Operational Strategies for On-demand Delivery Services

Operational Strategies for On-demand Delivery Services

Operationele strategieën voor on-demand bezorgservices

Thesis

to obtain the degree of Doctor from
Erasmus University Rotterdam
by the command of
rector magnificus

Prof.dr. R.C.M.E. Engels

and in accordance with the decision of the Doctoral Board

The public defense shall be held on
Thursday 21 November 2019 at 13:30 hours

by

Alp Muzaffer Arslan

born in Istanbul, Turkey

Erasmus University Rotterdam

Doctoral Committee

Promoter: Prof. dr. R. Zuidwijk

Co-promoter: Dr. N.A.H. Agatz

Other members:

Prof. dr. M.B.M de Koster

Prof. dr. Alan Erera

Prof. dr. R. Dekker

Prof. dr. Kees Jan Roodbergen

Dr. R. Spliet

Dr. Mathias Klapp

Erasmus Research Institute of Management – ERIM
The joint research institute of the Rotterdam School of Management (RSM) and the Erasmus School of Economics (ESE) at the Erasmus University Rotterdam
Internet: `http://www.erim.eur.nl`

ERIM Electronic Series Portal: `http://repub.eur.nl/pub/`

ERIM PhD Series in Research in Management, 481
ERIM reference number:EPS-2019-481-LIS
ISBN 978-90-5892-569-5

Design: PanArt en advies, `www.panart.nl`

This publication (cover and interior) is printed by Tuijtel on recycled paper, BalanceSilk®. The ink used is produced from renewable resources and alcohol free fountain solution. Certifications for the paper and the printing production process: Recycle, EU Ecolabel, FSC®, ISO14001. More info: `www.tuijtel.com`

Acknowledgments

Finalizing my Ph.D. trajectory has always seemed a distant future to me; therefore arriving this stage triggers rather complicated feelings. In contrast to generic consensus of pursuing a Ph.D. degree containing a lot of struggles, and as this experience often criticized by many of us, I should admit that I enjoyed most of it. I will remember this period of my life with pleasant memories. For this reason, in this part of the dissertation, I would like to express my gratitude to everyone who directly or indirectly contributed to it.

It is almost an academic sin not to be grateful to your supervisors in the first two paragraphs in the acknowledgments. This is another unwritten rule in the secret academic society among thousands. Niels, thank you for not only being a great and patient supervisor but also being a good friend. Rob, I am always grateful for your trust on me. I have enjoyed working with you. Lastly, I would like to mention Leo, even though we lost him in the early part on my Ph.D, path; I always remember his wisdom.

Mathias, thank you for hosting me in Santiago, contributing to this dissertation by coauthoring one of the chapters, and being a committee member. I am also very grateful for all other committee members; namely Prof. dr. Alan Erera, Prof. dr. Rene de Koster, Dr. Remy Spliet, Prof. dr. Kees Jan Roodbergen, and Prof. dr. Rommert Dekker.

My appreciation goes to everybody who is and has been creating such a friendly and hierarchy-free organization at Department TOM or informally T9. I will always consider this place as my academic home and I look forward to coming back here for any reason. Many thanks go to Carmen, Cherly, and Lianne for their endless patience and help. Also, I am very much in debt to many trainers at the Erasmus

Sport Center for their wonderful services and for keeping me and many others mentally and physically healthy.

Surely, I have met many good friends along the Ph.D. studies. Jun, Erik, Xishu, Ainara, Christina, Katharina, Alberto, Johann, Aniruth, Joshua, Jelmer, Kevin, Thomas, Cansu. I consider myself a lucky person to have encountered such unique and wonderful individuals. I am also blessed with having friends from all over the world with their constant supports. Menevis, Kutalmis, Ovunc, Dila, Nurettin, Ozcan, Firat thank you so much.

Well, Waka-Waka people or Gucci Gang, I just love you all. I am not sure anything would be this good without you. Special thanks go to Francesco for not just being awesome but also designing the cover page of the thesis, to Joydeep for constantly challenging my seventh sense and driving skills, to Arpan for his intellectual remarks and brilliant culinary talents, and to Kaveh being source of limitless kindness and coolness. Also our ghost member Jenny, I did not forget you either.

If you pursue your academic career abroad, I suppose your family sacrifices the most. My family is not an exception. There are not enough words to express how much I am grateful to my dad and mum for their incomparable care and affection, and for being perfect role-models. I love both of you. Lastly, I feel truly fortunate to have two dearest people; Ilona and Alper, in my life for their unconditional support and love.

Alp Arslan
Singapore, 2019

Contents

Acknowledgments **e**

1 Introduction **1**
1.1 Motivation . . . 1
1.2 On-demand Delivery in Online Shopping . . . 3
1.3 Utilizing Existing Traffic Flows . . . 5
1.4 Utilizing Existing Infrastructure . . . 6
1.5 Research Objectives and Methodology . . . 8
1.5.1 Research Objectives . . . 8
1.5.2 Research Methodology . . . 10
1.6 Outline of the Thesis . . . 10

2 Crowdsourced Delivery: A Dynamic Pickup and Delivery Problem **15**
2.1 Introduction . . . 15
2.2 Related literature . . . 18
2.3 Problem description . . . 20
2.4 Solution approach . . . 24
2.4.1 Rolling horizon approach . . . 24
2.4.2 Offline problem formulation . . . 25
2.5 Solving the routing subproblem . . . 27
2.5.1 Theoretical insights . . . 27
2.5.2 Exact recursive algorithm . . . 28
2.5.3 Heuristic recursive algorithm . . . 31
2.6 Instance generation . . . 32
2.7 Computational results . . . 33
2.7.1 Benchmark . . . 33

2.7.2 Base analysis . . . 36
2.7.3 Impact of departure time flexibility and delivery lead-time . 38
2.7.4 Impact of commitment strategies . . . 40
2.7.5 Impact of the problem size on the solution time . . . 42
2.8 Conclusions . . . 43
2.9 Appendix . . . 45
2.9.1 MIP formulation TSP-TWPC . . . 45
2.9.2 Matching formulation hindsight benchmark . . . 47
2.9.3 Determining the number of backup vehicles . . . 47
2.9.4 Proof of Observation 3 . . . 49

3 On-demand crowdshipping with store transfers 53
3.1 Introduction . . . 53
3.2 Literature Review . . . 55
3.3 Problem Description . . . 57
3.4 Solution Approach . . . 60
3.4.1 Offline matching problem . . . 60
3.4.2 Rolling horizon framework . . . 63
3.5 A Computational Study . . . 65
3.5.1 Instance generation . . . 65
3.5.2 Base results . . . 67
3.5.3 Impact of time flexibility, lead time and crowd drivers ratio 69
3.5.4 Impact of dynamics . . . 70
3.5.5 Benefits of bundling . . . 71
3.5.6 Detour lengths . . . 72
3.6 Concluding Remarks . . . 73

4 Splitting shopping and delivery tasks in an on-demand service 75
4.1 Introduction . . . 75
4.2 Literature Review . . . 78
4.3 Problem Description . . . 79
4.3.1 Problem Inputs, Notation, and Decisions . . . 79
4.3.2 Problem Formulation . . . 82

4.4 Solution Approach 83
4.4.1 Rolling Horizon Framework 83
4.4.2 Deterministic Pickup and Split Delivery Problem with Deadlines 84
4.5 Computational Study 88
4.5.1 Experimental Setup 88
4.5.2 PsDPd Heuristic Validation 89
4.5.3 Operating Policies 90
4.5.4 Base Case Results 91
4.5.5 Impact of the Number of Stores per Request 92
4.5.6 Impact of Shopping Economies of Scale α 95
4.5.7 Impact of the Delivery Deadline on the Packing Benefits . . 96
4.5.8 Routing Benefits 97
4.6 Concluding Remarks 99
4.7 Appendix 100
4.7.1 Arc Costs in the Task-Based Graph 100
4.7.2 Distributions in Base Case for Policy $C\&S$ 100

5 Conclusions and future outlook 103
5.1 Main results 103
5.2 Future outlook 106

Bibliography 109

About the author 117

Summary 119

Samenvatting (Summary in Dutch) 123

ERIM Ph.D. Series Research in Management 127

1 Introduction

1.1 Motivation

Internet-enabled advancements shape our daily-life routines. One of these habits is shopping. More and more people alter their purchasing preferences from brick and mortar stores to online shopping. Online shopping provides a broad range of products to entice customers which the conventional retail in a traditional store cannot easily offer. Online customers are also able to browse and order their required goods at the time and place most convenient to them. Furthermore, the purchased products are delivered to customers' preferred locations.

Online shopping refers to the purchase of goods or services by consumers via the internet. The term *e-commerce* is often used interchangeably with online shopping. Both of the terms typically cover any online transactions between organisations and people. However, this thesis solely focuses on services which include customers buying physical goods (Business-to-consumer, or B2C) via the internet and the corresponding fulfillment operations (such as receiving, processing and delivering orders). Therefore, in this thesis we use the term of online shopping only in this manner.

Online shopping is convenient to customers, but it has one important drawback: lack of instant gratification. Online customers must often wait until the next day to receive the ordered goods. As Skrovan (2017) has shown, the notion of immediate possession plays a crucial role in consumers' shopping preference; particularly when they decide on where to do the shopping. Subsequently, decreasing the delivery lead-times may help customers to switch to online shopping (Balasubramanian, 1998). Therefore, numerous internet retailers, start-ups and logistic service providers offer faster delivery services.

