}$ be 1 if and only if item ij travels from node n to node n' during the trip, v_n^{kij+} be 1 if and only if item ij arrives at node n via driver k, and

v_n^{kij-} be 1 if and only if item ij departs from node n via driver k. If item ij is served by different drivers for arrival and departure at the same node n, it indicates that node n acts as a transshipment point where the drivers meet and transfer item ij. We assume that any node n can potentially serve as a transshipment point with sufficient capacity to temporarily hold the items. The selection of transshipment points will be determined implicitly through the variables v_n^{kij+} and v_n^{kij-}. The continuous variables are as follows: τ_n^{k+} is the time at which driver k arrives at node n, τ_n^{k-} is the time at which driver k departs from node n, and λ_n^{ij} is the arrival time of item ij at node n. A summary of the notation is provided in Appendix A.

The model for the multi-store order problem is presented as follows:
$\mathrm{M}(\mathscr{I}, \mathscr{J}, \mathscr{K}) =$

$$\min_{\substack{x,y,z,\\ \tau,v,\lambda}} \quad \rho_1 \max_{\substack{k\in\mathscr{K}\\ i\in\mathscr{I}}} \left\{\tau_i^{k+}\right\} + \rho_2 \sum_{k\in\mathscr{K}} \sum_{n\in\mathscr{N}} \sum_{\substack{n'\in\mathscr{N},\\ n'\neq n}} t_{nn'} x_{nn'}^k \tag{3.1a}$$

$$\text{s.t.} \quad \sum_{k\in\mathscr{K}} v_j^{kij-} \geq e_{ij}, \quad \forall i \in \mathscr{I}, j \in \mathscr{J} \tag{3.1b}$$

$$\sum_{k\in\mathscr{K}} v_i^{kij+} \geq e_{ij}, \quad \forall i \in \mathscr{I}, j \in \mathscr{J} \tag{3.1c}$$

$$\sum_{k\in\mathscr{K}} v_n^{kij+} = \sum_{k\in\mathscr{K}} v_n^{kij-}, \quad \forall i \in \mathscr{I}, j \in \mathscr{J}, n \in \mathscr{N} : n \neq i, j \tag{3.1d}$$

$$z_{o_k}^k \geq z_n^k \geq \max\{v_n^{kij+}, v_n^{kij-}\}, \forall k \in \mathscr{K}, i \in \mathscr{I}, j \in \mathscr{J}, n \in \mathscr{N} \tag{3.1e}$$

$$\sum_{n'\in\mathscr{N}, n'\neq n} x_{n'n}^k = z_n^k, \quad \forall n \in \mathscr{N}, k \in \mathscr{K} \tag{3.1f}$$

$$\sum_{n'\in\mathscr{N}, n'\neq n} x_{nn'}^k = z_n^k, \quad \forall n \in \mathscr{N}, k \in \mathscr{K} \tag{3.1g}$$

$$\tau_{n'}^{k+} \geq x_{nn'}^k (\tau_n^{k-} + t_{nn'}), \quad \forall n, n' \in \mathscr{N} : n \neq n', k \in \mathscr{K} \tag{3.1h}$$

$$\tau_n^{k-} \geq \tau_n^{k+}, \quad \forall n \in \mathscr{N} \setminus \mathscr{O}, k \in \mathscr{K} \tag{3.1i}$$

$$\sum_{n'\in\mathscr{N}, n'\neq j} y_{jn'}^{ij} = e_{ij}, \quad \forall i \in \mathscr{I}, j \in \mathscr{J} \tag{3.1j}$$

$$\sum_{n'\in\mathscr{N}, n'\neq j} y_{n'i}^{ij} = e_{ij}, \quad \forall i \in \mathscr{I}, j \in \mathscr{J} \tag{3.1k}$$

$$\sum_{n'\in\mathscr{N}, n'\neq n} y_{nn'}^{ij} = \sum_{n'\in\mathscr{N}, n'\neq n} y_{n'n}^{ij}, \quad \forall i \in \mathscr{I}, j \in \mathscr{J}, n \in \mathscr{N}, n \neq i, j \tag{3.1l}$$

$$y_{nn'}^{ij} \leq \sum_{k\in\mathscr{K}} x_{nn'}^{k}, \quad \forall i \in \mathscr{I}, j \in \mathscr{J}, n, n' \in \mathscr{N} : n' \neq n \tag{3.1m}$$

$$y_{no_k}^{ij} \leq 0, \quad \forall i \in \mathscr{I}, j \in \mathscr{J}, n \in \mathscr{N}, k \in \mathscr{K} \tag{3.1n}$$

$$\lambda_{n'}^{ij} \geq y_{nn'}^{ij}(\lambda_n^{ij} + t_{nn'}), \quad \forall k \in \mathscr{K}, n \in \mathscr{N} : n \neq n', n' \in \mathscr{N} \cup \{k'\} \tag{3.1o}$$

$$v_n^{kij+} \geq \min\left\{x_{nn'}^{k}, y_{nn'}^{ij}\right\}, \quad \forall i \in \mathscr{I}, j \in \mathscr{J}, n, n' \in \mathscr{N} : n \neq n', k \in \mathscr{K} \tag{3.1p}$$

$$v_{n'}^{kij-} \geq \min\left\{x_{nn'}^{k}, y_{nn'}^{ij}\right\}, \quad \forall i \in \mathscr{I}, j \in \mathscr{J}, n, n' \in \mathscr{N} : n \neq n', k \in \mathscr{K} \tag{3.1q}$$

$$\lambda_n^{ij} \geq v_n^{kij+}\tau_n^{k+}, \quad \forall i \in \mathscr{I}, j \in \mathscr{J}, n \in \mathscr{N}, k \in \mathscr{K} \tag{3.1r}$$

$$\tau_n^{k-} \geq v_n^{kij-}\lambda_n^{ij}, \quad \forall i \in \mathscr{I}, j \in \mathscr{J}, n \in \mathscr{N}, k \in \mathscr{K} \tag{3.1s}$$

$$\alpha_i \leq \lambda_i^{ij} \leq \beta_i, \quad \forall i \in \mathscr{I}, j \in \mathscr{J} \tag{3.1t}$$

$$\sum_{i\in\mathscr{I}} \sum_{j\in\mathscr{J}} p_{ij} v_n^{kij-} \leq q^k, \quad \forall n \in \mathscr{N}, k \in \mathscr{K} \tag{3.1u}$$

$$x_{nn'}^{k}, y_{nn'}^{ij}, z_n^k, v_n^{kij+}, v_n^{kij-} \in \{0,1\}, \quad \forall i \in \mathscr{I}, j \in \mathscr{J}, n, n' \in \mathscr{N}, k \in \mathscr{K} \tag{3.1v}$$

$$\tau_n^{k+}, \tau_n^{k-}, \lambda_n^{ij} \geq 0, \quad \forall i \in \mathscr{I}, j \in \mathscr{J}, n \in \mathscr{N}, k \in \mathscr{K}. \tag{3.1w}$$

The objective function (3.1a) aims to minimize the weighted average of the latest delivery time and the total travel time for serving all orders, where ρ_1 is the weight assigned to the latest delivery time and ρ_2 is the weight assigned to the total travel time. Note that to prioritize on-time delivery, ρ_1 should be set significantly larger than ρ_2, while a relatively small value for ρ_2 is sufficient to discourage unnecessary long travel times. The term $\max_{k\in\mathscr{K}, i\in\mathscr{I}}\left\{\tau_i^{k+}\right\}$ can be equivalently replaced by a new variable l, along with the constraints $l \geq \tau_i^{k+}, \forall i \in \mathscr{I}, k \in \mathscr{K}$.

Constraints (3.1b) and (3.1c) stipulate that if customer i orders items from store j (i.e., $e_{ij} = 1$), item ij should be picked up from store j by a driver and delivered to customer i by a driver. Constraints (3.1d) ensure that if item ij arrives at node n (except for destination i and origin j), it must leave this node after being picked up by a driver. Constraints (3.1e) indicate that if driver k serves item ij by passing through node n, then driver k must leave its origin location o_k and also visit node n. Constraints (3.1f) and (3.1g) impose degree constraints on the nodes visited by driver k. Constraints (3.1h) specify the arrival and departure times of driver k at node n. If driver k travels from node n to node n' (i.e.,

$x^k_{nn'} = 1$), the arrival time of driver k at node n' must be no earlier than the departure time from the previous node n, accounting for the travel time from node n to node n'. These constraints can be linearized into $\tau^{k-}_n + t_{nn'} - \tau^{k+}_{n'} \leq M_1(1 - x^k_{nn'})$, where M_1 should be sufficiently large to ensure that $M_1 \geq \tau^{k-}_n + t_{nn'}$. Additionally, constraints (3.1i) ensure that the departure time at any node of a driver should be no earlier than their arrival time, except for their origin nodes o_k. Constraints (3.1j) to (3.1l) denote that item ij should leave store j and arrive at customer i, and that if item ij arrives at any other node, it must also leave this node (i.e., flow balance constraints). Constraints (3.1m) and (3.1n) state that item ij can traverse the arc (n,n') only if this arc is visited by a driver, and that arcs for drivers returning to origin nodes cannot be part of the path for items. Constraints (3.1o) specify the arrival time of item ij at node n. If item ij travels from node n to node n' (i.e., $y^{ij}_{nn'} = 1$), the arrival time of item ij at node n' must be no earlier than its arrival time at the previous node n, while accounting for the travel time from node n to node n'. These constraints can be linearized into $\lambda^{ij}_n + t_{nn'} - \lambda^{ij}_{n'} \leq M_2(1 - y^{ij}_{nn'})$, where M_2 must be large enough to ensure that $M_2 \geq \lambda^{ij}_n + t_{nn'}$. Constraints (3.1p) and (3.1q) indicate that if item ij passes the arc (n,n') visited by driver k, then item ij must be picked up by driver k to leave node n and dropped off by driver k at node n'. Since x, y, and v are binary, these constraints can be linearized into $v^{kij-}_n \geq x^k_{nn'} + y^{ij}_{nn'} - 1$ and $v^{kij+}_{n'} \geq x^k_{nn'} + y^{ij}_{nn'} - 1$, respectively. Constraints (3.1r) and (3.1s) state that the arrival time of item ij at node n should be no earlier than the arrival time of driver k at node n if this item is served by driver k to arrive at node n. Furthermore, the departure time of driver k should be no earlier than the arrival time of item ij at node n if this item is about to leave node n via driver k. These constraints can be linearized into $\tau^{k+}_n - \lambda^{ij}_n \leq M_3(1 - v^{kij+}_n)$ and $\lambda^{ij}_n - \tau^{k-}_n \leq M_4(1 - v^{kij-}_n)$, with $M_3 \geq \tau^{k+}_n$ and $M_4 \geq \lambda^{ij}_n$. Constraints (3.1t) ensure that the time windows are respected. Constraints (3.1u) represent capacity constraints for drivers at every node. Constraints (3.1v) and (3.1w) impose domain restrictions.

This model is capable of accommodating both single and multiple deliveries for multi-store orders placed by customers, while also allowing drivers to coordinate and transfer orders through transshipment, resulting in consolidated-order delivery with transshipment

(CODT). To evaluate the value of consolidation and transshipment in multi-store order delivery, we demonstrate ways to customize $\mathrm{M}(\mathscr{I}, \mathscr{J}, \mathscr{K})$ to model the separated-order delivery (SOD) and the consolidated-order delivery (COD) without transshipment.

The model for SOD requires that orders from each store are handled and delivered separately, which can be structured as a series of programs. For all $j' \in \mathscr{J}$,

$$\mathrm{M}_{\mathrm{SOD}}(\mathscr{I}_{j'}, \mathscr{J}_{j'}, \mathscr{K}_{j'}) = \min_{\substack{x,y,z,\\ \tau,v,\lambda}} \quad \rho_1 \max_{\substack{k\in\mathscr{K}_{j'}\\ i\in\mathscr{I}_{j'}}} \left\{\tau_i^{k+}\right\} + \rho_2 \sum_{k\in\mathscr{K}_{j'}} \sum_{n\in\mathscr{N}_{j'}} \sum_{\substack{n'\in\mathscr{N}_{j'},\\ n'\neq n}} t_{nn'} x_{nn'}^k \quad (3.2a)$$

$$\text{s.t.} \quad (3.1b) - (3.1w)$$

$$v_j^{kij-} = v_i^{kij+}, \quad \forall i \in \mathscr{I}_{j'}, j \in \mathscr{J}_{j'}, k \in \mathscr{K}_{j'}. \quad (3.2b)$$

The objective function (3.2a) specifies that each store optimizes their delivery operations separately to serve its respective customers. Here, $\mathscr{J}_{j'}$ denotes the set containing only store j' (i.e., $\mathscr{J}_{j'} = \{j'\}$). The set $\mathscr{I}_{j'}$ denotes customers who place orders from store j' (i.e., $\mathscr{I}_{j'} = \{i \in \mathscr{I} | e_{ij'} = 1\}$). There may be overlaps between different $\mathscr{I}_{j'}$ since customers can place orders from multiple stores. Similarly, $\mathscr{K}_{j'}$ denotes the drivers who serve orders from store j', and they jointly form a partition of $\mathscr{K}$. Furthermore, $\mathscr{N}_{j'} = \mathscr{I}_{j'} \cup \mathscr{J}_{j'} \cup \mathscr{O}_{j'}$, where $\mathscr{O}_{j'}$ is the set of origin nodes for drivers $k \in \mathscr{K}_{j'}$. To ensure at least one driver is available for each store, we assume that the number of drivers exceeds the number of stores. Note that the pre-assignment of drivers to stores (i.e., the set $\mathscr{K}_{j'}$) affects the $\mathrm{M}_{\mathrm{SOD}}$. To focus solely on the difference between separation and consolidation business models, we evaluate all possible partitions of the driver set $\mathscr{K}$ for the SOD. For each partition, we determine the worst-case performance across all stores by finding the maximum optimal objective value among them. The partition that yields the minimum worst-case performance (i.e., $\min\left\{\max_{j'\in\mathscr{J}} \mathrm{M}_{\mathrm{SOD}}(\mathscr{I}_{j'}, \mathscr{J}_{j'}, \mathscr{K}_{j'})\right\}$) is selected. This method works because our primary objective is to ensure on-time delivery for all customers, with the main target being to minimize the latest delivery time. By using this method, we can compare separation and consolidation models fairly without the results being influenced by the initial pre-assignment of drivers. Constraints (3.1b) - (3.1w) are applied, with $\mathscr{I}, \mathscr{J}, \mathscr{K}, \mathscr{N}$ replaced by $\mathscr{I}_{j'}, \mathscr{J}_{j'}, \mathscr{K}_{j'}, \mathscr{N}_{j'}$. Constraints (3.2b)

state that item ij is served by the same driver k encompassing both the pickup from store j and the delivery to customer i, implying no transshipment is allowed.

The model for COD without transshipment is:

$$\mathrm{M_{COD}}(\mathscr{I},\mathscr{J},\mathscr{K}) = \min_{\substack{x,y,z,\\ \tau,w,v,\lambda}} \quad \rho_1 \max_{\substack{k\in\mathscr{K}\\ i\in\mathscr{I}}} \left\{\tau_i^{k+}\right\} + \rho_2 \sum_{k\in\mathscr{K}} \sum_{n\in\mathscr{N}} \sum_{\substack{n'\in\mathscr{N},\\ n'\neq n}} t_{nn'} x_{nn'}^k \tag{3.3a}$$

$$\text{s.t.} \quad (3.1\text{b}) - (3.1\text{w})$$

$$v_j^{kij-} = v_i^{kij+}, \quad \forall i \in \mathscr{I}, j \in \mathscr{J}, k \in \mathscr{K} \tag{3.3b}$$

$$\sum_{k\in\mathscr{K}} z_i^k = 1, \quad \forall i \in \mathscr{I}. \tag{3.3c}$$

Constraints (3.3b) imply that item ij is assigned to the same driver for both pickup and delivery, and Constraints (3.3c) enforce a single delivery, ensuring that each customer is served exactly once for delivery, resulting in a consolidated-order delivery system without transshipment.

3.3.3 Dynamic Problem and Waiting Strategy

The MOP addresses the problem of serving customers who place orders in a specified time period, with transshipment facilitating spatial consolidation at both the store and customer levels. Since customer orders arrive dynamically over time in realistic settings, temporal consolidation can further improve operations by implementing an effective waiting strategy.

Assuming that customer arrivals follow a stochastic process, such as a Poisson process, the delivery system can be optimized by determining an appropriate *re-optimization interval*, defined as the time period between two consecutive optimizations. For each optimization, the system collects information on new customer orders that arrive within the interval, updates driver availability, allocates orders to available drivers, and plans the routing accordingly.

A longer waiting strategy can reduce the total travel time but may increase delivery delays. Conversely, an event-triggered strategy, which re-optimizes the system upon each order arrival without waiting, accelerates the delivery speed but may increase the total

travel time. The objective of the dynamic problem is to identify the waiting strategy that balances delivery time and waiting delays by determining the re-optimization interval, which also corresponds to the customer batching size during that interval. The implementation of the waiting strategies within the dynamic experimentation process are shown in Tables 3.10 and 3.11 in Appendix C.

3.4 Solution Procedure

Solving the model for MOP to optimality is computationally demanding due to the large number of constraints and variables arising from routing and time considerations. To address this, we adopt a learning-to-optimize approach that combines an offline learning phase with an online estimation and optimization phase. We next present the details of this framework in Section 3.4.1, introduce three learning methods in Section 3.4.2, and explain the optimization process in Section 3.4.3.

3.4.1 Learning-to-optimize Method

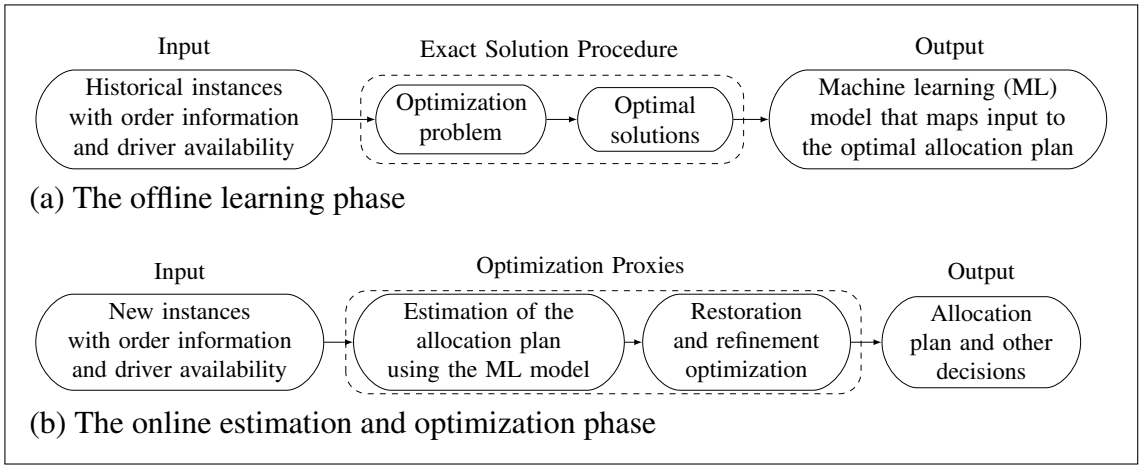


Figure 3.2 – Learning-to-optimize Method

The learning-to-optimize method is illustrated in Figure 3.2 and consists of two key steps. The first is the *offline learning phase*, as shown in Figure 3.2(a), where a mapping

is trained using machine learning models on historical instances. This step is conducted only once. Specifically, historical instances, which are solved to optimality using exact optimization techniques, are used to train a neural network model that learns the mapping from instance data to optimal allocation plans. The instance data includes order details, store and customer locations, initial driver positions, travel times between locations, and driver availability. The allocation plan determines the nodes visited by drivers, ensuring that each order is picked up from stores and delivered to customers. The second step is the *online estimation and optimization phase*, as shown in Figure 3.2(b). For each new instance, the estimated allocation plans are obtained from the learning model and then serve as a lower bound for the corresponding variables in the optimization problem. This optimization problem is a mixed-integer linear program, where restoration and refinement work together to ensure feasibility and improve the solution: restoration addresses potential infeasibility caused by the estimated allocation (i.e., the optimization input) underestimating time constraints and capacity limitations, while refinement further improves the estimated allocation decisions, refining the initial solutions to optimal ones through optimization. This process narrows the feasible solution space, provides flexibility to improve the estimated allocation plans, and completes the optimal driver routes and efficient paths for the MOP. This approach can offload a portion of the computational workload to the offline phase, facilitating the near real-time generation of high-quality solutions.

3.4.2 Learning Methods

Three neural network models are implemented to train the mapping from instance information to allocation plans. Let $\mathscr{N}$ denote the set of nodes, including driver origins, stores, and customers that drivers may visit, $\mathscr{K}$ denote the set of drivers, and $\mathscr{L}$ denote the feature space. We use bold symbols to represent vectors. We define the neural network model g, parameterized by θ, that maps the set of input features $f_n^k \in \mathbb{R}^{|\mathscr{L}|}$, for each node $n \in \mathscr{N}$ and driver $k \in \mathscr{K}$ to a binary allocation plan $\tilde{z} \in \{0,1\}^{|\mathscr{N}|\times|\mathscr{K}|}$, which represents

the decision of assigning driver k to visit node n. The function is presented by

$$g_\theta : \quad \mathbb{R}^{|\mathscr{N}|\times|\mathscr{K}|\times|\mathscr{L}|} \to \{0,1\}^{|\mathscr{N}|\times|\mathscr{K}|}. \tag{3.4}$$

All information related to orders, customers, stores, and drivers constitutes the feature set. These include store locations for order pickup, customer locations for delivery, driver initial locations, travel times between nodes, and nearest driver allocation to each node. Specifically, the features f_n^k for node n that can be served by driver k include: (1) node latitude; (2) node longitude; (3) distance between the driver and the node; (4) nearest driver allocation indicator, which is 1 if driver k is the nearest to node n; and (5) the ratio of the number of customers to the number of available drivers. Additionally, we generate additional features based on a *connected node* (CN) set for each node and driver. For store nodes, this set comprises the customers ordering from that store; for customer nodes, it includes the stores from which they have ordered; and for drivers, it consists of nodes that would be assigned to each driver based on the nearest allocation method. The additional features related to the CN set are: (6) the number of nodes in the CN union sets of both the node and the driver; (7) the Traveling Salesman Problem (TSP) cost for visiting nodes in this union CN set; and (8) the convex hull area enclosing the nodes in this union CN set.

1. Driver Assignment Neural Network (DANN): This method uses a neural network for binary prediction to determine whether driver k is assigned to node n. A training sample consists of a node-driver pair, where the input features are f_n^k and the label is z_n^{k*} for node n and driver k. Each driver is considered independently of all other drivers.

We implement a multi-layer neural network to learn the function g_θ, capturing the relationships between the input features and the binary decision outputs. The model predicts the probability of assigning node n to driver k:

$$Pr\left(\tilde{z}_n^k = 1 \mid f_n^k\right) = \sigma\left(\Theta(f_n^k)\right), \quad \forall n \in \mathscr{N}, k \in \mathscr{K},$$

where $\Theta(f_n^k) : \mathbb{R}^{|\mathscr{L}|} \to \mathbb{R}$ represents the neural network's output after multiple layers of computation on the input features $f_n^k \in \mathbb{R}^{|\mathscr{L}|}$. Since $\tilde{z}_n^k$ is binary, the final layer of the

neural network $\sigma(\cdot)$ applies the sigmoid activation function, $\sigma(x) = \frac{1}{1+e^{-x}}$, ensuring that the output is a probability between 0 and 1. If $Pr(\tilde{z}_n^k = 1 \mid f_n^k) > 0.5$, then $\tilde{z}_n^k = 1$, indicating that driver k is assigned to visit node n.

2. Driver Classification Neural Network (DCNN): This method treats driver assignment as a classification problem, where each driver is considered a distinct class. The neural network classifies each node into one of the available "driver classes", ensuring that at least one driver is assigned to visit each node. A training sample consists of a node with multiple drivers as potential choices, where the input features are $\left\{f_n^k\right\}_{k\in\mathscr{K}}$ and the labels are $\left\{z_n^{k*}\right\}_{k\in\mathscr{K}}$ for node n.

We again implement a multi-layer neural network to learn the function g_θ. For node n, the model outputs a probability distribution over classes k as follows:

$$Pr\left(\tilde{z}_n^k = 1 \mid f_n\right) = \left[\text{softmax}\left(\Theta\left(f_n\right)\right)\right]_k, \quad \forall n \in \mathscr{N}, k \in \mathscr{K},$$

where $\Theta(f_n) : \mathbb{R}^{|\mathscr{K}|\times|\mathscr{L}|} \to \mathbb{R}^{|\mathscr{K}|}$ represents the neural network's computation through multiple layers on the features $f_n \in \mathbb{R}^{|\mathscr{L}|\times|\mathscr{K}|}$ for node n and all drivers. The softmax function outputs a probability distribution over the $|\mathscr{K}|$ classes, ensuring that $\sum_{k=1}^{|\mathscr{K}|} Pr(\tilde{z}_n^k = 1 \mid f_n) = 1$. The driver k with the highest probability $Pr(\tilde{z}_n^k = 1 \mid f_n)$ is selected. Thus, $\tilde{z}_n^k = 1$ for the driver with the highest score, and $\tilde{z}_n^k = 0$ for all other drivers.

3. Graph-based Neural Network (GNN): This method employs a graph neural network to learn driver allocation plans by predicting edge labels that indicate whether a driver should visit a node within the graph structure, as illustrated in Figure 3.3, depicting an instance or area to be served.

Unlike traditional neural networks, the graph structure connects all nodes and includes edge features, global features, and labels. The *node features* $f_n \in \mathbb{R}^{|\mathscr{L}|}$ include: (1) driver type: 1 if the node represents a driver, 0 otherwise; (2) store type: 1 if the node represents a store, 0 otherwise; (3) customer type: 1 if the node represents a customer, 0 otherwise; (4) node latitude; (5) node longitude; (6) the number of nodes in the CN set; (7) the TSP cost of visiting the nodes in the CN set; and (8) the convex hull area enclosing the nodes in the CN set. The *edge features* $e_{nn'} \in \mathbb{R}^{|\mathscr{M}|}$ of all edges from nodes to drivers consist

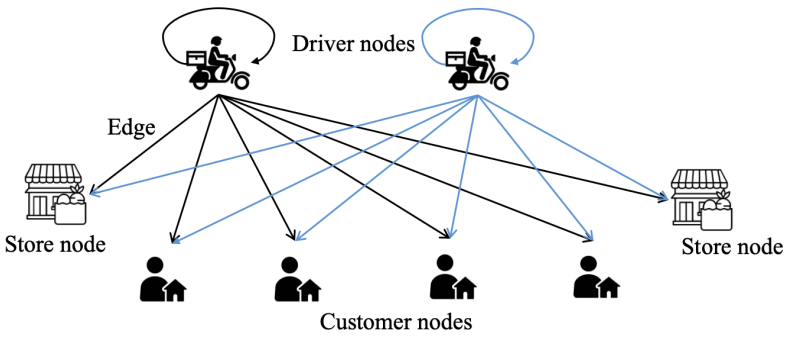


Figure 3.3 – Graph-based Neural Network structure for the case with 2 drivers, 2 stores, and 4 customers.

of (1) distance between the driver and the nodes; and (2) nearest driver allocation to the node. The *global feature* $g \in \mathbb{R}$ is the ratio of the number of customers to the number of available drivers. These features are similar to those used in DANN and DCNN, but are now represented in a graph structure. The *edge label* to be learned represents the optimal allocation of drivers to nodes. The GNN model leverages node features, edge features, and global features through multiple layers to capture complex relationships within the graph. A training sample consists of all node with all drivers as potential choices, where the input features are $\left(\{f_n\}_{n\in\mathcal{N}}, \{e_{nn'}\}_{n,n'\in\mathcal{N}}, g\right)$ and the labels are $\left\{z_n^{k*}\right\}_{k\in\mathcal{K},n\in\mathcal{N}}$ for the whole graph.

At each layer t, the embedding of node n, denoted $h_n^{(t)}$, is updated based on information from its neighboring nodes. The update is computed as:

$$h_n^{(t+1)} = \phi\left(h_n^{(t)}, \sum_{n'\in \text{Neight}(n)} \psi\left(h_n^{(t)}, h_{n'}^{(t)}, e_{nn'}\right), g\right),$$

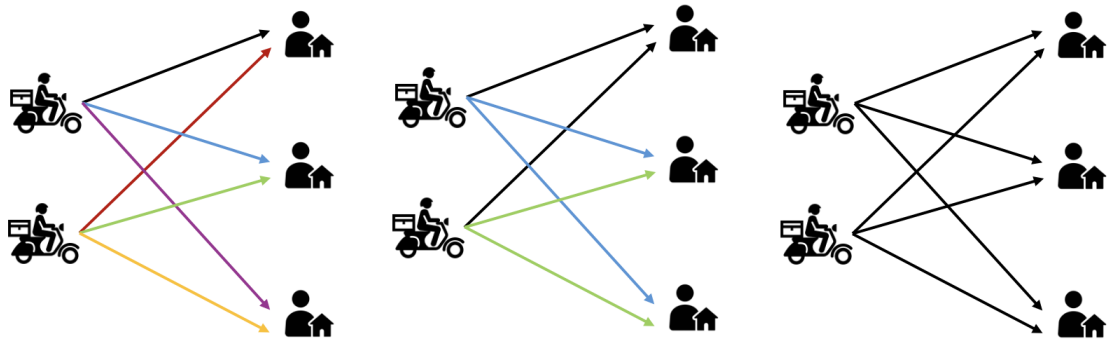
where $h_n^{(t)}$ is the embedding of node n at layer t with $h_n^{(0)} = f_n$ and $f_n \in \mathbb{R}^{|\mathscr{L}|}$, Neight$(n)$ denotes the set of neighbors of node n, $e_{nn'} \in \mathbb{R}^{|\mathscr{M}|}$ is the feature vector for the edge between node n and its neighbor n', g is the global feature, and ϕ and ψ are neural network layers.

After T layers, the GNN produces final embeddings $h_n^{(T)}$ for each node n. For all edges connecting node n, the final layer applies a softmax function to output a probability distribution over all possible drivers. The probability of assigning driver k to node n is

given by:

$$Pr\left(z_n^k = 1 \mid f_n, e_{nn'}, g\right) = \left[\text{softmax}\left(\varphi\left(h_n^{(T)}, h_{n'}^{(T)}, e_{nn'}, g\right)\right)\right]_k, \quad \forall n \in \mathcal{N}, k \in \mathcal{K},$$

where $\varphi : \mathbb{R}^{|\mathcal{N}|\times|\mathcal{L}|} \times \mathbb{R}^{|\mathcal{K}|\times|\mathcal{N}|\times|\mathcal{M}|} \times \mathbb{R} \to \mathbb{R}^{|\mathcal{K}|\times|\mathcal{N}|}$ is a neural network layer that combines the final node embeddings, edge features, and global features. The softmax function normalizes the edge outputs from each node n to all driver nodes, providing a probability distribution over drivers for node n, ensuring that $\sum_{k=1}^{|\mathcal{K}|} Pr\left(z_n^k = 1 \mid f_n, e_{nn'}, g\right) = 1$. The driver k with the highest probability $Pr\left(z_n^k = 1 \mid f_n, e_{nn'}, g\right)$ is selected to serve node n. This GNN framework effectively captures node interactions, edge features, and global features to predict the optimal driver-to-node allocations within the graph.



(a) DANN (b) DCNN (c) GNN

Note. The lines represent allocation decisions for a driver visiting a node. Lines of the same color indicate a single sample, while different colors correspond to different samples. In DANN, each driver-node pair is represented by a unique color; in DCNN, each node is assigned a color; and in GNN, the entire instance uses the same color.

Figure 3.4 – Learning models for allocation decisions with different ways of generating data samples

4. Summary: Overall, DANN, DCNN, and GNN use the same features but adopt different graphical structures for generating data samples to learn the allocation decisions. DANN treats each driver-node pair independently (see Figure 3.4 (a)). DCNN processes each node independently while incorporating driver information (see Figure 3.4 (b)). In contrast, GNN considers each instance as an interconnected network, allowing both driver and node information to be transferable during the learning process (see Figure 3.4 (c)).

In addition to these three learning methods, we also include Nearest Driver Allocation (NDA) as a benchmark, which assigns the closest driver to visit each store and customer node.