On-demand delivery is a new service for online shopping that allows customers to receive ordered goods within a few hours. In this service, a provider conveys a customer request to its relevant departments. Subsequently, the delivery vehicles, which can be e.g. a truck or a bike, pick up the ordered goods from affiliated stocking locations and deliver them to the customer's address before the deadline. In the supply chain management terminology, this leg of the chain is known as the last-mile.

The last-mile delivery is considered as the most inefficient and costly activity in the whole supply chain (Gevaers et al., 2011), because small-sized parcels have to be shipped to geographically dispersed locations. This issue causes the widely known lack-of-scale problem. In this problem, the service provider faces a trade-off between sending a delivery vehicle with a partial load to serve nearby customers and sending a fully loaded vehicle to serve more customers that are located far from each other. In either case, the service provider is not able to utilize the vehicle in the best possible way.

The on-demand delivery further intensifies the lack-of-scale problem due to the restrictive delivery lead-times. Delivery vehicles, in this service, are required to visit the stocking locations or depots multiple times to collect arriving customer orders while they are on delivery. Returning to depots more often within the service period, however, increases the marginal milage per order, and, consequently more resources are required to serve the same amount of customers as compared to *e.g,* a next-day delivery service. As a result, the delivery cost per order in an on-demand delivery service is often higher than in the more traditional next-day delivery. Therefore, offering expedited delivery is costly for providers.

Service providers, however, are unable to fully transfer the cost of the on-demand delivery to online shoppers due to the consumers' resistance to delivery fees. Buldeo Rai et al. (2019) show that consumers are highly price-sensitive towards delivery fees, and they are often unwilling to pay for faster deliveries. Therefore, service providers must absorb most of the costs to stay competitive in the market. At a consequence, they should explore cost-effective and innovative methods and strategies to reduce the operational cost of the on-demand delivery services. These methods and the potential benefits of these methods are the focus of this dissertation.

This chapter is structured as follows: In the next section, we identify challenges for on-demand delivery services in the business-to-consumer market and elaborate on our research motivation. In Section 1.3, we introduce crowdsourced delivery as an innovative and cost-effective method that aims to utilize the existing traffic flows for on-demand parcel delivery systems. In Section 1.4, we describe online shopping and delivery platforms as an alternative to warehouse-based e-commerce supply chains. In Section Section 1.5, we present an overview of the dissertation and briefly describe the methodology used. Finally, we outline the dissertation in Section 1.6.

1.2 On-demand Delivery in Online Shopping

The on-demand economy refers to digital marketplaces that offer immediate access to goods and services (Kerrigan, 2016). On-demand delivery in online shopping is a service that delivers goods ordered via the internet within a short time such as one or two hours. This service is expected to continue its growth particularly due to its popularity among young generations (Colby & Bell, 2016). The primary motive of the on-demand service delivery providers is to satisfy the increasing demand of instant gratification of consumers and consequently to enlarge their market shares. Consider the following examples from renowned internet retailers that recently started to offer on-demand delivery services:

- Amazon PrimeNOW offers free delivery within two hours for members in over 50 cities in the USA. The service allows customers to order from a wide range of products from local stores or restaurants (Amazon, 2018).
- Instacart is an on-demand grocery delivery service operating in North America, which is considered the most promising company in 2016 according to Forbes (Solomon, 2016). They provide delivery services as short as an hour.
- Bol.com, a Dutch online retailer, has delivery service within two hours for the city of Amsterdam for the select range of products (Bol.com, 2019).

The benefits of on-demand home delivery services mostly come from the convenience that they offer to customers. Field studies support that this fast delivery services

create its own demand such as Martin et al. (2016) show that up to 20 % of the online shoppers may willing to pay a premium fee for instant deliveries. Furthermore, fast shipments may decrease the delivery failures due to no-show at the drop-off time.

While benefits of offering on-demand home delivery services are many for e-tailers, exploiting them requires a comprehensive examination of their logistics operations. From the perspective of service providers, an on-demand delivery service influences two-core fulfillment processes: (i) *order processing* at the stocking location including order picking, sorting, and packaging; and (ii) *order distribution* from the stocking location to customer's address including dispatching and delivery of the orders (Klapp, 2016). The order processing covers all operations to prepare an order to be ready for delivery. The order distribution consists of decisions of forming vehicle routes and determining their dispatching times from the stocking locations.

In the presence of short delivery-lead times, however, these logistic operations face certain challenges. First of all, the order processing needs to be fast and responsive to dynamic order arrivals. Second, for efficient order distribution, providers require sophisticated decision support mechanisms to overcome dispatching and routing inefficiencies. Third, providers need to keep products available in the vicinity of their service regions. Last, they need to have the capacity of real-time monitoring the availability of products.

While the first and last challenges are somewhat related to the technological capabilities of providers, inefficiency in order distribution is primarily an issue of the lack of scaling. No surprise, on-demand delivery services are mostly offered in urban areas where they are supported by the sufficient customer density. The high population densities in cities offer geographically concentrated demand, which helps to find dense networks where the service may become profitable faster than rural areas (Savelsbergh & Van Woensel, 2016). Nevertheless, service providers require to rethink two fundamental questions in order to create competitive on-demand home delivery systems:

1. From what locations to serve customer demand?
2. How to efficiently organize the last-mile to customers' addresses?

In this dissertation, we will focus on examining innovative concepts to answer these questions above. In particular, we aim to utilize existing traffic flows and retail

infrastructures by studying *crowdsourced delivery* and *online shopping and delivery platforms*. In the next sections, we introduce these two concepts, and we show how they help the last-mile delivery providers to redesign cost efficient on-demand home delivery systems.

1.3 Utilizing Existing Traffic Flows

In recent years, the emergence of innovative paradigms in the context of collaborative consumption creates innovative alternatives for traditional businesses. Taking advantage of the networked world, sharing-based innovations connect people with specific needs to people with resources, to fulfill these needs on a platform where they can interact with each other. Therefore, in this dissertation, we explore the idea of utilizing the existing traffic flows to make on-demand parcel deliveries in urban areas in the context of crowdsourcing.

Crowdsourcing refers to the outsourcing of organisation's activities to often unknown group of people mostly via internet-based channels (Howe, 2006); therefore, crowdsourced delivery is defined as delegating delivery operations to non-professional people who have willingness, time, and resources to carry some parcels. These people are known as crowd or ad-hoc drivers, since they are not formal delivery employees (Punel et al., 2018). Crowdshipping potentially reduces the operational costs and also shortens delivery times.

Due to the novelty of the crowdshipping paradigm, various types of business models appear in practice. Rougès & Montreuil (2014) showed that most of the crowdshipping models are predominately used in Business to Customer (B2C) settings. A typical crowdshipping model has three primary stakeholders: senders, typically those who sell products and choose the delivery provider; crowd drivers; and a platform that facilitates the communications between drivers and senders. The crowdshipping platforms usually carry out operational activities such as pricing, matching of senders and drivers, and providing technologies such as tracking of parcels or smart payment methods, to increase the convenience and security of delivery experiences (Punel & Stathopoulos, 2017). While getting shipping service with lower fee motivates senders to participate in crowdshipping platforms, Miller

et al. (2017) show that crowd drivers seek various benefits ranging from pure monetary compensations to environmental and societal returns from this experience.

In this dissertation, we consider on-demand crowdshipping models in urban areas in which crowd drivers have already existing itineraries. We specifically focus on crowdshipping platforms that utilize crowd drivers' prespecified traffic flows to fulfill dynamically arriving delivery requests that must be served within a few hours. In particular, we propose operational models, *e.g.* dynamic matching and routing decisions, for the platforms that manage crowd drivers and professional delivery fleets in hybrid ways. We also explore the benefits of crowd drivers' integration into on-demand delivery services.

1.4 Utilizing Existing Infrastructure

In various ways, urban areas provide unexplored resources for logistics operations. Next to traffic flows that can be crowd-sourced for ad-hoc drivers, existing infrastructures such as shopping malls and retail stores, provide alternative stock or transfer locations for on-demand service providers. In this thesis, we envision two delivery models that incorporate the existing urban retailer infrastructures for effective last-mile delivery systems.

First, we focus on a system in which retailer stores are not only locations where crowd drivers originate their trips and collect parcels to deliver but also they act as transhipment points. With the possibility of parcel transfers between crowd drivers at stores, we aim to organize the parcel deliveries effectively and decrease crowd drivers' deviation from their original trips. Using stores has a substantial advantage in this system as two crowd drivers do not have to be at the same point at the same time to make a parcel exchange.

Second, we consider a new business model that uses retailer stores as fulfilment centres. In this business model, a platform provides an online shopping interface for local retailer stores such that customers can browse stores' assortments online and place their orders. Furthermore, these platforms organize their delivery resources to collect the order from corresponding stores and deliver them to the customer's

address. For convenience, we name these platforms or the business models as "shopping and delivery platforms."

Shopping and delivery platforms' supply chain model is fundamentally different from the classical online shopping, in which an order is fulfilled from the sellers' or the e-tailers' warehouses, typically by a third party logistics partner. Online shopping and delivery platforms, however, sell affiliated merchants goods and organize the deliveries of these goods that should be collected from these stores. One of the advantages of this model is that a platform does not require to invest in costly infrastructures such as warehouses, picking systems, *etc.* Furthermore, using stores in service region as stocking places may help last-mile planning due to the spatial proximity to the goods.