3.4.3 MILP-based Restoration and Refinement Problem

In the optimization phase, we formulate the restoration and refinement problem (RRP) as an MILP to restore feasibility, refine allocation decisions, and derive optimal solutions for other decisions. For decision refinement, the estimated allocation $\tilde{z}_n^k$ serves as a lower bound for the allocation decision z_n^k, allowing flexibility in assigning drivers to additional nodes and enabling the optimization of other decisions. However, given the lower bounds of allocation plans, the constraints in equations (3.1t) and (3.1u) may be violated. It is essential to ensure that all items ordered by customer i from store j are delivered within the specified time window while respecting capacity limitations. In other words, the machine learning output may underestimate the time constraints and capacity limitations, which could render Problem 3.1 infeasible given the estimated $\tilde{z}_n^k$. To address this, we apply soft time windows and capacity constraints while also minimizing the slackness to restore feasibility. The model for the RRP is formulated as follows:

$\mathrm{M}_{\mathrm{RRP}}(\mathscr{I}, \mathscr{J}, \mathscr{K}) =$

$$\min_{\substack{x,y,z,\\ \tau,w,v,\lambda,s}} \quad \rho_1 \max_{\substack{k\in\mathscr{K}\\ i\in\mathscr{I}}} \left\{\tau_i^{k+}\right\} + \rho_2 \sum_{\substack{k\in\mathscr{K}\\ n\in\mathscr{N}}} \sum_{\substack{n'\in\mathscr{N},\\ n'\neq n}} t_{nn'} x_{nn'}^k + \rho_3 \sum_{\substack{n\in\mathscr{N}\\ k\in\mathscr{K}}} \sum_{\substack{i\in\mathscr{I}\\ j\in\mathscr{J}}} (s_n^{1ij} + s_n^{2ij} + s_n^{3k}) \tag{3.5a}$$

$$\text{s.t.} \quad (3.1\text{b}) - (3.1\text{s}), (3.1\text{v}) - (3.1\text{w})$$

$$z_n^k \geq \tilde{z}_n^k, \quad \forall k \in \mathscr{K}, n \in \mathscr{N} \tag{3.5b}$$

$$\alpha_i - s_i^{1ij} \leq \lambda_i^{ij} \leq \beta_n + s_i^{2ij}, \quad \forall i \in \mathscr{I}, j \in \mathscr{J} \tag{3.5c}$$

$$\sum_{i\in\mathscr{I}} \sum_{j\in\mathscr{J}} p_{ij} v_n^{kij-} \leq q^k + s_n^{3k}, \quad \forall n \in \mathscr{N}, k \in \mathscr{K} \tag{3.5d}$$

$$s_i^{1ij}, s_i^{2ij}, s_n^{3k} \geq 0, \forall i \in \mathscr{I}, j \in \mathscr{J}, n \in \mathscr{N}, k \in \mathscr{K}. \tag{3.5e}$$

The objective function (3.5a) states that, in addition to minimizing the weighted average of the latest delivery time and the total travel time, we also minimize the penalty for violating time window and capacity constraints, which cannot be avoided due to the input allocation. Note that ρ_3 represents the penalty weight, and it should be set to a large value relative to ρ_1 and ρ_2 to avoid violations as much as possible. Constraints (3.5b) ensure that the estimated $\tilde{z}$ serves as a lower bound, while Constraints (3.5c) and (3.5d) represent soft time windows and capacity constraints with slack variables s_i^{1ij}, s_i^{2ij}, and s_n^{3k} to maintain feasibility. Finally, Constraints (3.5d) impose domain restrictions on the slack variables.

3.5 Numerical Study

In this section, we first introduce a real-world dataset, performance metrics, and implementation details. We then compare the performance of three systems to evaluate the benefits of offering consolidation and transshipment for the multi-store order delivery. Due to the computational challenges involved in solving the problem, we employ learning-to-optimize techniques to accelerate the solution process through various learning methods and compare the effectiveness of different learning-based optimization proxies. Finally, we conduct dynamic experimentations in a practical setting to serve customers in areas with varying customer locations and arrival rates, aiming to identify the most efficient delivery system and waiting strategy.

3.5.1 Dataset and Implementation Details

We use a customer location dataset from four regions in the U.S. (Los Angeles, Seattle, Tacoma, and Orange) provided by Amazon (Merchan et al. 2021), which contains the perturbed locations of customers. We obtain the expected travel times using the Google API. Instances with varying scales are created from the dataset, with customer numbers ranging from 2 to 20 and driver numbers ranging from two to five. Customer locations are sampled

uniformly from the dataset, with each location having an equal probability of selection, referred to as *uniformly sampled customers*. We also consider *clustered customers*, where a central point is pre-selected, and locations closer to this center have a higher probability of being chosen as customer locations. Drivers' initial locations and store locations are randomly generated and remain fixed within each region for the sensitivity analysis. To assess the benefits of consolidating orders from multiple stores, we assume that customers place orders from at least two stores and up to four stores, with whether each customer orders from a particular store, denoted as $e_{ij} \in 0, 1$, being randomly generated. For customer i, the start time windows α_i are uniformly generated within a range of 0 to 20, and the end time windows β_i are within a range of 20 to 70. Driver capacity q_k is set to 100, with item sizes p_{ij} randomly generated as integers between 0 and 10. Our primary objective is to ensure on-time delivery for all customers by minimizing the latest delivery time, so the weight ρ_1 is set to 1. A smaller weight $\rho_2 = 0.01$ is used to discourage unnecessary long travel times. To minimize constraint violations, a penalty weight $\rho_3 = 100$ is applied.

To compare the performance of three delivery systems and four learning methods, we define the following metrics. (1) Delivery time is the duration between order assignment and delivery, with the latest delivery time being the time to serve the last order arriving within the re-optimization interval. A lower value indicates faster overall delivery. (2) Total travel time is the total time drivers spend traveling, including the time from their starting location to pick up orders and deliver them. The travel time for serving one more customer is calculated as the ratio of total travel time to the number of customers. A lower value indicates reduced overall costs. (3) Wait time is the duration between order placement and assignment. A lower value means quicker driver pickup. (4) Order completion time is the duration from order placement to delivery, including both wait time and delivery time. (5) Runtime is the time required to find final optimal solutions through optimization, whether or not learning is used. A lower runtime indicates a more efficient solution method. (6) Accuracy in learning is the percentage of correct driver-to-customer allocations. Higher accuracy indicates better learning performance. (7) Allocation percentage (pct.) is the percentage of estimated allocation decisions estimated as 1, indicating

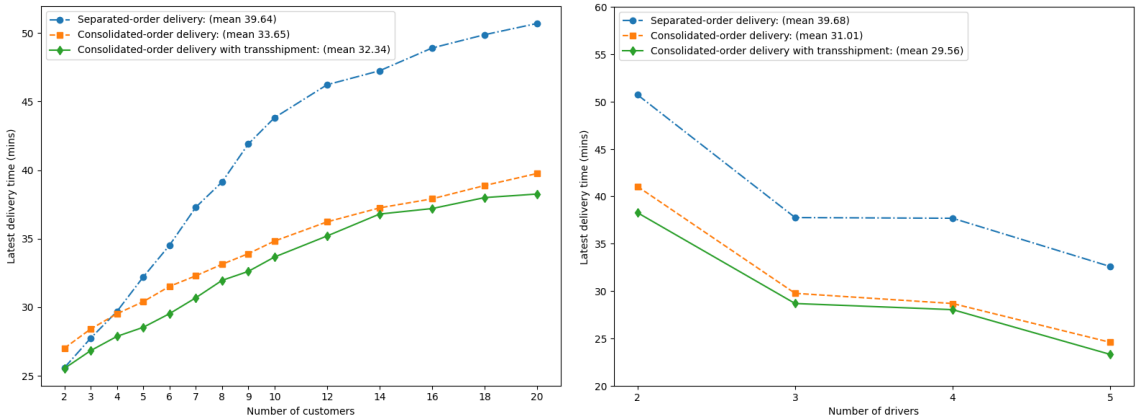
that the driver will visit that location. A lower value offers more flexibility in optimizing the solution. (8) Allocation standard deviation (std.) is the standard deviation of allocation decisions, which indicates the balance of driver workloads. A lower value signifies a more balanced distribution of work. (9) Gap is the difference between the estimated and exact optimal values for key metrics such as objective value, latest delivery time, and total travel time. A lower gap means the learning-based solution is closer to the true optimal solution.

We implement our algorithms using Python 3.10 and Gurobi 10.0.2 on a local computer equipped with a 2 GHz Quad-Core Intel Core i5 processor and 16 GB of RAM, supplemented by resources from Compute Canada's Graham cluster, which includes Multi-Core Intel Xeon processors (20 to 36 cores per node) and standard compute nodes with 64 GB of RAM. The optimization time limit for the exact solution procedure is set to 3600 seconds, while the time limit for learning-to-optimize during the comparison of learning methods is set to 600 seconds. For implementations in dynamic environments, this limit is further reduced to 300 seconds when applying learning-to-optimize with GNN, ensuring efficient real-time decision-making.

3.5.2 Comparison of delivery systems

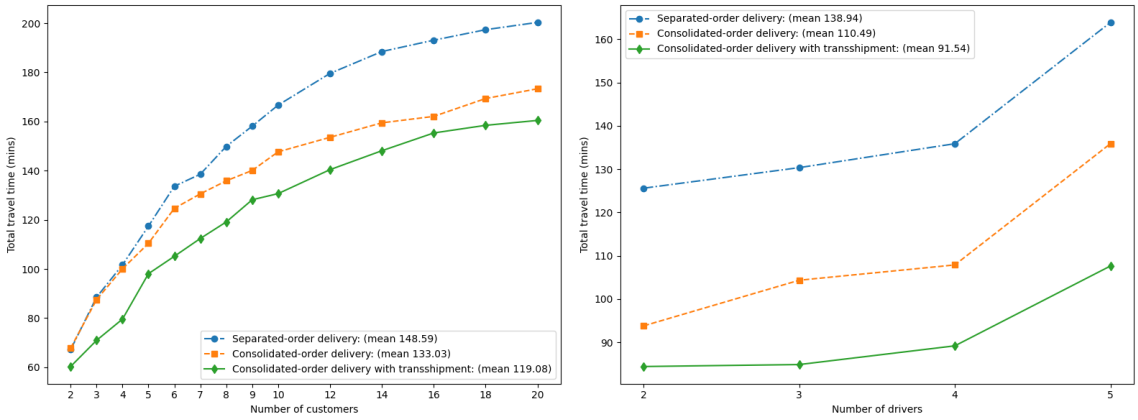
To compare the three systems, including separated order delivery (SOD), consolidated order delivery (COD), and consolidated order delivery with transshipment (CODT), we solve the models M_{SOD}, M_{COD}, and M using Gurobi within the specified time limit across various instances.

Figures 3.5(a) and (b) plot the latest delivery time, defined as the maximum duration between order assignment and delivery across customers. The SOD yields faster delivery times than the COD when there are few customers to serve, as the benefits of consolidating requests are minimal. However, as more customers join the system and place orders, the COD begins to dominate the SOD, and this dominance increases with the number of customers. Additionally, the COD consistently yields a shorter total travel time, which



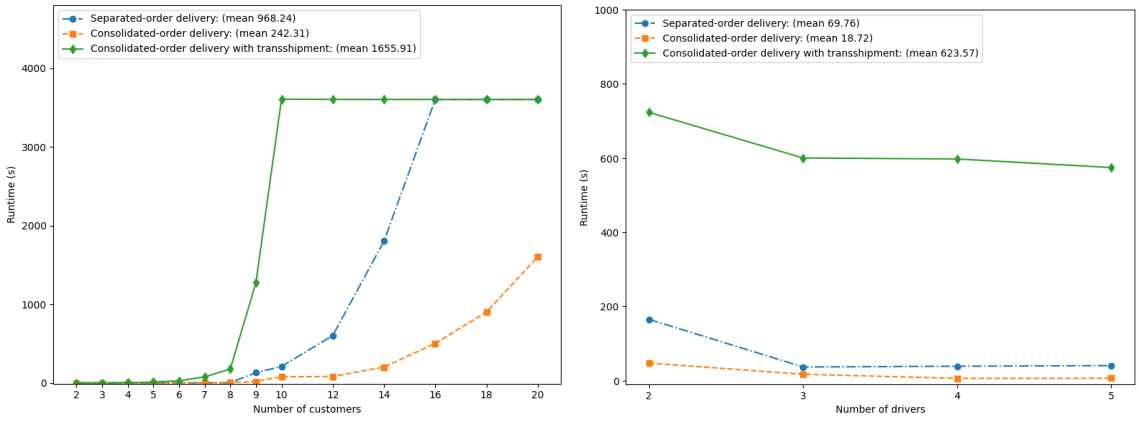
(a) Latest delivery time (Customer)

(b) Latest delivery time (Driver)



(c) Total travel time (Customer)

(d) Total travel time (Driver)



(e) Runtime (Customer)

(f) Runtime (Driver)

Figure 3.5 – Comparison of delivery systems with various number of customers and drivers

represents the total time drivers take to complete deliveries for all requests (see Figures 3.5(c) and 3.5(d)). It also converges to optimality faster (see Figures 3.5(e) and 3.5(f)).

In contrast, the CODT is always the most efficient among all three systems in terms of delivery time for serving each customer and the total travel time for fulfilling all requests, regardless of the instance scale (see Figures 3.5(a)−(d)). However, due to the complexity of coordination and the flexibility of transshipment at any location, solving the CODT using exact methods requires substantially more computational time (see Figures 3.5(e) and (f)). Instances with more than 12 customers are unlikely to converge to optimality within the 3600-second time limit, and a lower bound may not be found to produce a feasible solution within 600 seconds. To address this challenge, we implement learning-to-optimize techniques to explore whether high-quality solutions can be achieved more efficiently, mitigating this computational drawback.

Insight 7 *Consolidated-order delivery with transshipment outperforms other delivery systems in terms of both the delivery time for serving each customer and the total travel time for fulfilling all requests. However, the complexity from consolidation and transshipment increases the computational time required for exact solutions.*

Insight 8 *Without transshipment, consolidated-order delivery outperforms separated-order delivery in delivery time, total travel time, and exact solution time, except when the number of customers is very small.*

3.5.3 Comparison of learning algorithms

We compare four methods, including NDA, DANN, DCNN, and GNN, that map instance information to allocation decisions. We implement the training process under three different instance scales: small-scale instances, where the number of customers ($|\mathscr{I}|$) ranges from two to seven and the number of drivers ($|\mathscr{K}|$) ranges from two to five; medium-scale instances, where $|\mathscr{I}|$ ranges from eight to ten and $|\mathscr{K}|$ ranges from two to five; and large-scale instances that fail to converge to optimality, where $|\mathscr{I}|$ ranges from 12 to 20 and

$|\mathscr{K}|$ ranges from two to five. First, we train using both small and medium-scale instances and test on small, medium, and large-scale instances. In the second set of experiments, we train solely on small-scale instances but test on all three scales. This approach allows us to assess the model's ability to generalize and predict decisions for larger-scale instances that may not have been included in the training set.