This fundamental difference creates a need for new decision support systems to manage this type of supply chain. One essential feature of this chain is that the stocking place of an order is not fixed, and it depends on the order. An online customer chooses the store, which is also the stocking location of the ordered goods, when s/he orders via such a platform. Therefore, delivery employees of the platform should collect the goods from the specified store. This particular structure of order fulfillment and distribution makes the underlying optimization problem a variant of the dynamic pickup and delivery problem, instead of the vehicle routing problem.

Also, in this model, order picking, sorting, and packing are done by delivery employees. When a store does not involve in order processing, delivery employees should go into stores and pick the corresponding goods from shelves and purchase them on behalf of customers. In this case, shopping times in stores should explicitly be taken into account while planning delivery units operations.

Another challenge of this model is when a customer order contains goods from multiple stores. In this case, delivery employees must visit each store and collect the goods before delivery, and this structure adds another level of complexity in logistic operational operations.

The last but not the least challenge confronted by the online shopping and delivery platforms offering on-demand delivery services, is urgency for shipments. Due to short delivery lead-times, these platforms have certain similarities to meal delivery platforms, in which a delivery unit picks an already prepared package from the

associated restaurant and deliveries it within a short time. However, unlike meal delivery services, these platforms can assign multiple orders from different stores to a delivery unit at the same time. Consequently, the operational decisions for shopping and delivery platforms are more complicated for these platforms than for meal delivery platforms.

In this dissertation, we particularly study the use of 'request split' in the logistic models of online shopping and delivery platforms. With request splits, customer orders consisting of multiple store pickups can be served by different delivery employees, and this option allows platforms to manage delivery resources efficiently and fulfill more parcels.

1.5 Research Objectives and Methodology

1.5.1 Research Objectives

This dissertation studies two innovations in on-demand delivery services in online shopping. In particular, we organize our research around concepts that utilize exisiting traffic flows and retailer infrastructures to create efficient on-demand home delivery services. We contribute decision-support tools to facilitate the associated design and operations. Even though there is a large stream of literature on last-mile logistics operations and a series of decision-support mechanisms for these operations, academic studies that consider these concepts together are relatively limited. Given the short history of growing demand for express deliveries and non-professional crowd integration into the physical tasks, there is a strong need for quantitative models to support the accompanying logistics decisions. As a result, the main objectives of this research can be summarized as follows:

- To analyze the characteristics and challenges of on-demand delivery services, particularly in urban areas.
- To identify operational decisions that appear within crowdsourcing and online shopping and delivery platforms.

- To develop new quantitative models and efficient algorithms to support on-demand delivery providers' decisions that utilize existing traffic flows and/or retail infrastructures.
- Lastly, to quantify the potential benefits of using existing traffic flows and retail infrastructures in on-demand delivery, and to improve managerial decision making by numerical experiments.

Figure 1.1: Dissertation Overview

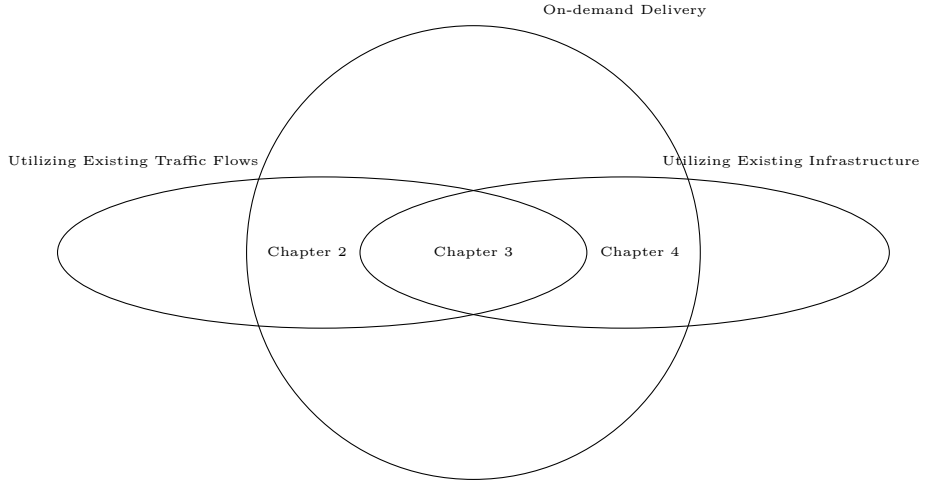


Figure 1.1 gives an overview of this dissertation by specifying the focus of each chapter. Overall, in all chapters we analyze various on-demand delivery problems appearing in the supply chain for the online shopping. In chapter 2, we investigate the potential of exploiting existing traffic flows in the on-demand home delivery services with the help of crowdsource integration into traditional delivery fleets. In chapter 3, we look into the intersection of these three concepts and we aim to utilize retailer stores as transhipment points where crowd drivers exchange parcels. In Chapter 4, we explore a request split operation for online shopping and delivery platforms offering on-demand delivery services.

1.5.2 Research Methodology

In this thesis, we investigate three different types of on-demand home delivery services; and for each one, we try to capture the relevant aspects of last-mile logistics into quantitative models in the field of Operations Research. Furthermore, we introduce the characteristics of crowdsourcing and online shopping and delivery platforms within our models that enable us to analyze the crucial features of these two concepts further.

Theoretically, the underlying optimization problem of each chapter can be modelled as a variant of dynamic pickup and delivery problem. As a result, we adopt several standard procedures that often use to formulate and solve these problems, such as Integer and Dynamic Programming techniques. Furthermore, we propose some heuristics to be able to analyze large instances.

To test the performance of our algorithms and quantify the potential benefits of discussed delivery models, we generate instances that represent real-life cases. We also assess specific features of these systems under various settings.

1.6 Outline of the Thesis

This dissertation consists of 5 chapters and is organized as follows: **Chapter 1** presents the motivation for the research. Finally,**Chapter 5** concludes the thesis and gives directions for future research.

Chapter 2: Crowdsourced delivery: A Dynamic Pickup and Delivery Problems with Ad-hoc Drivers [1]

This chapter studies the concept of crowdsourced delivery that aims to use excess capacity on journeys that already take place. We consider a service platform that automatically creates matches between parcel delivery tasks and ad-hoc drivers.

[1] Arslan, Alp M., et al. "Crowdsourced delivery: A dynamic pickup and delivery problem with ad hoc drivers." Transportation Science 53.1 (2019): 222-235

The platform also operates a fleet of dedicated vehicles to serve the tasks that cannot be served by the ad-hoc drivers. The matching of tasks, drivers, and dedicated vehicles in real-time gives rise to a new variant of the dynamic pick-up and delivery problem. We propose a rolling horizon framework and develop an exact solution approach to solve the matching problem each time new information becomes available. To investigate the potential benefit of crowdsourced delivery, we conduct a wide range of computational experiments.

In this chapter, we will answer the following research questions;

1. Which crowd driver's attributes influence the performance of on-demand crowdshipping models?
2. What specific operational model would be the most beneficial for the delivery systems that non-professional crowd drivers and traditional employees combined?
3. How different performance measures for a crowdshipping system depend on factors such as density and delivery lead-times?

Chapter 3: On-demand Crowdshipping with Store Transfers

In this chapter, we consider a system similar to Chapter 2, in which crowd drivers can make deliveries on their way home from the store. However, apart from Chapter 2, in this new system a crowd driver either completes the delivery or transfers the parcel to a store after which another driver makes the final delivery. To examine the benefits of such store transfers, we present an optimization approach to match delivery tasks and crowd drivers in real-time. Our numerical experiments show that store transfers reduce the system-wide cost, and decrease the inconvenience of the participating drivers by reducing the required detours.

In this chapter, we will answer the following research questions;

1. What are the benefits of allowing parcel transfers in on-demand crowdshipping?
2. How to maximize the benefits of store transfer in dynamic environment ?

3. Which particular settings that store transfers are the most beneficial?

Chapter 4: Splitting Shopping and Delivery tasks in On-demand Personal Shopper Service[2]

In this chapter, we introduce an online personal shopper service, a type of last-mile delivery system operating as an intermediary between online customers and brick and mortar stores. This service receives online customer requests, each potentially having shopping requirements from multiple stores, and arranges the deliveries of these requests to customers. We study the benefits of splitting customer orders into smaller delivery tasks served by different shoppers in parallel and clustering tasks from multiple orders sharing a common pickup location. We develop an online optimization algorithm to solve a personal shopping problem integrating request splitting, task to shopper assignment, and routing problems. In a case study, we suggest that one can increase the number of customers served, and decrease the average service time per request, by splitting customer requests into smaller tasks and efficiently consolidating these into shoppers.

In this chapter, we formulate mathematical optimization models and their solution algorithms that support online shopping and delivery platforms' operations. Particularly, we establish the following research questions:

1. What are the main characteristics of the fulfillment model for online shopping and delivery platforms?
2. How to formulate different operational strategies such as request split for these platforms?

Research statement

This Ph.D. dissertation has been written during the author's work at Erasmus University Rotterdam. The author is responsible for formulating research questions, building models, analyzing results and writing all the chapters of this thesis. While carrying out the research, the author received feedback from the doctoral advisors

[2] Arslan, Alp, Niels Agatz, and Mathias Klapp. "Splitting shopping and delivery tasks in an on-demand personal shopper service." ERIM Report Series (2019)

and the other members of the doctoral committee which subsequently increased the quality of research.

2 Crowdsourced Delivery: A Dynamic Pickup and Delivery Problem

This chapter has been published in Transportation Science (Arslan et al., 2019a)
Co-authors: Niels Agatz, Leo Kroon, and Rob Zuidwijk

2.1 Introduction

Despite the spectacular growth of online sales, internet retailers and logistic service providers still face many logistical challenges in the successful fulfilment of goods ordered online. One of the main difficulties is to provide cost-efficient home delivery services. The recent trend towards shorter delivery lead-times and same-day delivery further increases the strain on transport efficiency. At the same time, mobile internet technology gives rise to new opportunities to organize the last-mile. One of those new opportunities is *crowdsourced* delivery. This concept entails the use of excess capacity of private passenger vehicles on journeys that already take place to support delivery operations. By using existing traffic flows, this could potentially enable faster and cheaper deliveries. Moreover, it may help to reduce the negative environmental impact of the use of dedicated delivery vehicles, such as emissions. This development is part of a bigger trend that is called the "sharing economy" which allows people to enhance the use of resources through the redistribution, sharing and reuse of excess capacity in goods and services.

In 2013, Walmart investigated the use of in-store customers to deliver goods to its online customers on their way home from the store. In the same year, DHL ran a

pilot in Stockholm called 'MyWays', using ordinary people to perform some of their deliveries (Morphy, 2014). In a similar vein, Amazon recently launched a service called Amazon Flex in Seattle that supports the use of self-employed drivers to make deliveries.

In recent years, we have seen the advent of online service platforms and mobile smartphone apps to quickly connect delivery tasks (goods that need to be shipped) and drivers willing to make a delivery along their route. The drivers pick up parcels from a retail store, warehouse or dedicated pickup location, and deliver them to customer locations on their way home or to work. Some of these platforms, such as Friendshippr and Roadie, focus on long distance shipping, while others offer (on-demand) local delivery services such as Kanga, Renren Kuaidi, Deliv, Trunkrs and Amazon flex. Table 2.1 provides an overview of different operators.

Table 2.1: Examples of crowd-delivery platforms that offer same-day delivery services (June 2017)

Name	Compensation scheme	Information from ad-hoc drivers	Where
Deliv	Hourly rate	Time period	18 U.S. cities
Renren Kuaidi	Per package	Time period	16 Chinese cities
Trunkrs	Per package	Time, origin and destination	The Netherlands
Kanga	Hourly rate	Time-period	1 U.S. city
Amazon flex	Hourly rate	Time-period	30 U.S. cities

Instead of traditional employees or service providers, the drivers act voluntarily on their own initiative. They are willing to make deliveries along their route to help others, support environmentally friendly deliveries, and potentially earn some extra money. In particular, drivers are willing to take a parcel along a specific journey that they are already making. This is different from systems in which the drivers only perform deliveries to earn money. In this setting, drivers may vary greatly with respect to their time and detour flexibility. Some drivers may only want to make a small detour to take a parcel on a trip that they were already making, others may be willing to make multiple deliveries. When each driver can be matched with at most one delivery task, we can model the problem as a bipartite matching problem (Agatz et al., 2011). However, if we want to allow multiple pickups and drop-offs in a single trip, we also need to consider the route sequence, which makes the problem more challenging.

To ensure that all parcels are delivered in time, a peer to peer (P2P) delivery platform may use a third-party service to deliver the tasks for which no ad-hoc driver could be found, e.g. Dutch startup PickThisUp uses this model. Moreover, to ensure the reliability and trustworthiness of the ad-hoc drivers, it could use various feedback mechanisms and external regulations (see Einav et al. (2016) for an overview of P2P trust generating mechanisms).

In this paper, we focus on a delivery service platform that automatically matches delivery tasks and ad-hoc drivers to facilitate on-demand delivery. The platform also operates a set of dedicated back-up vehicles to serve tasks for which the use of an ad-hoc driver is not feasible or not efficient. As such, the crowdsourcing provider needs to assign delivery tasks to ad-hoc drivers and dedicated vehicles and determine the associated delivery routes. We consider a same-day delivery setting in which both tasks and drivers dynamically arrive over time. This type of service is most relevant for groceries, electronics and pharmaceutical products (Martin et al. (2016)).

The main contributions of this paper are as follows. Firstly, we introduce and describe a new route planning problem that involves the use of ad-hoc drivers and dedicated vehicles to perform on-demand deliveries. We present a rolling horizon framework and develop an exact solution approach (based on a matching formulation) to repeatedly solve the various versions of the off-line problem. Secondly, we conduct an extensive computational study to investigate under what circumstances it is viable to use crowdsourced transportation to enable on-demand deliveries. To quantify the benefits, we compare the performance of a crowdsourced system with a traditional dedicated delivery system. The results indicate that the use of ad-hoc drivers can significantly reduce transportation costs.

The remainder of the paper is organized as follows: we discuss the relevant literature in the next section. In Section 2.3, we formally describe the problem. In Section 2.4, we explain the implementation of our rolling horizon framework and formulate the problem. In Section 2.5, we provide a solution approach for the routing subproblem. In Section 2.7, we describe our instances and present the results from our numerical experiments. Finally, in Section 2.8, we provide some concluding remarks and directions for future research.

2.2 Related literature

Thus far, most research in the area of crowd-sourcing has focused on *virtual* tasks that can be done remotely over the internet such as text editing, translation and debugging (see e.g. Doan et al. (2011)). A recently emerging area of research considers the use of the crowd drivers to conduct ridesharing services and parcel delivery (Suh et al., 2012; Sadilek et al., 2013; Rougès & Montreuil, 2014).

From an operations perspective, one of the key features of a crowd-based service platform is that the workers are self-scheduling as they decide when and how often to work. This means the platform can only indirectly control the supply of workers. One way for the platform operator to control supply and demand is to dynamically adjust wages and prices. Several papers have studied such dynamic pricing strategies (i.e. surge pricing) to coordinate passenger demand and the supply of drivers in ridesharing platforms like Uber and Lyft (Chen, 2016; Cachon et al., 2017; Tang et al., 2016).

In this paper, we address the lack of control on the supply of drivers by considering the use of regular dedicated vehicles. We contribute to this stream of literature by specifically considering a hybrid platform that allows the use of a traditional fleet capacity to serve certain tasks.

At its core, the crowdsourced delivery problem is a pickup and delivery problem (PDP) that aims to transport goods from origins to destinations at minimum costs. This links our problem to the huge body of literature on PDPs, see Berbeglia et al. (2007) for an overview. Since we consider an on-demand service, our problem is also related to the literature on the dynamic pickup and delivery problems (Berbeglia et al., 2010). Our problem is also closely related to the recent work in the context of the same-day delivery of goods ordered online from a single depot (see Klapp et al. (2018b) and Voccia et al. (2017)). In this stream of research, developing strategies for finding the optimal timing for the vehicle departures and optimal assignment of parcels between the vehicles are the main challenges (Savelsbergh & Van Woensel, 2016).

Unlike the traditional PDP setting, we only use a dedicated fleet of vehicles as an option for the independent ad-hoc drivers. In that sense, our problem is similar to

ride-sharing or carpooling where individual travelers share a ride to save on their travel costs by using their own vehicles (Furuhata et al., 2013a; Agatz et al., 2012a). A recent study by Agatz et al. (2011) investigates the viability of dynamic ride-sharing in which trips are announced shortly before departure. The authors create single rider, single driver ride-share matches and propose a rolling horizon approach for dealing with real-time updates. The study shows that the success of a ride-sharing system depends on a sufficiently large number of participants. To guarantee a certain service level to the riders, the ride-share service provider could use (a small number of) dedicated drivers to serve riders that would otherwise remain unmatched. Lee & Savelsbergh (2015) investigate how many of such dedicated drivers are needed to achieve a certain service level. They formulate the problem as an integer program and present a heuristic approach to solve realistic-size instances. In a similar vein, Stiglic et al. (2015) explore the benefits of using meeting points to improve the performance of a ride-sharing system. When riders are willing to walk to and from a meeting point, this may allow drivers to carry multiple riders without the inconvenience of many additional stops. Baldacci et al. (2004) describe a static car-pooling problem that aims to assign a set of drivers to riders. Similar to our paper, drivers can do multiple pickups along their routes. In contrast to our problem, they assume a simplified routing structure in which all riders and drivers have the same destination, i.e. the workplace, and consider a static problem setting in which all requests are known in advance. They use maximum ride time restrictions to ensure the convenience of the passengers. The authors formulate the problem as a set-partitioning problem and propose an exact solution method based on column generation.

Several recent papers study the use of existing traffic flows to enable freight transportation. Li et al. (2014) and Li et al. (2016) consider a setting in which taxis transport parcels along with their passengers. Both papers propose heuristic solution strategies to insert parcel requests into existing taxi routes. Depending on the flexibility of the passengers, a taxi may stop multiple times to pick up or drop-off a parcel. In a similar vein, Ghilas et al. (2013) and Masson et al. (2014) explore the potential of using public transportation in freight transportation. The authors introduce a PDP that aims to synchronize delivery vehicles with the scheduled city buses. In line with these studies, Fatnassi et al. (2015) investigate the possible

integration of passenger and freight transportation in the context of the automated transport systems for city logistics.

Most similar to our work is the work of Archetti et al. (2016) that analyzes a setting in which occasional drivers complement a traditional delivery service. Similar to our study, the authors aim to minimize the sum of the amount paid to the ad-hoc drivers and the routing cost of the dedicated vehicles. In contrast to our work, they consider a static problem setting without time windows in which the occasional drivers are allowed to make only a single delivery. Archetti et al. present a heuristic solution approach that combines variable neighborhood search and tabu search.