Table 3.1 – Best Learning Method for Uniformly Sampled Customers in the Learning Process

Training Scale	**Testing Type**	**Testing Scale**			**Best Method**	**Accuracy**	**Allocation Pct. (%)**	**Allocation Std.**
		$\|\mathscr{I}\|$	$\|\mathscr{K}\|$	NoI				
$\|\mathscr{I}\|$: [2, 10]; $\|\mathscr{K}\|$: [2, 5]; NoI: 3600	T1	[2, 7]	[2, 3]	1200	GNN	0.83	41.67	0.31
	T1	[2, 7]	[4, 5]	1200	GNN	0.92	22.50	0.43
	T1	[8, 10]	[2, 3]	600	DCNN	0.84	41.67	0.30
	T1	[8, 10]	[4, 5]	600	DCNN	0.90	22.50	0.04
	T5	[12, 20]	[2, 5]	400	GNN	0.74	32.08	0.55
	T6	[2, 20]	[2, 5]	4000	GNN	0.88	32.08	0.15
$\|\mathscr{I}\|$: [2, 7]; $\|\mathscr{K}\|$: [2, 3]; NoI: 1200	T1	[2, 7]	[2, 3]	1200	GNN	0.84	41.67	0.30
	T2	[2, 7]	[4, 5]	1200	GNN	0.92	22.50	0.70
	T3	[8, 10]	[2, 3]	600	GNN	0.86	41.67	0.35
	T4	[8, 10]	[4, 5]	600	GNN	0.90	22.50	0.09
	T5	[12, 20]	[2, 5]	400	GNN	0.73	32.08	0.40
	T6	[2, 20]	[2, 5]	4000	GNN	0.87	32.08	0.15

Note. $|\mathscr{I}|$*: range of customer numbers;* $|\mathscr{K}|$*: range of driver numbers; NoI: number of instances. T1 corresponds to testing and training with the same scale, while T2−T5 all involve larger scales. Specifically, T2 has a larger customer scale, T3 a larger driver scale, and T4 both scales increased. T5 represents overall large-scale instances, and T6 includes all testing instances.*

Table 3.2 – Best Learning Method for Uniformly Sampled Customers in the Optimization Process

Training Scale	**Testing Type**	**Testing Scale**			**Best Method**	**Objective Gap(%)**	**Delivery Time Gap(%)**	**Travel Time Gap(%)**	**Run-time (s)**
		$\|\mathscr{I}\|$	$\|\mathscr{K}\|$	**NoI**					
$\|\mathscr{I}\|$: [2, 10]; $\|\mathscr{K}\|$: [2, 5]; NoI: 3600	T1	[2, 7]	[2, 3]	1200	DCNN	2.10	2.10	1.19	5
	T1	[2, 7]	[4, 5]	1200	DCNN	0.94	0.94	0.15	5
	T1	[8, 10]	[2, 3]	600	DCNN	1.66	1.65	4.92	14
	T1	[8, 10]	[4, 5]	600	DCNN	1.66	1.68	10.42	30
	T5	[12, 20]	[2, 5]	400	GNN	1.19	1.23	16.76	457
	T6	[2, 20]	[2, 5]	4000	DCNN	1.56	1.56	2.75	40
$\|\mathscr{I}\|$: [2, 7]; $\|\mathscr{K}\|$: [2, 3]; NoI: 1200	T1	[2, 7]	[2, 3]	1200	DCNN	2.07	2.07	1.67	6
	T2	[2, 7]	[4, 5]	1200	NDA	1.25	1.25	0.37	6
	T3	[8, 10]	[2, 3]	600	GNN	3.19	3.17	5.30	21
	T4	[8, 10]	[4, 5]	600	GNN	1.89	1.92	12.13	33
	T5	[12, 20]	[2, 5]	400	GNN	1.73	1.77	17.56	406
	T6	[2, 20]	[2, 5]	4000	GNN	2.78	2.78	3.80	38

Table 3.3 – Best Learning Method for Clustered Customers in the Learning Process

Training Scale	Testing Type	Testing Scale $\lvert\mathscr{I}\rvert$	Testing Scale $\lvert\mathscr{K}\rvert$	Testing Scale NoI	Best Method	Accuracy	Allocation Pct. (%)	Allocation Std.
$\lvert\mathscr{I}\rvert$: [2, 10]; $\lvert\mathscr{K}\rvert$: [2, 5]; NoI: 3600	T1	[2, 7]	[2, 3]	1200	GNN	0.80	41.67	0.57
	T1	[2, 7]	[4, 5]	1200	GNN	0.90	22.50	0.09
	T1	[8, 10]	[2, 3]	600	GNN	0.73	41.67	0.29
	T1	[8, 10]	[4, 5]	600	GNN	0.88	18.71	0.45
	T5	[12, 20]	[2, 5]	400	GNN	0.69	32.08	0.76
	T6	[2, 20]	[2, 5]	4000	GNN	0.84	32.08	0.51
$\lvert\mathscr{I}\rvert$: [2, 7]; $\lvert\mathscr{K}\rvert$: [2, 3]; NoI: 1200	T1	[2, 7]	[2, 3]	1200	GNN	0.81	41.67	0.19
	T2	[2, 7]	[4, 5]	1200	GNN	0.89	22.50	0.35
	T3	[8, 10]	[2, 3]	600	GNN	0.70	41.67	0.76
	T4	[8, 10]	[4, 5]	600	GNN	0.86	22.50	0.30
	T5	[12, 20]	[2, 5]	400	GNN	0.69	32.08	0.17
	T6	[2, 20]	[2, 5]	4000	GNN	0.83	32.08	0.94

Table 3.4 – Best Learning Method for Clustered Customers in the Optimization Process

Training Scale	Testing Type	Testing Scale $\lvert\mathscr{I}\rvert$	Testing Scale $\lvert\mathscr{K}\rvert$	Testing Scale NoI	Best Method	Objective Gap(%)	Delivery Time Gap(%)	Travel Time Gap(%)	Run-time (s)
$\lvert\mathscr{I}\rvert$: [2, 10]; $\lvert\mathscr{K}\rvert$: [2, 5]; NoI: 3600	T1	[2, 7]	[2, 3]	1200	DANN	1.88	1.86	2.08	6
	T1	[2, 7]	[4, 5]	1200	DANN	0.23	0.23	4.34	5
	T1	[8, 10]	[2, 3]	600	GNN	1.16	1.12	11.79	12
	T1	[8, 10]	[4, 5]	600	GNN	1.01	0.97	21.48	24
	T5	[12, 20]	[2, 5]	400	GNN	1.74	1.69	16.76	450
	T6	[2, 20]	[2, 5]	4000	DANN	1.93	1.90	1.47	111
$\lvert\mathscr{I}\rvert$: [2, 7]; $\lvert\mathscr{K}\rvert$: [2, 3]; NoI: 1200	T1	[2, 7]	[2, 3]	1200	GNN	2.35	2.33	4.30	7
	T2	[2, 7]	[4, 5]	1200	GNN	1.79	1.73	24.09	8
	T3	[8, 10]	[2, 3]	600	GNN	2.90	2.89	3.57	15
	T4	[8, 10]	[4, 5]	600	GNN	1.20	1.18	7.55	31
	T5	[12, 20]	[2, 5]	400	GNN	2.36	2.30	16.76	457
	T6	[2, 20]	[2, 5]	4000	GNN	2.07	2.07	9.01	38

To simplify and clarify the results, we present the best-performing learning method and its learning performance for uniformly sampled customers in Table 3.1, with its re-optimization performance in Table 3.2. For clustered customers, the best learning method and its performance are shown in Table 3.3, and the corresponding optimization performance by solving M_{RRP} is provided in Table 3.4. To differentiate between the training and testing scales, we consider six testing types, each comprising distinct testing instances. T1 corresponds to the scenario where the testing scale aligns with the training scale. T2 pertains to cases with a larger customer scale, while T3 refers to instances with a larger driver scale. T4 encompasses situations where both customer and driver scales are increased. T5 includes instances characterized by larger overall scales, and T6 represents all testing in-

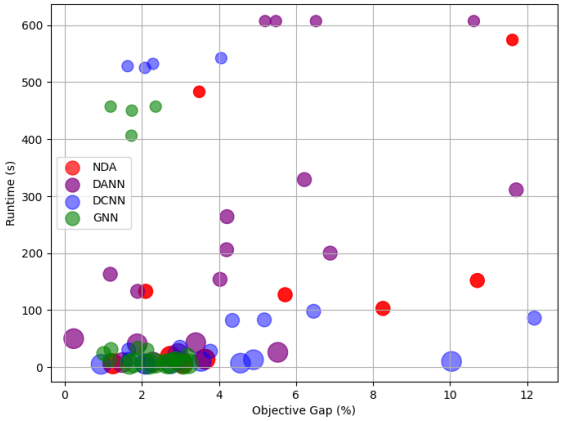
stances. Both the training scale and testing scale clearly show the range of customer and driver numbers. For the learning process, we present the best method with the highest accuracy, along with its percentage and standard deviation of allocation plans. For the optimization process, we display the best method with the lowest objective gap, including its delivery time gap, total travel time gap, and optimization runtime. More details on the metrics can be found in Section 3.5.1. Detailed performance results for each method under both uniformly sampled customer and clustered customer cases are provided in Appendix B.

According to Tables 3.1 and 3.3, we find that GNN demonstrates the overall best performance in the learning process with the highest accuracy under most scenarios, regardless of whether the testing scale is included in the training set. In instances where both small and medium-scale datasets are included in the training set, DCNN can achieve performance comparable to GNN for those scales. In other words, GNN excels at generalizing and predicting decisions for larger-scale instances that may not have been part of the training data. Both GNN and DCNN perform well if the scale of future instances matches that of historical instances.

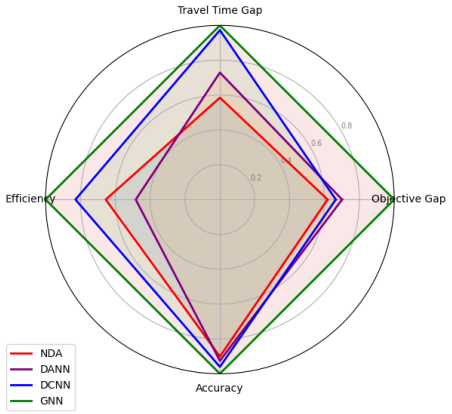
This difference in performance arises because, in DANN, the assignment decision for each driver-customer pair is considered independently, which means there is no guarantee that each customer will be served by exactly one driver. This is evident in the unstable allocation percentages observed with DANN (see Tables 3.6 and 3.8 in Appendix B). In contrast, NDA, DCNN, and GNN ensure that at least one driver is allocated to serve each customer. However, in NDA, the allocation of drivers to customers is based solely on the distances between them. DCNN incorporates more features, leading to higher accuracy compared to NDA. Nevertheless, DCNN still treats customers independently, as its goal is to find the best driver for each individual customer. GNN, on the other hand, connects all customers and drivers through edges, ensuring that every customer is served and allowing for control over the number of customers each driver serves by summing edge labels.

Tables 3.2 and 3.4 display the optimal outputs when the estimated allocation decisions are used as input for optimizing the restoration and refinement problem. First, we

observe that higher accuracy does not always correspond to a lower optimality gap. This is because there may be multiple good solutions that yield low delivery times during the optimization process. Second, no single method consistently outperforms the others in all cases. Overall, GNN has a smaller optimality gap, along with lower delivery time and travel time, particularly in the clustered customer cases when the testing scale exceeds the training scale. When the training scale includes the testing scale, DCNN tends to have a smaller gap in scenarios with uniformly sampled customers, along with a lower allocation standard deviation. In contrast, DANN exhibits a smaller gap in cases with clustered customers, accompanied by a lower allocation percentage. A lower allocation standard deviation indicates a more balanced workload among drivers, while a lower allocation percentage provides greater flexibility in finding optimal allocation decisions during the optimization process.



(a) Trade-off between optimality gap and runtime



(b) Relative performances of different metrics

Figure 3.6 – Comparison of performances in optimization of four learning methods

A summary of the re-optimization results across both scenarios, based on different training processes, is provided in Figure 3.6. Figure 3.6(a) displays the trade-off between expected runtime and optimality gap for the four methods across all scenarios, with circle size indicating sample size. In terms of runtime for optimizing the restoration and refinement problem, GNN proves more efficient than the other methods, while DANN generally results in the highest runtime across most scenarios. Overall, NDA and DANN are more

time-intensive in the optimization process compared to GNN and DCNN. A lower runtime for optimization reflects that the input-estimated allocation decisions provide a stronger lower bound. By using learning-to-optimize techniques, we achieve optimal solutions in 30 seconds for medium-scale instances with 8 to 10 customers, and in 450 seconds for large-scale instances with 12 to 20 customers. In contrast, pure optimization methods may take up to 3600 seconds for medium-scale instances and may fail to converge in 3600 seconds for large-scale cases, as shown in Figure 3.5. Figure 3.6(b) presents relative performance metrics, including accuracy, objective gap, travel time gap, and runtime efficiency, relative to the best approach among all methods. A value of 1 indicates the best performance: highest accuracy, lowest objective gap, lowest travel time gap, and highest efficiency. We conclude that GNN delivers the most comprehensive performance across metrics and scenarios, consistently achieving the lowest optimization runtime and smallest optimality gap by providing an effective lower bound.

Insight 9 *In a learning-to-optimize framework, we can use learning to obtain a lower bound for the allocation decision, which helps reduce the search space for optimization, thereby accelerating the solution procedure.*

Insight 10 *Overall, GNN performs the best with the highest accuracy, the smallest optimality gap, efficient runtime, and superior scalability for larger instances not included in the training set.*

Insight 11 *In the learning-to-optimize framework, higher accuracy during the learning process does not necessarily lead to a smaller optimality gap in the optimization process. A lower allocation percentage allows for greater flexibility and may lead to a smaller gap, but at the cost of a longer optimization runtime. A lower allocation standard deviation, indicating a balanced workload, may also imply a shorter delivery time and lead to a smaller gap.*

The four learning methods each have their own advantages and disadvantages. (1) NDA is simple to implement and performs well without historical data, effectively gen-

eralizing with an increasing number of drivers. However, it struggles with complex customer-driver interactions and performs poorly with clustered customer distributions, as well as struggling to generalize with an increasing number of customers. (2) DANN achieves high accuracy with sufficient training data and works well for instances of similar scale, especially in cases with clustered customer distributions. However, it is computationally inefficient, struggles to generalize with larger customer sets, and yields unstable allocation percentages that can lead to longer travel times. (3) DCNN also achieves high accuracy with sufficient data and works well for instances of similar scale, performing effectively with uniform customer distributions. However, it requires extensive training data and careful tuning, and it struggles to generalize with larger driver sets. (4) GNN excels at modeling complex interactions between customers and drivers using structured graphs, achieving high accuracy across different scenarios while being computationally efficient. It also generalizes well to instances with more customers and drivers. However, it faces challenges in collecting sufficient historical samples, as each instance corresponds to a single graph, and its highest accuracy may not always correlate with the lowest optimality gap. The detailed evidence can be found in Tables 3.6 to 3.9 in Appendix B.

3.5.4 Experimentation in a Dynamic Environment

In the dynamic delivery problem, the waiting strategy, which groups customer orders based on arrival time, can reduce total driver travel times for serving all customers but may increase order completion times (the duration from order placement to delivery). To evaluate the efficiency of different waiting strategies and delivery systems, we simulate the dynamic experimentation process under various waiting strategies with limited driver availability.

As shown in Figure 3.7, customers continuously arrive, and delivery can be optimized either immediately upon order arrival to ensure the fastest delivery or after a fixed interval, optimizing delivery for all batched customers. The simulation models customer arrivals as a Poisson process, with rates varying from 4 to 10 customers every 10 minutes. Sim-

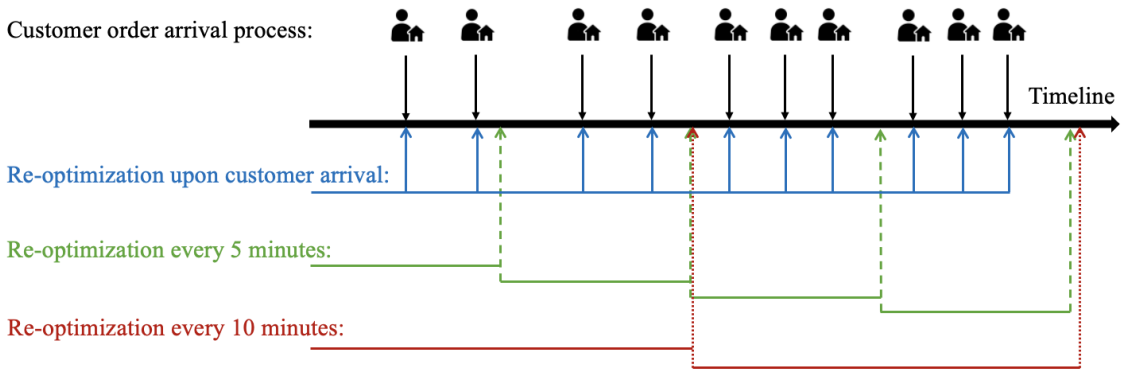
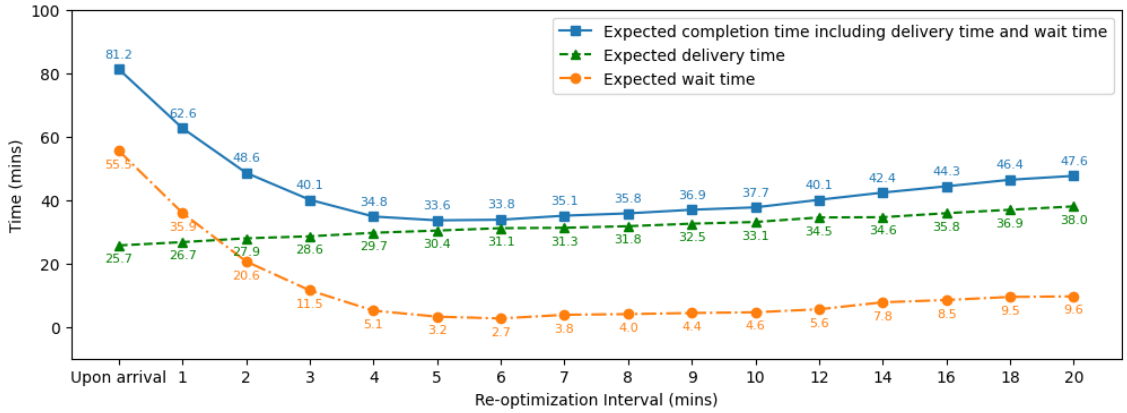


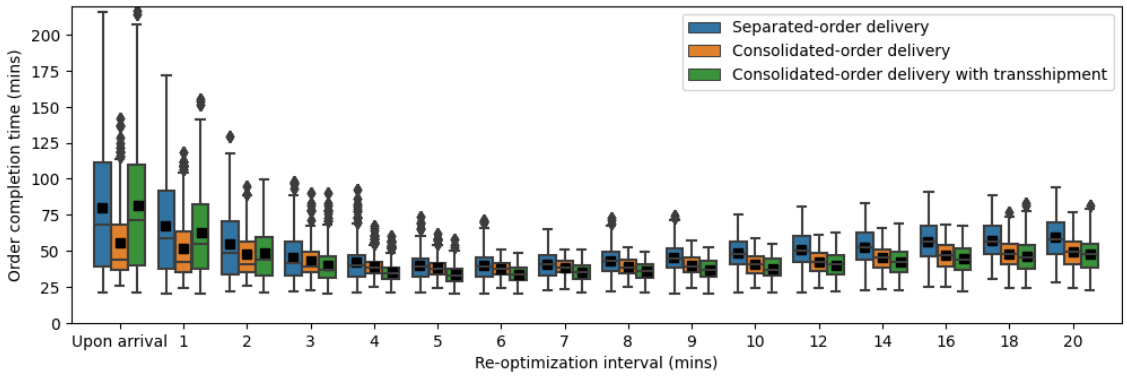
Figure 3.7 – Customer arrival process and dynamic experimentation process

ulations are conducted over durations of one and two hours, with the number of drivers ranging from 10 to 20. Each setup is repeated 40 times to ensure robust results, simulating real-world conditions while keeping other parameters consistent with the static model. For a given re-optimization interval, the system optimizes driver assignments and routes to serve orders arriving during the interval and updates availability based on task completion for the next interval. This process repeats with new customer arrivals at each interval. Performance metrics such as completion time, wait time, delivery time, and travel time are recorded. The detailed experimentation process is presented in Table 3.10 in Appendix C1. Given limited drivers, two re-optimization strategies are considered when no drivers are available within the rolling horizon. The first strategy always fixes re-optimization time points and assigns customers to the earliest available drivers if no driver is free, with results shown below. The second strategy delays optimization until enough drivers are available, with details and a comparison of these strategies provided in Appendix C.