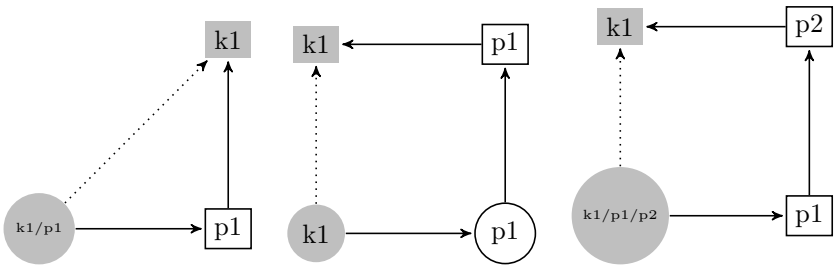
2.3 Problem description

We consider an online crowdsourcing platform that continuously receives new delivery tasks and driver trip announcements over time. Let N denote the set of all origins and destinations, d_{lm} the travel distance, and t_{lm} the travel time between locations $l, m \in N$.

Let P be the set of (parcel) delivery task announcements. Delivery task $p \in P$ has a pickup location o_p, which can be a retail store, warehouse or dedicated pickup point, and a drop-off location d_p, which is usually the home of the online buyer. The task has an earliest pickup time e_p when it is ready to be picked up and a latest arrival time l_p that corresponds to the time that it has to be delivered. Without loss of generality, we consider a setting in which the parcel needs to be delivered within a certain delivery lead-time L_p, e.g. within two hours, where $L_p = l_p - e_p$, $L_p \geq t_{o_p,d_p}$. This is similar to the same-day delivery service model that is used by companies UberRUSH and Shutl. For a given deadline, we can calculate the implied latest departure time $\bar{l_p}$ by $l_p - t_{o_p,d_p}$.

Let K be the set of driver announcements. The driver's trip announcement $k \in K$ specifies his origin o_k and destination d_k. A driver $k \in K$ has an earliest departure time e_k and a latest arrival time l_k. The driver also specifies a maximum travel time, T_k, where $t_{o_k,d_k} \leq T_k \leq l_k - e_k$ and a departure time flexibility, denoted by $F_k = l_k - e_k - t_{o_k,d_k}$. Note that the maximum travel time implicitly also defines the maximum detour flexibility, denoted by $T_k - t_{o_k,d_k}$.

Figure 2.1: A driver(grey) and tasks(white) travelling from his origin (circle) to destination (square)



(a) 1-stop **(b)** 2-stops(1) **(c)** 2-stops(2)

→ crowdsourced trip, ⇢ original trip

Besides the detour and departure time flexibility, drivers may also want to specify the maximum number of additional stops that they are willing to make. Let $Q_k \in \mathbb{Z}^+$ denote the *stop willingness* of driver k. The stop willingness restricts the number of different locations that is visited by the driver and therefore reflects the level of inconvenience the ad-hoc driver is willing to accept. We believe that picking up multiple tasks at one and the same location is more convenient than picking them up at different locations.

Figure 3.1 presents an example of routes that involve one, two or three stops. Figure 3.1a denotes a setting in which the driver's origin coincides with the pickup location of the task so that he or she only needs one additional stop to make the delivery. This corresponds to Walmart's idea to let store customers deliver packages to online buyers along their route from the store to home. Figure 3.1b shows an example in which the driver's origin is different from the pickup location of the task. In this case, the driver needs to make two additional stops, i.e. one pickup and one drop-off. Another example that requires two additional stops is depicted in Figure 3.1c where the driver picks up two parcels at his origin and then makes two drop-offs.

To simplify notation, we assume that the time and stop restrictions are more restrictive than the capacity restrictions. This seems like a reasonable assumption as most consumer goods are small enough to easily fit in the trunk of a car. That

is, 86 percent of Amazon's packages are under five pounds and small enough to be shipped even by a drone (Popper, 2015). To accommodate a setting in which we transport larger objects such as furniture or white goods, we could easily introduce an additional constraint on the volume.

We define a *job* j as a set of one or more tasks. The set J denotes the collection of all jobs that are part of at least one feasible match. A match (k, j) between driver k and job j is feasible if there exists a *feasible* route r_{kj} by driver k to serve job j in which the driver starts from his origin o_k, covers all tasks in j and ends at his destination d_k. A route r is *feasible* if it satisfies the following constraints.

- *Stop constraint.* The number of unique locations visited in route r_{kj} is less than or equal to $Q_k + 2$ (including the origin and destination of the driver).
- *Driving time constraint.* The total travel time of r_{kj} is less than or equal to T_k.
- *Time schedule constraints.* Driver k does not depart before its earliest departure time e_k or arrive after its latest arrival time l_k. Each task $p \in P$ is not picked up before its earliest pickup time e_p or arrive after its latest arrival time l_p.
- *Precedence constraints.* For each task $p \in P$, a driver picks the parcel up before dropping it off. This implies that the difference between the drop-off time and the pickup time of task $p \in P$ is greater than or equals to t_{o_p,d_p}

Let R_{kj} be the set of all feasible routes for driver k and job j and R be the set that consists of all feasible routes.

Let B be the set of dedicated vehicles. Each vehicle $b \in B$ has a capacity Q^c, starts and ends all dispatches from the same depot and has an earliest departure time e_b and a latest arrival time l_b back at the depot. A match (b, j) between vehicle b and job j is feasible if there exists a feasible route r in which the vehicle starts from the depot, covers all tasks in j and ends at the depot. Routes for the vehicles are feasible if they satisfy the time schedule and precedence constraints. Table 2.2 summarizes the main notation that is used in this paper.

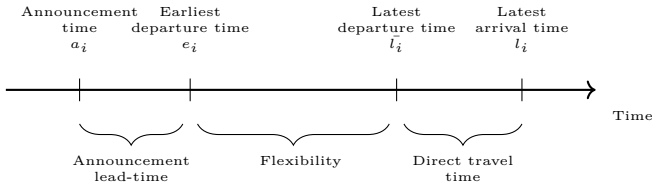
The system is characterized by the continuous arrival of drivers and tasks. Each driver $k \in K$ is announced to the delivery platform at time $a_k \leq e_k$ and each delivery

Table 2.2: Notation used in this paper

N	Set of all locations, index i
P	Set of parcel tasks, index p
K	Set of ad-hoc drivers, index k
B	Set of dedicated vehicles, index b
J	Set of all jobs, i.e. combinations of tasks, index j
L_p	Delivery lead-time of task p
T_k	Maximum travel time of driver k
Q_k	Maximum number of additional stops along the route that driver k is willing to make
F_k	Departure time flexibility of driver k
Q^c	Capacity of a dedicated vehicle

task $p \in P$ arrives at time a_p. We call the time between the announcement time a_k and the earliest departure time e_k the announcement lead-time, $A_k = e_k - a_k$. For tasks, the announcement lead-time represents the time to prepare a task for pickup. The general time line of a delivery task and a driver can be found in Figure 2.2.

Figure 2.2: Annoucement i timeline; $i = p$ for delivery tasks and $i = k$ for ad-hoc drivers



The objective of the crowdsourced delivery problem is to determine the assignments of delivery tasks to which ad-hoc drivers or dedicated vehicles such that the system-wide total cost is minimized. To model this problem, we make the following assumptions:

- All delivery requests need to be served by an ad-hoc driver or a dedicated vehicle, i.e, we do not allow the rejection of delivery tasks.
- If the costs of serving a task is the same for an ad-hoc driver and a dedicated vehicle, we prefer to use the ad-hoc driver.
- A dedicated vehicle has to return to the depot before starting another route. That is, we do not allow en-route diversions.

2.4 Solution approach

Since both delivery tasks and drivers arrive dynamically throughout the day, we use an event-based rolling horizon framework that repeatedly solves the problem of matching tasks to drivers each time t that a new task or driver arrives. We describe our rolling horizon approach in Section Section 2.4.1 and the offline matching problem in Section 2.4.2.

2.4.1 Rolling horizon approach

At each iteration q of the rolling horizon approach, we determine the matches based on all information that is available to the system at that point in time. In particular, we run the optimization for all active, i.e. known and still available, tasks and drivers.

At time t, task p is *active* if it is not part of a match committed before time t, arrived before t ($a_p \leq t$) and has not expired yet ($\bar{l}_p \geq t$). This is similar for driver k. The job j is *active* if all tasks in j are active and there is at least an active driver k or dedicated vehicle which has a feasible route for job j. The drivers and tasks that are associated with a match that is committed at time t are not included in any of the optimization runs after t. Each dedicated vehicle $b \in B$ is available in each optimization run with an earliest departure time from the depot e_b that depends on earlier job assignments.

Each optimization run results in a number of tentative matches between jobs, ad-hoc drivers and dedicated vehicles. In principle, we choose to commit these tentative matches as late as possible. However, we also analyse variants where commitments are made early. The late commitments mean that we do not commit to a tentative match before its latest departure time. The latest departure time of a certain tentative match (k, j) is the latest time that driver k can start driving to serve all tasks in j within their time schedules and then reach his or her destination on time. This is similar for the dedicated vehicles.

Next, we describe the offline problem that we solve in each optimization run at event arrival times t within our rolling horizon framework, based on all available information at time t.

2.4.2 Offline problem formulation

As in Stiglic et al. (2015), we can model this problem as a matching problem with side-constraints. Let $D = K \cup B$ denote the set of all drivers, i.e. ad-hoc drivers and dedicated drivers. We create a node for each driver $d \in D$, and a node for each job $j \in J$. An arc between node d and node j represents a feasible match between driver d and job j. The weight of the arc denotes the routing costs of serving job j by driver d, which may be the costs of the optimal route or the costs of any feasible route.

Let A be the set of all feasible arcs. Let $J_d, \ d \in D$ denote the collection of jobs that driver d can serve, and $J_p, \ p \in P$ denote the set of jobs that contains task p. Let x_{dj} be the binary decision variable that indicates whether the arc between driver d and job j is in the solution ($x_{dj} = 1$) or not ($x_{dj} = 0$). The coefficient c_{dj} represents the weight of the arc (k, j), which denotes the cost if driver d is assigned the job j. Then, the problem that aims to minimize the total cost can be formulated as follows:

$$\min \sum_{(d,j)\in A} c_{dj} x_{dj} \tag{2.1}$$

$$\text{s.t} \sum_{j\in J_d} x_{dj} \leq 1 \ \ \forall d \in D, \tag{2.2}$$

$$\sum_{j\in J_p} \sum_{d\in D} x_{dj} = 1 \ \ \forall p \in P, \tag{2.3}$$

$$x_{dj} \in \{0, 1\} \ \ \forall (d, j) \in A. \tag{2.4}$$

Equation (2.1) is the objective function that aims to minimize the sum of the costs of ad-hoc driver matches and the dedicated vehicle matches. Constraints (2.2) make sure that each driver is assigned to at most one job. Constraints (2.3) make sure that each task is assigned to one of the drivers or a dedicated vehicle.

When a job j contains only a single task, there exists only one route, which is the origin of the driver and the task followed by the destination of the task and the

driver. However, for jobs containing multiple tasks, there might be more than one feasible route. Thus, the determination of the optimal route for corresponding jobs is a subproblem of our matching formulation. The solution approach for this subproblem is the topic of Section 2.5.

Note that for a new optimization run, we only have to create new jobs J_q and their corresponding decision variables (x_{kj}) for the task or driver that arrived after the previous iteration $q-1$. For all currently active drivers and tasks that were already active in run $q-1$, we only need to check whether the jobs found previously are still time feasible. That is, jobs that were not in a tentative match may have expired, e.g because the driver's latest departure time when serving a particular job occurs before optimization run q. Conceptually, this is similar to the approach that was presented in Chen & Xu (2006).

Figure 2.3: An illustration of the rolling horizon framework

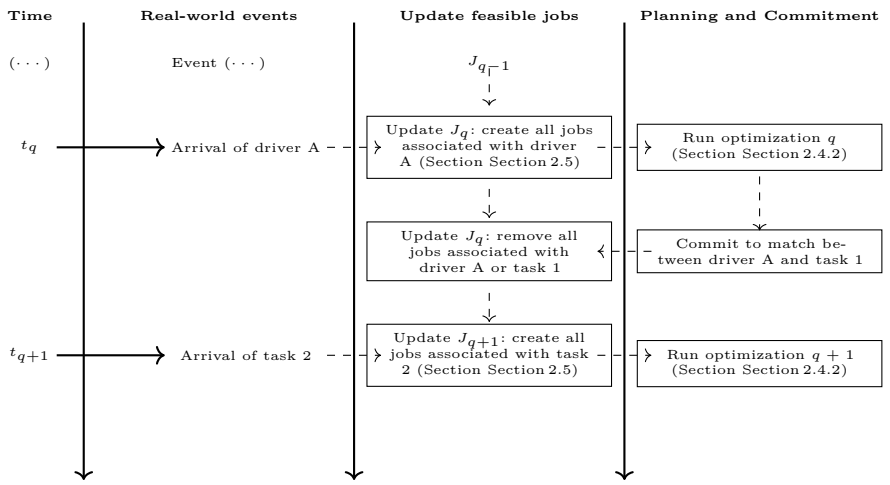


Figure 2.3 illustrates the interaction between the real-world events and the planning and execution. A real-world event, i.e, the arrival of a driver or task, triggers a new optimization run. However, before we can run the optimization model, we have to first create all relevant new jobs that are associated with the new driver or task arrival. The solution of this matching problem provides a new set of tentative matches that replaces the previous set. When a match is committed, we

immediately delete all associated jobs from the memory. Moreover, before each optimization run, we delete the jobs that are no longer time feasible.

2.5 Solving the routing subproblem

While our matching problem can be solved quickly solved with an Integer Programming solver, it may be quite time consuming to find all feasible jobs (x_{kj} variables). In the worst case, when each driver could serve all tasks in a single trip, the number of feasible matches for p tasks and k drivers is $O(k2^p)$. Additionally, assessing the feasibility of serving a specific set of tasks, we need to determine the sequence in which to serve them. This means that it involves solving the Traveling Salesman with Time Windows and Precedence Constraints (TSP-TWPC) as a subproblem, see Mingozzi et al. (1997). A description of TSP-TWPC can be found in the appendix of this paper. Savelsbergh (1985) showed that finding a feasible solution for the TSP-TW is NP-complete. However, in our problem setting, due to the time and stop restrictions, the number of tasks per job is relatively small, which implies that the number of feasible routes is likely to be far less in practice.

2.5.1 Theoretical insights

In this section, we will present some theoretical observations that will help us to efficiently find all feasible jobs and associated routes.

Observation 1: A job $j \in J$ does not have a feasible route if there is a subset $j' \subset j$ that has no feasible route (Stiglic et al., 2015).

This observation implies that any *feasible* job for a specific driver is a union of smaller feasible jobs. A match between one driver and two tasks is only feasible if both tasks are individually feasible with this driver. A match between one driver and three tasks is only feasible if all task pairs are also feasible and so forth. Another implication of this observation is that all unions that include two incompetible tasks are infeasible. We use these two properties to reduce the number of jobs to be considered in our recursive algorithm.

Observation 2: A route r is not feasible if one of the sub-routes $r' \subset r$ is not feasible. A sub-route can be obtained by removing one or more tasks from the original route.

For each feasible job with w tasks, we store all feasible routes in our recursive algorithm. For each driver, we use the feasible routes for w tasks to construct the feasible routes with $w + 1$ tasks by iteratively inserting a task, i.e. a pickup and a drop-off, in the route. According to Observation 2, we do not have to consider the route sequences that we found to be infeasible with w tasks.

Observation 3: Any feasible pickup and delivery route r can be transformed to a feasible *clustered route* r', where $\text{dist}(r') \leq \text{dist}(r)$. A *clustered route* is a pick-up and delivery route that does *not* revisit the same pickup location while still carrying tasks that originate from that location.

This implies that there exists an optimal *clustered route*, which means that we only have to consider clustered routes when recursively building the routes in order to find the shortest route. The proof of this observation can be found in the appendix.

Observation 4: When picking up several tasks at the same location one after the other, then the pickup sequence has no impact on the routing length.

This means that we can reduce the search space for feasible routes by applying some simple symmetry breaking rules in our recursive algorithm.

2.5.2 Exact recursive algorithm

Based on these observations, all feasible job to driver assignments can be determined by using the recursive algorithm as presented in Algorithm 1. This algorithm starts with determining the jobs with just a single task, and subsequently combines these single tasks to make jobs of two tasks, three tasks and so on. Let J_k^w be the set of jobs with w tasks that are feasible for driver k and let R_k be the set of feasible routes associated with driver k.

Each pair of driver k and job j has a set of feasible routes associated with it that we store in set R_{kj}. For a single driver and a single task there is only one feasible

Algorithm 1 Recursive algorithm

Input: The list J^1 of all feasible pairs of a driver and a single-task job and the set $R_{k,j}$ of associated routes to serve job j with driver k.
Output: All feasible driver-job matches

1: **for all** $k \in K$ **do**
2: w $\leftarrow 2$
3: **for all** $j \in J_k^{w-1}$ **do**
4: **for all** $\{p\} \in J_k^1 \wedge\ p \notin j$ **do**
5: $R_{k,j\cup p} \leftarrow \emptyset$
6: **if** `SUBFEAS`$((j,p,k))$ **then**
7: $R_{k,j\cup p} \leftarrow$ `FINDROUTES`$((R_{k,j},p))$
8: **end if**
9: **if** $R_{k,j\cup p} \neq \emptyset$ **then**
10: $R_k \leftarrow R_k \cup R_{k,j\cup p}$
11: **else**
12: the job $(j \cup p)$ is infeasible
13: **end if**
14: **end for**
15: **end for**
16: **if** $w < Q_k \wedge J_k^w \neq \emptyset$ **then**
17: $w \leftarrow w + 1$
18: **else**
19: J_k and R_k are determined. Go to a new driver.
20: **end if**
21: **end for**

route. All single-task jobs that can be served by driver k are clustered in the set J_k^1 and all routes that are associated with J_k^1 are added to the set R_k.

Next, the algorithm generates feasible jobs and routes for each driver sequentially. To examine whether a job with size w is feasible for driver k, we check each combination of a job with size $w-1$ and a single task that is not already included in this job by calling the `SUBFEAS` subroutine as described in Algorithm 2. Based on **Observation 1**, `SUBFEAS` checks if all subsets of job j are feasible as a necessary condition for the feasibility of job j. A job is feasible if at least one feasible route exists for driver k.