Figure 3.8(a) illustrates the trade-off between delivery time and wait time across re-optimization intervals ranging from event-triggered to 20 minutes, resulting in a function that initially decreases and then increases for order completion time, which includes both delivery and wait times. Under the event-triggered strategy, delivery times are shorter, but wait times are significantly longer. This occurs because drivers are immediately assigned to serve individual customers, resulting in fast delivery but inefficient use of drivers. As a result, drivers are occupied with deliveries before they can serve the next customer,



(a) Trade-off between wait time and delivery time for consolidated-order delivery with transshipment



(b) Completion time including wait time and delivery time

Figure 3.8 – Completion time under varying re-optimization intervals

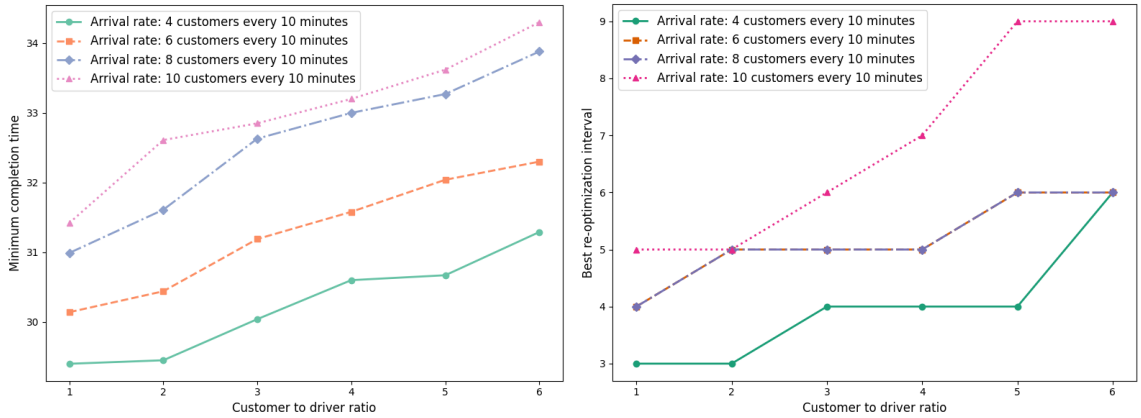
and customers must wait longer for the earliest available driver, reducing overall driver availability and increasing wait times. In contrast, the 20-minute re-optimization interval results in longer delivery times but more moderate wait times. This is because more customers are batched together and served by the same driver, improving the utilization of available drivers. However, since each driver must serve more customers once dispatched, delivery times are longer. Additionally, as customers must wait for batching, the wait time before re-optimization (i.e., before drivers are assigned) increases as the re-optimization interval lengthens. Overall, the five-minute re-optimization interval, which minimizes completion time, strikes a favorable balance between wait time and delivery time. This interval ensures that customers spend minimal time waiting for an available driver while also allowing for prompt delivery once the orders are assigned. Figure 3.8(b) displays the

completion times for three systems under varying re-optimization intervals. For all three systems, order completion time initially decreases before increasing as the re-optimization interval extends. Notably, all systems achieve the lowest order completion time with the 5-minute re-optimization interval among all waiting strategies. COD outperforms the other systems in extremely short re-optimization intervals, such as the event-triggered and one-minute or two-minute intervals, while CODT excels across all other re-optimization intervals. The detailed distribution of delivery time, wait time, travel time for serving one additional customer, and customer scale for the three systems under different re-optimization intervals is presented in Figure 3.10 in Appendix C1.

Insight 12 *Consolidated-order delivery is the most efficient system regarding order completion time and average travel time for serving each customer under the event-triggered strategy or a short re-optimization interval. This suggests that when there are few customers to serve, it is advantageous for drivers to pick up all requests from different stores without transshipment and deliver them as a combined order, as the savings in wait time outweigh the costs of delivery time.*

Insight 13 *Consolidated-order delivery with transshipment with a five-minute re-optimization interval is the most effective strategy, yielding the lowest order completion time among all delivery systems across different waiting strategies. This approach entails connecting orders that arrive within each five-minute interval, assigning the earliest available drivers to visit different stores, meeting at a transshipment node to exchange items, and making a single delivery to fulfill all requests for each customer.*

The best waiting strategy may vary depending on whether it is a busy or leisurely time for customer arrivals and changes in driver availability. Figure 3.9 illustrates the best re-optimization interval that results in the minimum order completion time across various customer arrival rates and different ratios of the customer number to available driver number. As the customer arrival rate increases, the best re-optimization interval also becomes larger, leading to a longer completion time. This indicates that when customers arrive



(a) Minimum completion time

(b) Best re-optimization interval

Figure 3.9 – Best re-optimization interval for varying ratios of customer number to driver number

more frequently, it is beneficial to wait longer to optimize and fulfill orders. Despite the increased completion time, this strategy still remains more efficient than the others. This is because the reduction in wait time for available drivers outweighs the increase in delivery times. This principle holds true regardless of the customer arrival rate, but it becomes more pronounced when customers arrive more frequently. Moreover, as the ratio of customers to drivers increases, indicating a more limited number of drivers available to serve the same number of customers at a consistent arrival rate, it is advisable to wait longer. This is because once a driver is engaged in the early period, it takes more time for them to become available to serve subsequent customers arriving later.

3.6 Conclusion

The multi-store consolidated-order delivery service allows customers to purchase products from multiple stores in a single transaction without additional delivery fees, ensuring all items are consolidated and delivered in one combined delivery for greater convenience. This service provides customers with the flexibility to shop across stores, compare prices, save on delivery fees, and receive all items in a single delivery. For stores, it can boost sales by encouraging larger orders, while delivery platforms benefit from reduced driver

demand and lower travel times. However, challenges arise in managing longer routes and extended delivery times when a single driver must detour to multiple stores for pickups and then deliver items to various locations. Transshipment, which enables drivers to coordinate and transfer items effectively at transshipment nodes, can mitigate this issue and further improve efficiency.

We develop a mixed-integer linear program for the multi-store order problem with consolidation and transshipment, which can also be adapted to variants without consolidation or transshipment. Solving the model that incorporates complex routing and time variables using exact methods is computationally intensive, particularly for large-scale instances. To overcome this problem, we implement a learning-to-optimize framework that combines neural network-based learning methods with a restoration and refinement optimization process. Using the learning-to-optimize method, the multi-store order problem can achieve high-quality solutions efficiently. In the learning process for allocation plans, we implement four methods, each with its own distinct advantages and disadvantages. The graph-based neural network generally shows superior performance, achieving a better trade-off between optimality gap and solution time while adapting well to larger scales not represented in the training set. This adaptability is due to its ability to exchange information among nodes, edges, and globally. The nearest driver allocation is the simplest to implement, requiring no training or historical samples. Both the driver classification neural network and driver assignment neural network perform well when the testing scale matches the training scale.

Through experiments conducted in various U.S. regions with varying customer locations and arrival frequencies, we find a trade-off between delivery time, which is the duration from order pickup to delivery, and wait time, which is the duration from order placement to pickup, indicating the existence of an optimal waiting strategy. It is beneficial to wait longer to batch more customers in cases of frequent customer arrivals and limited driver availability. Overall, the consolidated-order delivery system with transshipment and a five-minute waiting strategy consistently outperforms the others in terms of order completion time and driver travel time, regardless of customer arrival frequency or

driver availability relative to customer demand, due to its superior spatial and temporal consolidation.

Our work has some limitations that can be addressed in future research. It would be promising to formulate a more complex and realistic setting that includes transshipment costs, limited capacity, and potential errors. Transshipment nodes can be fixed locations or provided by stores or customers, each with associated costs and capacity limitations. Additionally, transshipment nodes may fail to open as scheduled, and deliveries before transfer could experience unexpected delays due to traffic or bad weather, causing further delays in both transfer and final delivery. Both the uncertainty in travel time and potential transshipment errors could be considered for more reliable on-time deliveries. In a rolling horizon implementation, earlier-arriving orders should be prioritized over later ones when driver availability is limited, and a piecewise linear function that increases with delivery time for each order could be applied. Future research can also explore stochastic and dynamic methods to address uncertain and time-dependent travel conditions in consolidated delivery services, enabling real-time order allocation and dynamically revised driver routing decisions. In our learning process, classification neural networks and graph neural networks are applied independently. However, other efficient machine learning algorithms may yield better performance. For instance, embedding algorithms, which train models independently before combining them into a stronger overall model, or ensemble learning algorithms, which sequentially train multiple models and aggregate their outputs, can enhance performance beyond that of any single model.

3.7 Appendix

Appendix A: Summary of Notation

The notation is presented in Table 3.5.

Table 3.5 – Problem Notation

Index	**Description**
$\mathscr{I}$	Set of customers
$\mathscr{J}$	Set of stores
$\mathscr{K}$	Set of drivers
$\mathscr{N}$	Set of nodes, including customers, stores, and driver initial locations
Parameters	**Description**
e_{ij}	Binary parameter indicating if customer i orders from store j
ij	Index of item ordered by customer i from store j
p_{ij}	Size of item ij
q^k	Capacity of driver k
$t_{nn'}$	Travel time between nodes n and n'
$[\alpha_i, \beta_i]$	Time window for customer i
Decisions	**Description**
z_n^k	Binary variable indicating if driver k visits node n
$x_{nn'}^k$	Binary variable indicating if driver k travels from node n to node n'
$y_{nn'}^{ij}$	Binary variable indicating if item ij travels from node n to node n' during the trip
v_n^{kij+}	Binary variable indicating if item ij arrives at node n via driver k
v_n^{kij-}	Binary variable indicating if item ij departs from node n via driver k
τ_n^{k+}	Continuous variable specifying the time driver k arrives at node n
τ_n^{k-}	Continuous variable specifying the time driver k departs from node n
λ_n^{ij}	Continuous variable specifying the arrival time of item ij at node n

Appendix B: Detailed Metrics of Learning and Optimization Processes

We present the detailed performance of each method under both uniformly sampled customer and clustered customers in this section. The learning performance for the uniformly sampled customers is shown in 3.6, and the re-optimization performance is presented in Table 3.7. The learning performance for the clustered customers is detailed in Table 3.8, with the corresponding optimization performance displayed in Table 3.9.

Table 3.6 – Comparison of Learning Methods for Uniformly Sampled Customers in the Learning Process

Training Scale	**Testing Scale**	**Method**	**Accuracy**	**Allocation Percentage**	**Allocation Std.**
$\|\mathscr{I}\|$: (2, 7); $\|\mathscr{K}\|$: (2, 3); NoI: 1200	$\|\mathscr{I}\|$: (2, 7); $\|\mathscr{K}\|$: (2, 3); NoI: 1200	NDA	0.79	41.67%	0.26
		DANN	0.82	44.02%	0.00
		DCNN	0.82	41.67%	0.18
		GNN	**0.84**	41.67%	0.30
	$\|\mathscr{I}\|$: (2, 7); $\|\mathscr{K}\|$: (4, 5); NoI: 1200	NDA	0.88	22.50%	0.39
		DANN	0.82	29.95%	0.59
		DCNN	0.82	22.50%	0.61
		GNN	**0.92**	22.50%	0.70
	$\|\mathscr{I}\|$: (8, 10); $\|\mathscr{K}\|$: (2, 3); NoI: 600	NDA	0.78	41.67%	0.94
		DANN	0.81	52.24%	0.28
		DCNN	0.84	41.67%	0.56
		GNN	**0.86**	41.67%	0.35
	$\|\mathscr{I}\|$: (8, 10); $\|\mathscr{K}\|$: (4, 5); NoI: 600	NDA	0.85	22.50%	0.02
		DANN	0.78	37.67%	0.42
		DCNN	0.81	22.50%	0.43
		GNN	**0.90**	22.50%	0.09
	$\|\mathscr{I}\|$: (12, 20); $\|\mathscr{K}\|$: (2, 5); NoI: 400	NDA	0.71	32.08%	0.97
		DANN	0.69	44.32%	0.50
		DCNN	0.69	32.08%	0.84
		GNN	**0.73**	32.08%	0.40
	$\|\mathscr{I}\|$: (2, 20); $\|\mathscr{K}\|$: (2, 5); NoI: 4000	NDA	0.83	32.08%	0.07
		DANN	0.81	39.09%	0.22
		DCNN	0.82	32.08%	0.59
		GNN	**0.87**	32.08%	0.15
$\|\mathscr{I}\|$: (2, 10); $\|\mathscr{K}\|$: (2, 5); NoI: 3600	$\|\mathscr{I}\|$: (2, 7); $\|\mathscr{K}\|$: (2, 3); NoI: 1200	NDA	0.79	41.67%	0.26
		DANN	0.81	40.24%	0.04
		DCNN	0.81	41.67%	0.09
		GNN	**0.83**	41.67%	0.31
	$\|\mathscr{I}\|$: (2, 7); $\|\mathscr{K}\|$: (4, 5); NoI: 1200	NDA	0.88	22.50%	0.39
		DANN	0.89	20.55%	0.40
		DCNN	0.91	22.50%	0.24
		GNN	**0.92**	22.50%	0.43
	$\|\mathscr{I}\|$: (8, 10); $\|\mathscr{K}\|$: (2, 3); NoI: 600	NDA	0.78	41.67%	0.81
		DANN	0.82	49.18%	0.26
		DCNN	**0.84**	41.67%	0.30
		GNN	0.82	41.67%	0.70
	$\|\mathscr{I}\|$: (8, 10); $\|\mathscr{K}\|$: (4, 5); NoI: 600	NDA	0.85	22.50%	0.02
		DANN	0.85	28.75%	0.34
		DCNN	**0.90**	22.50%	0.04
		GNN	0.89	22.50%	0.05
	$\|\mathscr{I}\|$: (12, 20); $\|\mathscr{K}\|$: (2, 5); NoI: 400	NDA	0.71	32.08%	0.20
		DANN	0.69	37.09%	0.04
		DCNN	0.70	32.08%	0.59
		GNN	**0.74**	32.08%	0.55
	$\|\mathscr{I}\|$: (2, 20); $\|\mathscr{K}\|$: (2, 5); NoI: 4000	NDA	0.83	32.08%	0.06
		DANN	0.84	32.38%	0.15
		DCNN	0.86	32.08%	0.05
		GNN	**0.88**	32.08%	0.15

Table 3.7 – Comparison of Learning Methods for Uniformly Sampled Customers in the Optimization Process