Algorithm 3, called `FINDROUTES` determines all feasible routes by inserting the origin and the destination of the task at all feasibility positions of all feasible routes of the job. Based on **Observation 3** and **Observation 4**, the number of possible insertions can be reduced significantly without sacrificing optimality. At line 7 in Algorithm 3, the **viableinsert** function checks whether or not the insertion generates redundant routes based on **Observation 3** and **Observation 4**. This additional check speeds up the system in two ways. Firstly, we save time by not checking unnecessary insertions. Secondly, we reduce the number of feasible routes per jobs which speeds up the evaluation in the subsequent steps. If `FINDROUTES` returns a non-empty set, this means the pair of the input job and task creates a feasible job for driver k. Else, the examined job is infeasible.

Algorithm 2 SUBFEAS.

Input: Job j, task p and driver k
Output: False if at least one subset of j and p is infeasible for driver k, true otherwise

1: **function** SUBFEAS(j, p, k)
2: $z \leftarrow$ `true`
3: **for all** $j \in J_k^{|j|}$ **do**
4: j' $\leftarrow \{(j \setminus \{q\}) \cup \{p\}\}, \ \forall q \in j$
5: **if** $j' \notin J_k^{|j|}$ **then**
6: $z \leftarrow$ `false`, **return** z
7: **end if**
8: **end for**
9: **return** z
10: **end function**

Algorithm 3 FINDROUTES.

Input: Job j, task p and driver k
Output: All feasible routes of job j and task p for driver k

```
1: function FINDROUTES(R_{k,j}, p)
2:     z ← ∅
3:     p_1 ← o_p
4:     p_2 ← d_p
5:     for all r_{k,j} ∈ R_{k,j} do
6:         for i ← 0 to |r_{k,j}| − 1 do
7:             if viableinsert(i) then
8:                 if feasible(i, p_1) then
9:                     for l ← i + 1 to |r_{k,j}| do
10:                        if feasible(l, p_2) then
11:                            z ← z ∪ [r_{j,k} ∪ (p_1, p_2)]
12:                        end if
13:                    end for
14:                end if
15:            end if
16:        end for
17:    end for
18:    return z
19: end function
```

2.5.3 Heuristic recursive algorithm

Since the number of feasible jobs and associated routes grows exponentially with the number of tasks, our exact approach will not be able to solve large instances. Therefore, we consider the following heuristic speedups within our recursive method to limit the number of feasible routes and jobs that is considered in each step.

- *Eliminate non-promising jobs:* Algorithm 1 recursively builds jobs of size $w+1$ by inserting one task into all jobs of size w. Instead of using all jobs of size w as a seed, this procedure only expands the jobs with the lowest routing costs and enough time slack to facilitate additional tasks.
- *Trip time limit:* By limiting the maximum duration of a trip for the dedicated vehicle, we reduce the number of possible feasible routes.
- *Eliminate non-promising routes:* Instead of keeping all feasible routes of each job, we can store only a limited number of the shortest routes per job (only

a subset of z, line 11 in Alg. 3) of a task j. See Malandraki & Dial (1996) for a similar implementation.

2.6 Instance generation

We generate several instances that represent different task and driver characteristics within a square region with a size of 15 km and a depot for the dedicated vehicles at the center, i.e. at [7.5,7.5]. In particular, we use three different instance types, which we refer to as geographies, to generate the origins and destinations of the tasks and ad-hoc drivers.

g1 The first geography, *one-to-many*, is inspired by Walmart and considers a setting in which all tasks and ad-hoc drivers start from one single origin (i.e. store), located in the center of the region, while all destinations are uniformly spread over the region. Note that this type of setting is considered in most other studies in same day delivery area, such as Klapp et al. (2018b), and Voccia et al. (2017).

g2 The second geography, *few-to-many*, has five different origin locations, one in the center and four randomly selected from the service area. Here, each task and ad-hoc driver originates from one of the five locations, where each location is chosen with equal likelihood.

g3 In the third geography, *many-to-many*, origins and destinations of both tasks and ad-hoc drivers are uniformly distributed over the service area.

The announcement times of tasks (a_p) and ad-hoc drivers (a_k) are drawn from a uniform distribution spanning a ten hours service period. We assume that each ad-hoc driver has the same *system-wide* departure time flexibility of 20 minutes and a stop willingness of two stops. The announcement lead-time A is 15 minutes for all tasks and ad-hoc drivers. We use euclidian distances and assume a constant speed of 50km (31 miles) per hour.

The characteristics of the base case instances are summarized in Table Table 3.1.

Table 2.3: Characteristics of base case instances

Definition	Values
# tasks	100
# ad-hoc drivers	100
Delivery lead-time (L)	90 min
Stop willingness of ad-hoc drivers (Q)	2
Announcement lead-time (A)	15 min
Departure time flexibility (F)	20 min
Vehicle speed	50 km/h
Dedicated vehicle capacity (Q^c)	10

2.7 Computational results

In this section, we evaluate the performance of our solution approaches and assess the viability of the crowdsourced delivery platform in the different settings described in Section Section 2.6. Section Section 2.7.1 presents the results of our heuristic approach to determine a hindsight lower bound, and section Section 2.7.2 presents the base case results. Sections Section 2.7.3 and Section 2.7.4 assess the impact of different lead times, commitment strategies on the performance of the system. In Section Section 2.7.5, we analyze the solution times of our exact approach for different instance sizes.

All experiments were implemented in C++ and conducted on a 2,7 GHz Intel Core i5 and 8 GB 1867 MHz DDR3 of installed RAM. Gurobi 6.51 was used as an IP solver see Gurobi Optimiziation (2016)).

2.7.1 Benchmark

To evaluate the performance of our rolling horizon strategies, we report a *hindsight* benchmark solution that serves as a theoretical lower bound on the solution quality. In the hindsight problem, the dedicated vehicles may perform multiple trips from the depot during the service period. This means we can characterize the underlying problem as a multi-trip vehicle routing problem (MTVRP) which is known to be difficult to solve to optimality (Cattaruzza et al., 2016). To solve this problem

within our matching framework, we relax the fleet size restrictions to determine a lower bound on the solution of the original problem (Archetti et al., 2015). Relaxing the fleet size restrictions allows us to model the dedicated vehicles as ad-hoc drivers, i.e. we assume that at any given time we can dispatch a dedicated vehicle from the depot. The output of the relaxed problem is a set of trips and dispatch time windows for the dedicated vehicles. To find the minimum number of required vehicles we would need to perform these trips, we solve a scheduling problem (for details see Appendix 8.3).

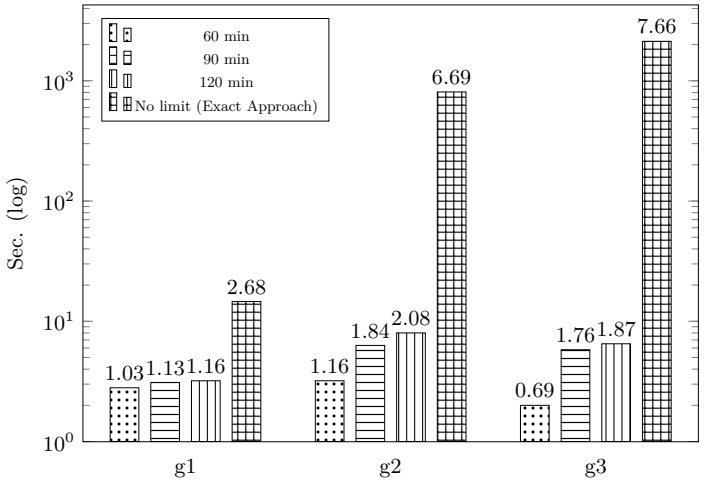
Table 2.4: Heuristic validation for different tour limits, lead-time = 90 min, 50 tasks: 50 drivers, averaged over 5 instances

Tour limit	60 min		90 min		120 min	
% OPT gap	avg	max	avg	max	avg	max
$g1$: *one-to-many*						
AHDR-2	0.0	0.1	0.0	0.1	0.0	0.1
AHDR-4	0.0	0.0	0.0	0.0	0.0	0.0
DEDR	0.9	3.4	0.4	0.9	0.4	0.9
$g2$: *few-to-many*						
AHDR-2	3.9	7.1	1.9	3.9	1.9	3.5
AHDR-4	2.8	6.1	1.7	4.8	1.6	4.4
DEDR	8.1	11.6	2.8	4.4	2.6	3.8
$g3$: *many-to-many*						
AHDR-2	8.1	10.1	2.5	3.7	1.6	3.6
AHDR-4	6.6	10.0	2.6	3.9	1.9	3.8
DEDR	12.2	14.9	3.2	4.6	2.2	4.1

Moreover, we also implemented the heuristic speed-ups as discussed in Section Section 2.5.3. Particularly, we only keep the five shortest routes per job, and we keep the $(Q^c)^2$ most promising job in this heuristic validation. Table Table 2.4 presents the optimality gaps for three different tour length limits for the dedicated vehicles. AHDR-2 and AHDR-4 represent the crowd-sourced delivery problems when the ad-hoc drivers are willing to stop 2 and 4 at most, respectively and DEDR represent the problem if only dedicated vehicles make the deliveries.

We see that the heuristics perform well for the crowd-sourced delivery problem instances AHDR-2 and AHDR-4. As expected, the quality deteriorates with a stricter tour length limit. The quality of the solutions deteriorates significantly if the tour length limit is reduced from 90 minutes to 60 minutes. That is, for a

Figure 2.4: Solution times for the hindsight benchmark, lead-time = 90 min, 50 tasks: 50 drivers, averaged over 5 instances



tour length limit equal to the delivery lead-time of 90 minutes, we see an average optimality gap of up to 2.6 percent. The reason for this is that our heuristics only affect the routes of the dedicated vehicles and that the number of delivery tasks that is served by the dedicated vehicles is relatively small.