Training Scale	Testing Scale	Method	Objective Gap (%)	Delivery Time Gap (%)	Travel Time Gap (%)	Runtime (s)
$\|\mathscr{I}\|$: (2, 7); $\|\mathscr{K}\|$: (2, 3); NoI: 1200	$\|\mathscr{I}\|$: (2, 7); $\|\mathscr{K}\|$: (2, 3); NoI: 1200	NDA	3.06	3.06	1.18	7
		DANN	2.97	2.96	4.31	11
		DCNN	**2.07**	2.07	1.67	6
		GNN	3.21	3.2	1.55	6
	$\|\mathscr{I}\|$: (2, 7); $\|\mathscr{K}\|$: (4, 5); NoI: 1200	**NDA**	**1.25**	1.25	0.37	6
		DANN	5.53	5.48	43.79	26
		DCNN	10.04	10.01	11.9	10
		GNN	2.73	2.72	1.49	6
	$\|\mathscr{I}\|$: (8, 10); $\|\mathscr{K}\|$: (2, 3); NoI: 600	NDA	8.26	8.24	2.77	103
		DANN	6.89	6.89	11.04	200
		DCNN	3.78	3.78	2.19	28
		GNN	**3.19**	3.17	5.3	21
	$\|\mathscr{I}\|$: (8, 10); $\|\mathscr{K}\|$: (4, 5); NoI: 600	NDA	2.1	2.14	11.94	133
		DANN	4.2	4.24	23.79	206
		DCNN	12.19	12.28	27.11	86
		GNN	**1.89**	1.92	12.13	33
	$\|\mathscr{I}\|$: (12, 20); $\|\mathscr{K}\|$: (2, 5); NoI: 400	NDA	3.49	3.47	3.74	483
		DANN	5.48	5.47	1.84	607
		DCNN	2.29	2.33	18.38	532
		GNN	**1.73**	1.77	17.56	406
	$\|\mathscr{I}\|$: (2, 20); $\|\mathscr{K}\|$: (2, 5); NoI: 4000	NDA	2.81	2.8	2.32	50
		DANN	4.54	4.56	19.98	92
		DCNN	6.01	6.03	9.9	53
		GNN	**2.78**	2.78	3.8	38
$\|\mathscr{I}\|$: (2, 10); $\|\mathscr{K}\|$: (2, 5); NoI: 3600	$\|\mathscr{I}\|$: (2, 7); $\|\mathscr{K}\|$: (2, 3); NoI: 1200	NDA	3.06	3.06	1.18	7
		DANN	2.28	2.28	0.33	9
		DCNN	**2.10**	2.10	1.19	5
		GNN	3.07	3.07	0.77	5
	$\|\mathscr{I}\|$: (2, 7); $\|\mathscr{K}\|$: (4, 5); NoI: 1200	NDA	1.25	1.25	0.37	6
		DANN	1.49	1.48	4.36	8
		DCNN	**0.94**	0.94	0.15	5
		GNN	2.19	2.19	3.22	5
	$\|\mathscr{I}\|$: (8, 10); $\|\mathscr{K}\|$: (2, 3); NoI: 600	NDA	8.26	8.24	2.77	103
		DANN	4.03	4.01	6.57	154
		DCNN	**1.66**	1.65	4.92	14
		GNN	2.82	2.81	4.46	12
	$\|\mathscr{I}\|$: (8, 10); $\|\mathscr{K}\|$: (4, 5); NoI: 600	NDA	2.1	2.14	11.94	133
		DANN	1.89	1.91	7.17	133
		DCNN	**1.66**	1.68	10.42	30
		GNN	2.13	2.15	7.4	25
	$\|\mathscr{I}\|$: (12, 20); $\|\mathscr{K}\|$: (2, 5); NoI: 400	NDA	3.49	3.47	3.74	483
		DANN	5.20	5.18	2.99	607
		DCNN	1.63	1.67	17.51	528
		GNN	**1.19**	1.23	16.76	457
	$\|\mathscr{I}\|$: (2, 20); $\|\mathscr{K}\|$: (2, 5); NoI: 4000	NDA	2.81	2.8	2.32	50
		DANN	2.32	2.31	2.76	67
		DCNN	**1.56**	1.56	2.75	40
		GNN	2.52	2.51	1.39	35

Table 3.8 – Comparison of Learning Methods for Clustered Customers in the Learning Process

Training Scale	Testing Scale	Method	Accuracy	Allocation Percentage	Allocation Std.
$\|\mathscr{I}\|$: (2, 7); $\|\mathscr{K}\|$: (2, 3)); NoI: 1200	$\|\mathscr{I}\|$: (2, 7); $\|\mathscr{K}\|$: (2, 3); NoI: 1200	NDA	0.64	41.67%	0.30
		DANN	0.73	37.43%	0.01
		DCNN	0.77	41.67%	0.58
		GNN	**0.81**	41.67%	0.19
	$\|\mathscr{I}\|$: (2, 7); $\|\mathscr{K}\|$: (4, 5); NoI: 1200	NDA	0.78	22.50%	0.90
		DANN	0.76	23.11%	0.51
		DCNN	0.82	22.50%	0.26
		GNN	**0.89**	22.50%	0.35
	$\|\mathscr{I}\|$: (8, 10); $\|\mathscr{K}\|$: (2, 3); NoI: 600	NDA	0.58	41.67%	0.7
		DANN	0.63	63.14%	0.12
		DCNN	0.63	41.67%	0.67
		GNN	**0.70**	41.67%	0.76
	$\|\mathscr{I}\|$: (8, 10); $\|\mathscr{K}\|$: (4, 5); NoI: 600	NDA	0.8	22.50%	1.36
		DANN	0.70	27.04%	0.67
		DCNN	0.80	22.50%	0.58
		GNN	**0.86**	22.50%	0.30
	$\|\mathscr{I}\|$: (12, 20); $\|\mathscr{K}\|$: (2, 5); NoI: 400	NDA	0.60	32.08%	0.91
		DANN	0.63	68.31%	0.566
		DCNN	0.63	32.08%	0.76
		GNN	**0.69**	32.08%	0.17
	$\|\mathscr{I}\|$: (2, 20); $\|\mathscr{K}\|$: (2, 5); NoI: 4000	NDA	0.70	32.08%	0.54
		DANN	0.73	33.81%	0.27
		DCNN	0.78	32.08%	0.55
		GNN	**0.83**	32.08%	0.94
$\|\mathscr{I}\|$: (2, 10); $\|\mathscr{K}\|$: (2, 5); NoI: 3600	$\|\mathscr{I}\|$: (2, 7); $\|\mathscr{K}\|$: (2, 3); NoI: 1200	NDA	0.64	41.67%	0.30
		DANN	0.72	28.49%	0.51
		DCNN	0.77	41.67%	0.93
		GNN	**0.80**	41.67%	0.57
	$\|\mathscr{I}\|$: (2, 7); $\|\mathscr{K}\|$: (4, 5); NoI: 1200	NDA	0.78	22.5%	0.90
		DANN	0.86	17.48%	1.37
		DCNN	0.90	22.50%	0.03
		GNN	**0.90**	22.50%	0.09
	$\|\mathscr{I}\|$: (8, 10); $\|\mathscr{K}\|$: (2, 3); NoI: 600	NDA	0.58	41.67%	0.11
		DANN	0.65	38.15%	0.19
		DCNN	0.72	41.67%	0.21
		GNN	**0.73**	41.67%	0.29
	$\|\mathscr{I}\|$: (8, 10); $\|\mathscr{K}\|$: (4, 5); NoI: 600	NDA	0.80	18.71%	0.35
		DANN	0.84	18.53%	0.92
		DCNN	0.86	18.71%	0.07
		GNN	**0.88**	18.71%	0.45
	$\|\mathscr{I}\|$: (12, 20); $\|\mathscr{K}\|$: (2, 5); NoI: 400	NDA	0.60	32.08%	0.28
		DANN	0.66	31.11%	0.59
		DCNN	0.66	32.08%	0.30
		GNN	**0.69**	32.08%	0.76
	$\|\mathscr{I}\|$: (2, 20); $\|\mathscr{K}\|$: (2, 5); NoI: 4000	NDA	0.70	32.08%	0.61
		DANN	0.78	29.31%	0.88
		DCNN	0.82	32.08%	0.63
		GNN	**0.84**	32.08%	0.51

Table 3.9 – Comparison of Learning Methods for Clustered Customers in the Optimization Process

Training Scale	Testing Scale	Method	Objective Gap(%)	Delivery Time Gap(%)	Travel Time Gap(%)	Runtime (s)
$\|\mathscr{I}\|$: (2, 7); $\|\mathscr{K}\|$: (2, 3); NoI: 1200	$\|\mathscr{I}\|$: (2, 7); $\|\mathscr{K}\|$: (2, 3); NoI: 1200	NDA	3.64	3.58	25.77	14
		DANN	2.94	2.92	8.61	24
		DCNN	3.54	3.53	9.96	10
		GNN	**2.35**	2.33	4.3	7
	$\|\mathscr{I}\|$: (2, 7); $\|\mathscr{K}\|$: (4, 5); NoI: 1200	NDA	2.74	2.62	62.36	19
		DANN	3.40	3.28	58.73	43
		DCNN	4.90	4.87	9.08	13
		GNN	**1.79**	1.73	24.09	8
	$\|\mathscr{I}\|$: (8, 10); $\|\mathscr{K}\|$: (2, 3); NoI: 600	NDA	10.71	10.69	12.31	152
		DANN	11.72	11.69	20.47	311
		DCNN	4.35	4.30	14.55	82
		GNN	**2.90**	2.89	3.57	15
	$\|\mathscr{I}\|$: (8, 10); $\|\mathscr{K}\|$: (4, 5); NoI: 600	NDA	5.72	5.63	39.31	127
		DANN	4.21	4.11	41.43	264
		DCNN	6.46	6.38	23.95	98
		GNN	**1.20**	1.18	7.55	31
	$\|\mathscr{I}\|$: (12, 20); $\|\mathscr{K}\|$: (2, 5); NoI: 400	NDA	11.62	11.57	3.4	574
		DANN	10.62	10.59	12.5	607
		DCNN	4.06	4.00	17.51	542
		GNN	**2.36**	2.30	16.76	457
	$\|\mathscr{I}\|$: (2, 20); $\|\mathscr{K}\|$: (2, 5); NoI: 4000	NDA	4.65	4.58	36.19	47
		DANN	4.53	4.48	29.79	109
		DCNN	4.39	4.36	12.09	55
		GNN	**2.07**	2.07	9.01	38
$\|\mathscr{I}\|$: (2, 10); $\|\mathscr{K}\|$: (2, 5); NoI: 3600	$\|\mathscr{I}\|$: (2, 7); $\|\mathscr{K}\|$: (2, 3); NoI: 1200	NDA	3.64	3.58	25.77	14
		DANN	**1.88**	1.86	2.08	41
		DCNN	4.56	4.55	9.97	7
		GNN	2.65	2.64	3.17	6
	$\|\mathscr{I}\|$: (2, 7); $\|\mathscr{K}\|$: (4, 5); NoI: 1200	NDA	2.74	2.62	62.36	19
		DANN	**0.23**	0.23	4.34	50
		DCNN	2.75	2.72	4.14	6
		GNN	1.68	1.64	23.44	5
	$\|\mathscr{I}\|$: (8, 10); $\|\mathscr{K}\|$: (2, 3); NoI: 600	NDA	10.71	10.69	12.31	152
		DANN	6.22	6.22	5.96	329
		DCNN	5.18	5.15	11.74	83
		GNN	**1.16**	1.12	11.79	12
	$\|\mathscr{I}\|$: (8, 10); $\|\mathscr{K}\|$: (4, 5); NoI: 600	NDA	5.72	5.63	39.31	127
		DANN	1.18	1.13	12.16	163
		DCNN	2.99	2.97	11.53	35
		GNN	**1.01**	0.97	21.48	24
	$\|\mathscr{I}\|$: (12, 20); $\|\mathscr{K}\|$: (2, 5); NoI: 400	NDA	11.62	11.57	3.40	574
		DANN	6.52	6.50	10.02	607
		DCNN	2.08	2.03	17.51	525
		GNN	**1.74**	1.69	16.76	450
	$\|\mathscr{I}\|$: (2, 20); $\|\mathscr{K}\|$: (2, 5); NoI: 4000	NDA	4.65	4.58	36.19	47
		DANN	**1.93**	1.90	1.47	111
		DCNN	3.60	3.57	8.89	46
		GNN	1.96	1.95	11.10	34

Appendix C: Dynamic Experiments

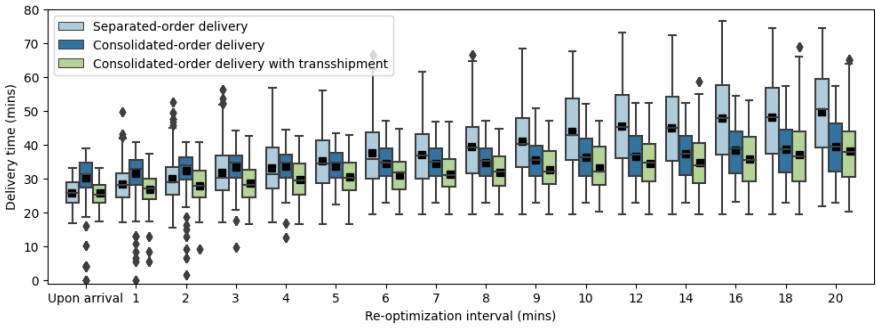
Appendix C1: Dynamic Experiments with Earliest-Available-Driver-Assignment Strategy

The detailed experimentation process for earliest-available-driver-assignment strategy is presented in Table 3.10.

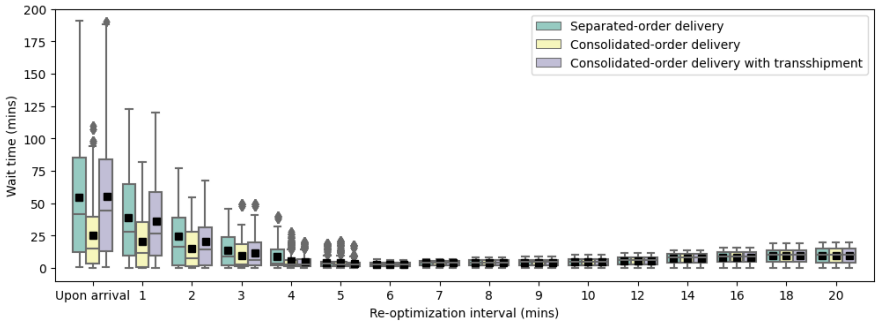
Table 3.10 – Dynamic Experimentation Process

Step	Description
Initialization	Initialize the experimentation with the following elements: • Define the customer arrival process as a Poisson process with a specified arrival rate. Set service duration $\lvert\mathscr{T}\rvert$ to 1 or 2 hours. Generate customer arrival times a_i for customer $i \in \mathscr{I}$ arriving within this duration. • Let $\lvert\mathscr{K}\rvert$ available drivers serve customers who order from store $j \in \lvert\mathscr{J}\rvert$. For driver $k \in \mathscr{K}$, generate their origin node and set availability time $\tau_{o_k} = 0$. Additionally, determine the locations of both customers and stores. • Define the re-optimization intervals, which can vary from event-triggered (e.g., upon a new customer arrival) to a fixed interval (e.g., 10 minutes). Obtain the set of optimization time points $\{p_0, p_1, p_2, \ldots, p_T\}$, and define the final time P for the experimentation.
Step 1	For the initial period $t = 0$, corresponding to the time interval $[p_0, p_1]$, optimize the system at time point p_1, • Input the information, including the customers that arrive within this period (i.e., $i \in \mathscr{I}_0$ where $\mathscr{I}_0 = \{i \in \mathscr{I} \mid p_0 \le a_i \le p_1\}$) and place orders from stores $j \in \mathscr{J}$, as well as the drivers $k \in \mathscr{K}$ with their available times to serve customers being $\tau_{o_k} = 0$. • Run the optimization problem to assign drivers to customers and plan their routes. If no drivers are currently available, customers are assigned to the earliest available drivers who can serve the orders once they complete their assigned tasks. • Update the drivers' availability times based on the completion time of their last served customer (i.e., $\tau_{o_k} \leftarrow \max_{i \in I_0} \left\{\tau_i^{k-}\right\}$), and set the driver location to the last served customer location. • Record the number of customers served in the interval $[p_0, p_1]$, as well as the completion time, wait time, delivery time, and expected travel time for each customer.
Step 2	While $t \le \lvert\mathscr{T}\rvert - 1$, repeat the following steps: Increment t and update the time interval to $[p_t, p_{t+1}]$. Optimize the system at the time point p_{t+1}, • Input the information, including the customers that arrive within this period (i.e., $i \in \mathscr{I}_t$ where $\mathscr{I}_t = \{i \in \mathscr{I} \mid p_t \le a_i \le p_{t+1}\}$) and place orders from stores $j \in \mathscr{J}$, as well as the drivers $k \in \mathscr{K}$ with their earliest available times to serve customers being τ_{o_k}. • Run the optimization problem to assign drivers to customers and plan their routes. If no drivers are currently available, customers are assigned to the earliest available drivers who can serve the orders once they complete their assigned tasks. • Update the drivers' availability times based on the completion time of their last served customer (i.e., $\tau_{o_k} \leftarrow \max_{i \in I_t} \{\tau_i^{k-}\}$), and set the driver location to the last served customer location. • Record the number of customers served in the interval $[p_t, p_{t+1}]$, as well as the completion time, wait time, delivery time, and expected travel time for each customer.
Output	The experimentation outputs include the total number of customers served during each time interval, and for each customer, the order completion time, wait time, delivery time, and expected travel time.

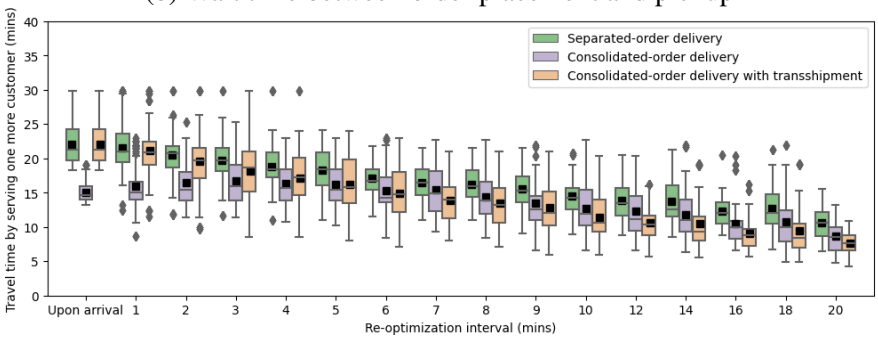
The dynamic experimentation results for earliest-available-driver-assignment strategy are shown in Figure 3.10.



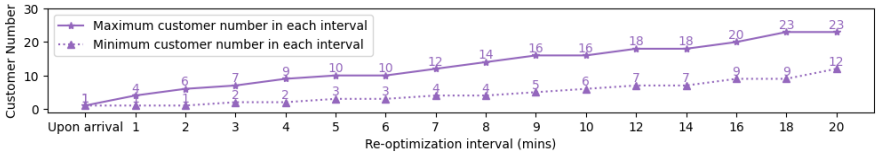
(a) Delivery time between order pickup and arrival



(b) Wait time between order placement and pickup



(c) Driver travel time by serving one more customer



(d) Number of customers in each re-optimization interval

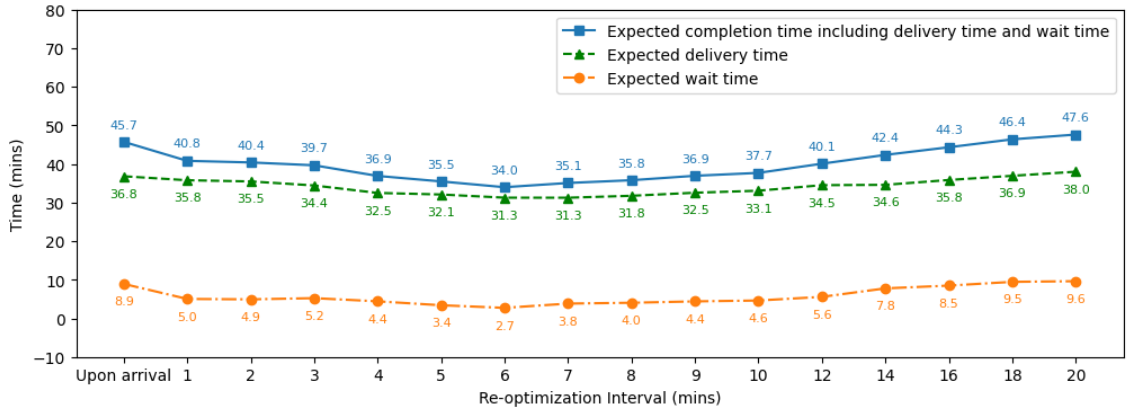
Figure 3.10 – Delivery time, wait time, travel time, and customer number under varying re-optimization intervals

Appendix C2: Dynamic Experiments with Driver-Availability-Triggered Strategy

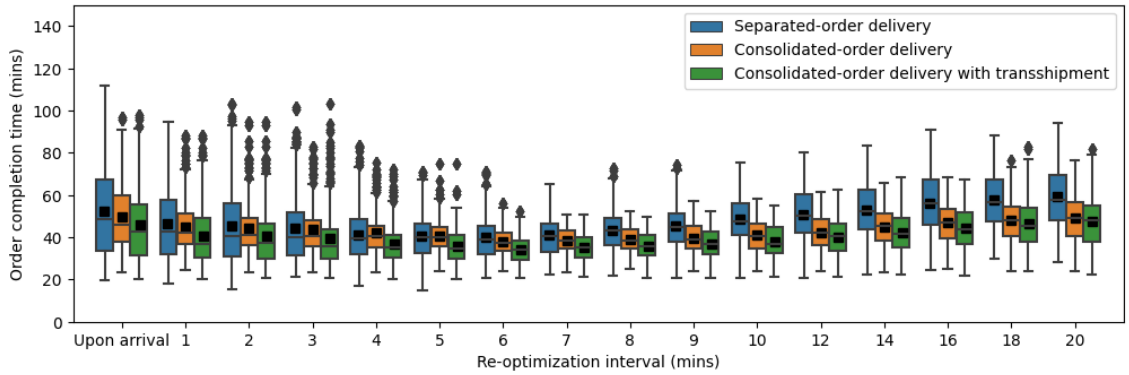
We present the dynamic experimentation process with driver-availability-triggered optimization in Table 3.11.

Table 3.11 – Dynamic Experimentation Process with Driver-Availability-Triggered Optimization

Step	Description
Initialization	Initialize the experimentation with the following elements: • Define the customer arrival process as a Poisson process with a specified arrival rate. Set service duration $\lvert\mathscr{T}\rvert$ to 1 or 2 hours. Generate customer arrival times a_i for each customer $i \in \mathscr{I}$ who arrives within this duration. • Let $\lvert\mathscr{K}\rvert$ available drivers serve customers who order from store $j \in \lvert\mathscr{J}\rvert$. For driver $k \in \mathscr{K}$, generate their origin node and set availability time $\tau_{o_k} = 0$. Additionally, determine the locations of both customers and stores. • Define the re-optimization intervals, which can vary from event-triggered (e.g., upon a new customer arrival) to a fixed interval (e.g., 10 minutes). Obtain the set of optimization time points $\{p_0, p_1, p_2, \ldots, p_T\}$, and define the final time P for the experimentation.
Step 1	For the initial period $t = 0$, corresponding to the time interval $[p_0, p_1]$, if there are available drivers $k \in \{k \in \mathscr{K} \mid o_k \leq p_1\}$, optimize the system at the time point p_1, • Input the information, including the customers that arrive within this period (i.e., $i \in \mathscr{I}_0$ where $\mathscr{I}_0 = \{i \in \mathscr{I} \mid p_0 \leq a_i \leq p_1\}$) and place orders from stores $j \in \mathscr{J}$, as well as the drivers $k \in \mathscr{K}$ with their available times to serve customers being $\tau_{o_k} = 0$. • Run the optimization problem to assign available drivers to customers and plan their routes. • Update the drivers' availability times based on the completion time of their last served customer (i.e., $\tau_{o_k} \leftarrow \max_{i \in I_0} \left\{\tau_i^{k-}\right\}$), and set the driver location to the last served customer location. • Record the number of customers served in the interval $[p_0, p_1]$, as well as the completion time, wait time, delivery time, and expected travel time for each customer.
Step 2	While $t \leq \lvert\mathscr{T}\rvert - 1$, repeat the following steps: Increment t and update the time interval to $[p_t, p_{t+1}]$. If there are available drivers $k \in \{k \in \mathscr{K} \mid o_k \leq p_{t+1}\}$, optimize the system at the time point p_{t+1}, • Input the information, including the customers that arrive within this period (i.e., $i \in \mathscr{I}_t$ where $\mathscr{I}_t = \{i \in \mathscr{I} \mid p_t \leq a_i \leq p_{t+1}\}$) and place orders from stores $j \in \mathscr{J}$, as well as the drivers $k \in \mathscr{K}$ with their earliest available times to serve customers being τ_{o_k}. • Run the optimization problem to assign available drivers to customers and plan their routes. • Update the drivers' availability times based on the completion time of their last served customer (i.e., $\tau_{o_k} \leftarrow \max_{i \in I_t} \{\tau_i^{k-}\}$), and set the driver location to the last served customer location. • Record the number of customers served in the interval $[p_t, p_{t+1}]$, as well as the completion time, wait time, delivery time, and expected travel time for each customer.
Output	The experimentation outputs include the total number of customers served during each time interval, and for each customer, the order completion time, wait time, delivery time, and expected travel time.



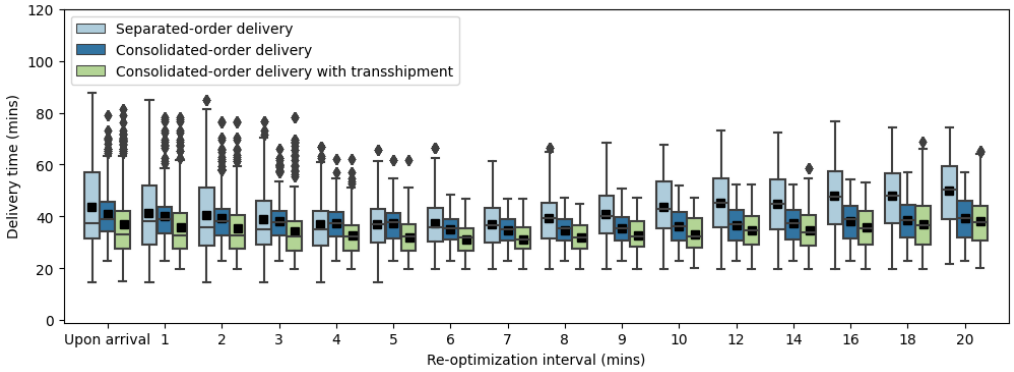
(a) Completion time, wait time, and delivery time for consolidated-order delivery with transshipment



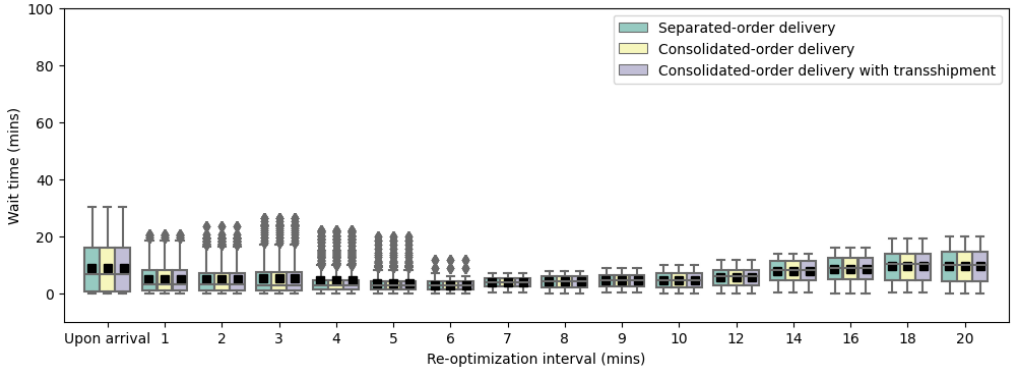
(b) Completion time including wait time and delivery time

Figure 3.11 – Completion time under varying re-optimization intervals for the driver-availability-triggered strategy

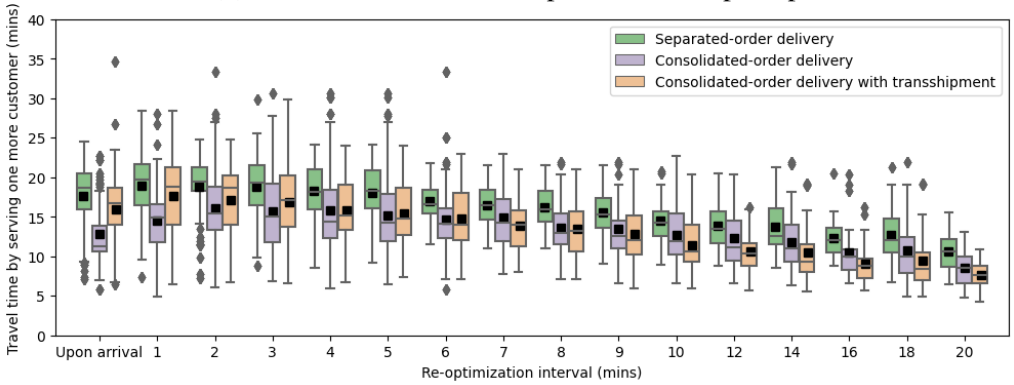
Under the dynamic experimentation with the driver-availability-triggered strategy, Figure 3.11(a) presents the order completion time, including delivery and wait times, for consolidated-order delivery with transshipment (CODT). Although the trade-off between delivery and wait times disappears under this strategy, the order completion time still follows a smooth pattern, with trends that initially decrease and then increase. The five-minute re-optimization interval yields the lowest delivery time, while the six-minute interval achieves the lowest wait time and overall order completion times. Figure 3.11(b) compares completion times across the three systems under varying intervals, while Figure 3.12 provides detailed distributions of delivery time, wait time, travel time per customer, and customer scale. CODT consistently outperforms the other systems in order comple-



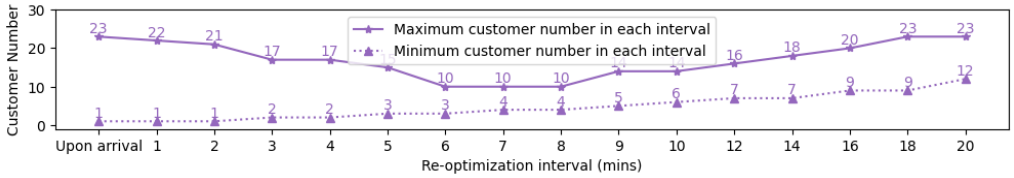
(a) Delivery time between order pickup and arrival



(b) Wait time between order placement and pickup

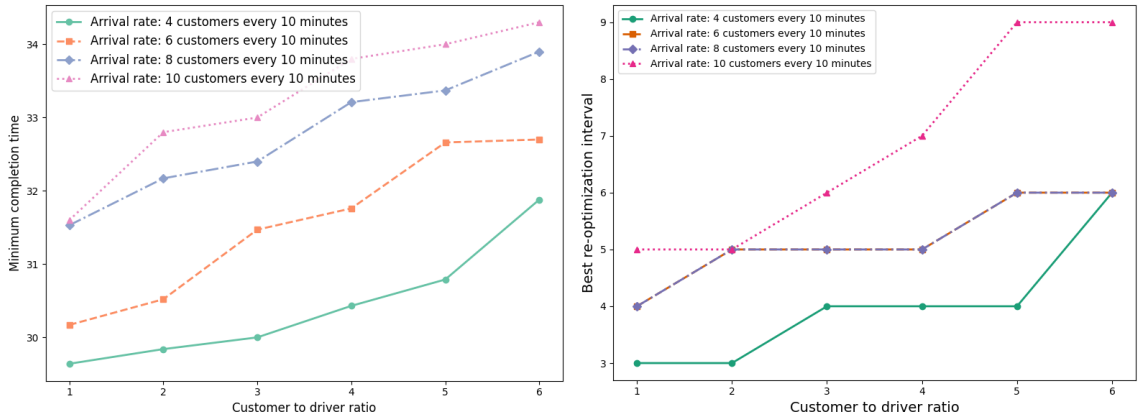


(c) Driver travel time by serving one more customer



(d) Number of customers in each re-optimization interval

Figure 3.12 – Delivery time, wait time, travel time, and customer number under varying re-optimization intervals for the driver-availability-triggered strategy



(a) Minimum completion time

(b) Best re-optimization interval

Figure 3.13 – Best re-optimization interval for varying ratios of customer number to driver number

tion time, delivery time, and driver travel time per customer. Figure 3.13 highlights the best re-optimization interval that minimizes order completion time across varying customer arrival rates and customer-to-driver ratios. As arrival rates or customer-to-driver ratios increase, longer re-optimization intervals are optimal, reflecting the benefit of waiting longer to batch and fulfill orders. These findings align with insights from the earliest-available-driver-assignment strategy.

Appendix C3: Comparison of the Two Strategies

We compare the earliest-available-driver-assignment strategy with the driver-availability-triggered strategy in this section. As shown in Figures 3.8(a) and 3.11(a), the best completion time under the driver-availability-triggered strategy, at 34.0 minutes, is higher than the best time of 33.6 minutes under the earliest-available-driver-assignment strategy. Therefore, we conclude that fixed-interval optimization with the earliest-available-driver-assignment strategy is slightly superior to the driver-availability-triggered strategy.

Table 3.12 presents the best re-optimization intervals and corresponding minimum completion times across different customer arrival rates and customer-to-driver ratios, highlighting how the optimal waiting strategy varies. Overall, the earliest-available-

driver-assignment strategy consistently outperforms the driver-availability-triggered strategy, though the gap is small. The best re-optimization interval for the driver-availability-triggered strategy is approximately one minute longer, as both shorter and longer intervals can result in larger batching sizes and longer completion times in this approach. In summary, a shorter-duration waiting strategy is optimal for the consolidated-order delivery system with transshipment.

Customer to Driver Ratio	**Arrival Rate (Num. of customers every 10 mins)**	**Earliest-available-driver-assignment Strategy**		**Driver-availability-triggered Strategy**	
		Best Re-optimization Interval (mins)	**Minimum Completion Time (mins)**	**Best Re-optimization Interval (mins)**	**Minimum Completion Time (mins)**
1	4	3	29.40	3	29.64
1	6	4	30.14	4	30.17
1	8	4	30.99	4	31.53
1	10	4	31.42	5	31.60
2	4	3	29.45	3	29.84
2	6	4	30.44	5	30.52
2	8	4	31.61	5	32.17
2	10	6	32.61	5	32.80
3	4	4	30.04	4	30.00
3	6	5	31.19	5	31.47
3	8	5	32.63	5	32.40
3	10	6	32.85	6	33.00
4	4	4	30.60	4	30.43
4	6	5	31.58	5	31.76
4	8	5	33.00	5	33.21
4	10	6	33.20	7	33.80
5	4	4	30.67	4	30.79
5	6	5	32.04	6	32.66
5	8	5	33.27	6	33.37
5	10	9	33.62	9	34.00
6	4	4	31.29	6	31.88
6	6	6	32.30	6	32.70
6	8	6	33.88	6	33.90
6	10	9	34.30	9	34.30

Table 3.12 – Comparison between Earliest-Available-Driver-Assignment Strategy and Driver-Availability-Triggered Strategy

References

Auad, Ramon, Alan Erera, and Martin Savelsbergh (2024). "Dynamic courier capacity acquisition in rapid delivery systems: A deep Q-learning approach". In: *Transportation Science* 58.1, pp. 67–93.

Berbeglia, Gerardo, Jean-François Cordeau, and Gilbert Laporte (2010). "Dynamic pickup and delivery problems". In: *European Journal of Operational Research* 202.1, pp. 8–15.

Bogyrbayeva, Aigerim et al. (2024). "Machine learning to solve vehicle routing problems: A survey". In: *IEEE Transactions on Intelligent Transportation Systems* 25.6, pp. 4754–4772.

Cao, Junyu and Wei Qi (2023). "Stall economy: The value of mobility in retail on wheels". In: *Operations Research* 71.2, pp. 708–726.

Carlsson, John Gunnar et al. (2024). "Provably good region partitioning for on-time last-mile delivery". In: *Operations Research* 72.1, pp. 91–109.

Dantzig, George B and John H Ramser (1959). "The truck dispatching problem". In: *Management Science* 6.1, pp. 80–91.

DoubleDash (2023). *Introducing DoubleDash, a new way to shop multiple stores in one order*. Last accessed on Nov 01, 2023. URL: `https://about.doordash.com/en-us/news/introducing-doubledash-a-new-way-to-shop-multiple-stores-in-one-order`.

Epipresto (2023). *The place to order from all your specialized and independent stores!* Last accessed on Nov 01, 2023. URL: `https://epipresto.ca/en`.

Hildebrandt, Florentin D and Marlin W Ulmer (2022). "Supervised learning for arrival time estimations in restaurant meal delivery". In: *Transportation Science* 56.4, pp. 1058–1084.

Hong, L Jeff, Zhiyuan Huang, and Henry Lam (2021). "Learning-based robust optimization: Procedures and statistical guarantees". In: *Management Science* 67.6, pp. 3447–3467.

Instacart (2022). *Introducing OrderUp, a new way to save time and money on Instacart with orders from two retailers– with just one delivery fee*. Last accessed on Nov 01, 2023. URL: https://www.instacart.com/company/updates/introducing-orderup-a-new-way-to-save-time-and-money-on-instacart-with-orders-from-two-retailers-with-just-one-delivery-fee/.

Julien, Esther, Krzysztof Postek, and Ş İlker Birbil (2024). "Machine learning for k-adaptability in two-stage robust optimization". In: *INFORMS Journal on Computing* Forthcoming. URL: https://doi.org/10.1287/ijoc.2022.0314.

Koç, Çağrı, Gilbert Laporte, and İlknur Tükenmez (2020). "A review of vehicle routing with simultaneous pickup and delivery". In: *Computers & Operations Research* 122, p. 104987.

Kotary, James, Vincenzo Di Vito, et al. (2024). "Learning Joint Models of Prediction and Optimization". In: *ECAI 2024*. IOS Press, pp. 2476–2483.

Kotary, James, Ferdinando Fioretto, et al. (Aug. 2021). "End-to-end constrained optimization learning: A survey". In: *International Joint Conference on Artificial Intelligence*. IJCAI, pp. 4475–4482.

Larsen, Eric, Emma Frejinger, et al. (2024). "Fast continuous and integer L-shaped heuristics through supervised learning". In: *INFORMS Journal on Computing* 36.1, pp. 203–223.

Larsen, Eric, Sébastien Lachapelle, et al. (2022). "Predicting tactical solutions to operational planning problems under imperfect information". In: *INFORMS Journal on Computing* 34.1, pp. 227–242.

Li, Zhuoxin and Gang Wang (2024). "On-Demand Delivery Platforms and Restaurant Sales". In: *Management Science* Forthcoming. URL: https://doi.org/10.1287/mnsc.2021.01010.

Liu, Sheng, Long He, and Zuo-Jun Max Shen (2021). "On-time last-mile delivery: Order assignment with travel-time predictors". In: *Management Science* 67.7, pp. 4095–4119.

Lyu, Zefeng and Andrew Junfang Yu (2023). "The pickup and delivery problem with transshipments: Critical review of two existing models and a new formulation". In: *European Journal of Operational Research* 305.1, pp. 260–270.

Mao, Wenzheng et al. (2022). "On-demand meal delivery platforms: Operational level data and research opportunities". In: *Manufacturing & Service Operations Management* 24.5, pp. 2535–2542.

Maragno, Donato et al. (2023). "Mixed-integer optimization with constraint learning". In: *Operations Research* Forthcoming. URL: https://doi.org/10.1287/opre.2021.0707.

Merchan, Daniel et al. (2021). *Amazon Last Mile Routing Research Challenge Dataset*. Accessed January 6, 2022. Seattle: Amazon.com. URL: https://registry.opendata.aws/amazon-last-mile-challenges.

Mitrović-Minić, Snežana and Gilbert Laporte (2006). "The pickup and delivery problem with time windows and transshipment". In: *INFOR: Information Systems and Operational Research* 44.3, pp. 217–227.

Nowak, Maciek, Ozlem Ergun, and Chelsea C White III (2009). "An empirical study on the benefit of split loads with the pickup and delivery problem". In: *European Journal of Operational Research* 198.3, pp. 734–740.

Ojha, Ritesh et al. (2023). "Optimization-based Learning for Dynamic Load Planning in Trucking Service Networks". In: *arXiv preprint arXiv:2307.04050*.

Özarık, Sami Serkan, Paulo da Costa, and Alexandre M Florio (2024). "Machine learning for data-driven last-mile delivery optimization". In: *Transportation Science* 58.1, pp. 27–44.

Qi, Meng et al. (2023). "A practical end-to-end inventory management model with deep learning". In: *Management Science* 69.2, pp. 759–773.

Raghavan, S and Rui Zhang (2024). "The driver-aide problem: Coordinated logistics for last-mile delivery". In: *Manufacturing & Service Operations Management* 26.1, pp. 291–311.

Rais, Abdur, Filipe Alvelos, and Maria Sameiro Carvalho (2014). "New mixed integer-programming model for the pickup-and-delivery problem with transshipment". In: *European Journal of Operational Research* 235.3, pp. 530–539.

Rieck, Julia, Carsten Ehrenberg, and Jürgen Zimmermann (2014). "Many-to-many location-routing with inter-hub transport and multi-commodity pickup-and-delivery". In: *European Journal of Operational Research* 236.3, pp. 863–878.

Sadana, Utsav et al. (2024). "A survey of contextual optimization methods for decision-making under uncertainty". In: *European Journal of Operational Research*.

Savelsbergh, Martin and Tom Van Woensel (2016). "50th anniversary invited article—city logistics: Challenges and opportunities". In: *Transportation Science* 50.2, pp. 579–590.

Statista (2024). *Market insights into quick commerce of online food and grocery delivery (Worldwide)*. Last accessed on Oct 15, 2024. URL: `https://www.statista.com/outlook/dmo/online-food-delivery/worldwide`.

Su, E et al. (2023). "An exact algorithm for the pickup and delivery problem with crowd-sourced bids and transshipment". In: *Transportation Research Part B: Methodological* 177, p. 102831.

Ulmer, Marlin W et al. (2021). "The restaurant meal delivery problem: Dynamic pickup and delivery with deadlines and random ready times". In: *Transportation Science* 55.1, pp. 75–100.

Van Hentenryck, Pascal (2021). "Machine learning for optimal power flows". In: *Tutorials in Operations Research: Emerging Optimization Methods and Modeling Techniques with Applications*, pp. 62–82.

Wang, Shanshan, Erick Delage, and Leandro C Coelho (2024). "Data-driven stochastic vehicle routing problems with dead-lines". In: *Les Cahiers du GERAD ISSN* 711, p. 2440.

General Conclusion

Last-mile delivery, the final step in the e-commerce supply chain, is important and indispensable, driving extensive research focused on improving service quality, reducing travel costs, and enhancing operational efficiency. To provide on-time and on-demand deliveries efficiently with low costs and high reliability, we investigate three challenging delivery problems. First, crowdkeeping delivery services utilize unused crowd space for temporary parcel storage to reduce delivery costs, eliminate failed deliveries, and benefit all system participants. Second, ultra-fast delivery services are designed for rapid and reliable delivery under uncertainties in travel times and demand arrival times, striking a balance between the delivery speed and operational profitability. Finally, multi-store order delivery services with consolidation and transshipment ensure fast deliveries and short travel times for fulfilling all requests, while also incorporating waiting strategies to facilitate temporal consolidation. Mathematical programs, including bilevel programs, probabilistic envelope constrained programs, and mixed-integer linear programs, are developed to tackle these challenging problems. These models are solved efficiently by reformulating them into tractable programs and using advanced methods such as state-of-the-art solvers, the row generation algorithm, or learning-to-optimize approaches that integrate machine learning with optimization. By leveraging efficient solution techniques and conducting numerical experiments on real-world datasets, this work offers managerial insights in delivery strategies that can enhance the efficiency, reliability, and customer satisfaction of last-mile logistics.

This study opens several opportunities for future research. In Chapter 1, the crowd-

keeping delivery model assumes complete information and deterministic conditions. Future work could address uncertainties in customer density and keeper availability, as well as multi-period optimization for dynamic service allocation among evolving groups of customers and keepers, and the coexistence of various delivery types under a customer choice model. In Chapter 2, the ultra-fast delivery model currently assumes unlimited drivers and immediate delivery services from depots to customers. However, a more accurate model could further account for driver routing, customer batching, product assortment, and customer utility variation in stochastic programming to enhance realism and applicability. In Chapter 3, the consolidated multi-store delivery model could be improved by considering transshipment costs and capacity, uncertain travel times, potential transfer errors, and the rolling horizon rescheduling of unfulfilled earlier-arriving orders. Additionally, integrating advanced machine learning techniques, such as embedding or ensemble learning algorithms, can improve solution quality and scalability. Together, these extensions could further refine the models and enhance their relevance to practical last-mile logistics applications.

Moreover, the synergies between these three business models present an interesting avenue for further exploration. While each model is designed to optimize delivery in distinct ways, their integration could offer a comprehensive solution to last-mile logistics challenges. For example, a hybrid approach that combines consolidated orders from multiple stores, utilizes crowd space as transshipment nodes or temporary receivers, and incorporates hierarchical service levels under uncertain travel times and order arrival times could significantly enhance operational efficiency, service reliability, and customer satisfaction. A key challenge for future research is whether these models can be implemented simultaneously while adapting well to marketing needs, or whether they can be efficiently optimized to deliver high-quality solutions.

References

Auad, Ramon, Alan Erera, and Martin Savelsbergh (2024). "Dynamic courier capacity acquisition in rapid delivery systems: A deep Q-learning approach". In: *Transportation Science* 58.1, pp. 67–93.

CaiNiao (2023). *How to make money by being a keeper? (In Chinese)*. Last accessed on Jan 01, 2023. URL: https://cn.alicdn.com/video/cainiao/yizhan/main.mp4.

Cao, Junyu and Wei Qi (2023). "Stall economy: The value of mobility in retail on wheels". In: *Operations Research* 71.2, pp. 708–726.

Carlsson, John Gunnar et al. (2024). "Provably good region partitioning for on-time last-mile delivery". In: *Operations Research* 72.1, pp. 91–109.

DoubleDash (2023). *Introducing DoubleDash, a new way to shop multiple stores in one order*. Last accessed on Nov 01, 2023. URL: https://about.doordash.com/en-us/news/introducing-doubledash-a-new-way-to-shop-multiple-stores-in-one-order.

Epipresto (2023). *The place to order from all your specialized and independent stores!* Last accessed on Nov 01, 2023. URL: https://epipresto.ca/en.

Getir (2022). *Getir: groceries in minutes*. Last accessed on Nov 01, 2023. URL: https://getir.com/us/.

Goodfood (2022). *Goodfood is in financial trouble and gives up fast delivery (In French)*. Last accessed on Aug 01, 2023. URL: https://www.lapresse.ca/affaires/

entreprises/2022-10-14/goodfood-a-des-ennuis-financiers-et-laisse-tomber-la-livraison-rapide.php.

Gorillas (2022). *Grocery app Gorillas drops 10-minute delivery pledge, adds store pick-up option*. Last accessed on Aug 01, 2023. URL: https://nypost.com/2022/02/25/grocery-app-gorillas-drops-10-minute-delivery-pledge-adds-store-pick-up-option/.

Hildebrandt, Florentin D and Marlin W Ulmer (2022). "Supervised learning for arrival time estimations in restaurant meal delivery". In: *Transportation Science* 56.4, pp. 1058–1084.

Instacart (2022). *Introducing OrderUp, a new way to save time and money on Instacart with orders from two retailers– with just one delivery fee*. Last accessed on Nov 01, 2023. URL: https://www.instacart.com/company/updates/introducing-orderup-a-new-way-to-save-time-and-money-on-instacart-with-orders-from-two-retailers-with-just-one-delivery-fee/.

Jacobs, K et al. (2019). *The last-mile delivery challenge*. Tech. rep. Capgemini Research Institute, Paris.

Li, Zhuoxin and Gang Wang (2024). "On-Demand Delivery Platforms and Restaurant Sales". In: *Management Science* Forthcoming. URL: https://doi.org/10.1287/mnsc.2021.01010.

Liu, Sheng, Long He, and Zuo-Jun Max Shen (2021). "On-time last-mile delivery: Order assignment with travel-time predictors". In: *Management Science* 67.7, pp. 4095–4119.

Mao, Wenzheng et al. (2022). "On-demand meal delivery platforms: Operational level data and research opportunities". In: *Manufacturing & Service Operations Management* 24.5, pp. 2535–2542.

Moreno, Carlos et al. (2021). "Introducing the "15-Minute City": Sustainability, resilience and place identity in future post-pandemic cities". In: *Smart Cities* 4.1, pp. 93–111.

Pickme (2024). *Let's build tomorrow's delivery together! (In French)*. Last accessed on Feb 11, 2023. URL: https://www.mypickme.com/.

Raghavan, S and Rui Zhang (2024). “The driver-aide problem: Coordinated logistics for last-mile delivery”. In: *Manufacturing & Service Operations Management* 26.1, pp. 291–311.

Savelsbergh, Martin and Tom Van Woensel (2016). “50th anniversary invited article—city logistics: Challenges and opportunities”. In: *Transportation Science* 50.2, pp. 579–590.

Statista (2024). *Market insights into e-commerce (Worldwide)*. Last accessed on Dec 5, 2024. URL: https://www.statista.com/outlook/emo/ecommerce/worldwide?currency=usd.