However, the results show that even in the most challenging case, in which all tasks are served by the dedicated vehicles (DEDR), the results are reasonable when we apply a tour length limit of up to 90 minutes. In particular, we see an average gap of up to 3.2 percent.

Figure Figure 2.4 shows the computation times for the heuristics with different three tour length limitations (60, 90, 120 mins) and the exact approach. Overall, we see that the solution time is highest in the $g3$ instances in which each delivery task also has a different origin. It is clear from the figure is that the heuristics significantly reduce the computation times, especially for the more difficult *few-to-many* and *many-to-many* instances.

2.7.2 Base analysis

For the base analysis, we present the results for both the dynamic setting and the hindsight benchmark. In Table Table 2.5, we compare the solutions for the three instance types (g1 (one-to-many), g2 (few-to-many), g3 (many-to-many)) for a stop willingness Q of 2 and 4: AHDR-2 and AHDR-4. As an additional benchmark, we also present the solution that we would obtain without ad-hoc drivers (DEDR), in which all tasks are served by the dedicated vehicles. The cost of the DEDR solution is chosen as a baseline for the cost benchmark and its cost is normalized to 100.

We evaluate the solutions in the various experiments by the following statistics:

- *Total cost*: the total compensation paid to the matched drivers and the cost of the dedicated vehicle trips.
- *Tasks matched*: the fraction of tasks that are served by an ad-hoc driver; the complement represents the percentage of tasks that are served by the dedicated services.
- *Drivers matched*: the number of ad-hoc drivers that are assigned to a job; some of the ad-hoc drivers that offered their vehicles will not be used.
- *# dedicated vehicles*: the number of dedicated vehicles that is required to serve all tasks.

From the results, we see that there are clear benefits from the use of ad-hoc drivers. Table Table 2.5 shows a reduction in costs from the use of ad-hoc drivers of between 18.8 and 37 percent as compared to the DEDR solution in the dynamic setting. We see the highest number of task matches and associated cost savings in the first geography. This is intuitive because in the first geography the drivers do not have to make detours to accommodate a pickup, since drivers and tasks all start at the same location. On the other end of the spectrum, the third geography has the lowest matching rate and savings.

Recall that in our experiments the transportation costs are proportional to the system-wide vehicle miles. This implies that the cost reductions correspond to distance reductions. Note that the reduction of the total system-wide vehicle miles

from the use of ad-hoc drivers may provide environmental benefits such as reduced emissions and congestion. This, however, is only the case if the ad-hoc drivers produce less or equal emissions per mile compared to dedicated drivers.

We also observe that the cost-efficiency of the system increases with the stop willingness of the drivers. This again relates to an increase in the number of tasks that are served by the ad-hoc drivers. The number of tasks matched increases for all instance types if the drivers' stop willingness increases from two to four. Interestingly, we see that in g1 and g3 we need fewer drivers when each individual driver can make more stops. This suggests that by combining the delivery of multiple tasks, we need fewer drivers to do the same amount of work.

Table 2.5: Base analysis, lead-time = 90 min, 100 tasks: 100 drivers, averaged over 5 instances

	Dynamic				Hindsight			
	Total Costs	Tasks matched (%)	Drivers matched (%)	# dedicated vehicles	Costs	Tasks matched (%)	Drivers matched (%)	# dedicated vehicles
g1: one-to-many								
AHDR-2	67.9	66.2	43.4	2.8	54.5	77	47.5	1.0
AHDR-4	63.0	77.0	40.2	2.6	50.6	84.4	51.2	1.0
DEDR	100	0.0	0.0	3.4	94	0.0	0.0	2.0
g2: few-to-many								
AHDR-2	77.5	42.2	36.8	4.8	62.2	49.2	40.4	2.4
AHDR-4	73.9	54.4	37.4	4.8	69.0	58.6	38	2.2
DEDR	100	0.0	0.0	6.4	90.1	0.0	0.0	4.6
g3: many-to-many								
AHDR-2	81.2	41.8	41.8	7.0	71.4	50.6	50.6	5.2
AHDR-4	78.8	53.6	40	6.8	66.7	64.2	43.4	5.0
DEDR	100	0.0	0.0	8.2	94.2	0.0	0.0	6.4

Table Table 2.5 also shows the required number of dedicated vehicles in the different settings. We can see that the number of dedicated vehicles decreases with the stop willingness of the ad-hoc drivers. This is intuitive because when the ad-hoc drivers can serve more tasks we need less dedicated vehicles. Furthermore, we observe that the smallest number of dedicated vehicles is used in geography 1 and the highest number in geography 3. The reason for this is that with less origins it is easier to match ad-hoc drivers and tasks, which again means we need less dedicated vehicles.

If we compare our rolling horizon solutions to the theoretical hindsight benchmark, we see a gap between 6 and 24.5 percent. This gap is larger for the settings that include ad-hoc drivers than for the settings that do not. This makes sense as the settings with ad-hoc drivers involve more uncertainty and can therefore benefit

more when all information is available. That is, while the dedicated vehicles are employed by the company, the ad-hoc drivers are private independent entities whose arrivals over time are difficult to predict. This also explains why more tasks are served by the ad-hoc drivers in the hindsight solutions than in the rolling horizon solution.

Figure Figure 2.5 shows the average distances driven by the ad-hoc drivers and the dedicated vehicles per task in the online and the hindsight setting. In both figures, the distances are normalized with the associated (online or hindsight) results of DEDR. The results show that the distance traveled per task is a lot less for the ad-hoc drivers. The dedicated vehicles loose some efficiency when ad-hoc drivers are present in the online setting; however in the hindsight setting the dedicated vehicles also get benefit from the presence of ad-hoc drivers. This result can be explain by the lack of consolidation in the online setting.

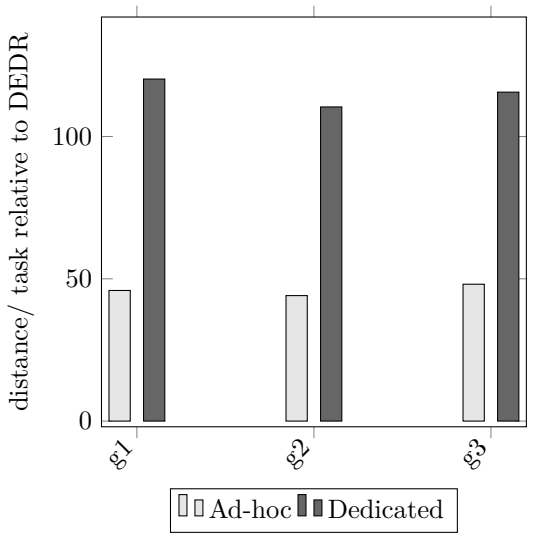
2.7.3 Impact of departure time flexibility and delivery lead-time

In this section, we examine the effect of the drivers' departure time flexibility and the delivery lead-times on the performance of the system. Table Table 2.6 presents the results for a departure time flexibility of 10, 20, or 30 minutes, and a delivery lead-time of 60, 90, or 120 minutes. Similar to the previous results, the costs are normalized to 100 for the base case. As expected, the costs decrease for higher departure time flexibilities and delivery lead-times. In a similar vein, the number of matched tasks increases with the departure time flexibility and with the delivery lead-time.

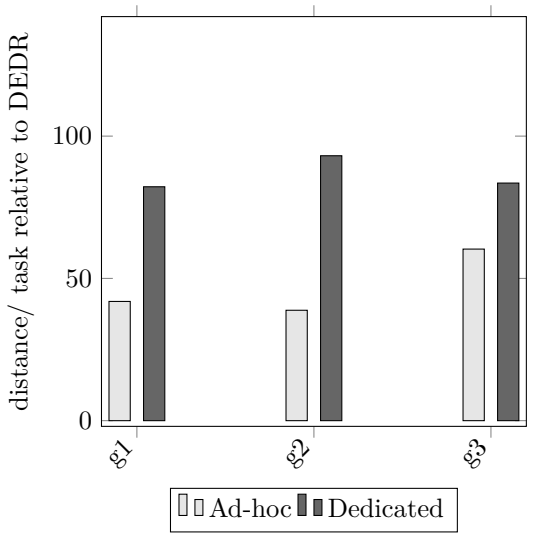
Table 2.6: The impact of delivery lead-time (L) and departure time flexibility (F), g2: few-to-many, 100 tasks: 100 drivers, averaged over 5 instances

	Costs			Tasks matched (%)		
	F=10 min	F=20 min	F=30 min	F=10 min	F=20 min	F=30 min
$L = 60$ min	147.6	134	120.3	20	33.5	46.9
$L = 90$ min	112.9	100	86.3	28.9	42.2	56.7
$L = 120$ min	97.9	97.8	81.9	31.6	47.7	61.1

Figure 2.5: Average distances per task; lead-time = 90 min, stop willingness = 2, 100 tasks: 100 drivers, averaged over 5 instances



(a) Online



(b) Hindsight

2.7.4 Impact of commitment strategies

Up to now, we used a strategy that commits to tentative matches as late as possible. In this section, we study the impact of the so-called *earliest-commit* strategy, in which we immediately commit to a tentative match as soon as we find it. While this strategy minimizes the waiting time for the drivers and the lead-time for the tasks, it may not be the best strategy in terms of the total system performance. Moreover, we also consider two hybrid strategies in which we commit early to the matches that involve an ad-hoc driver but as late as possible to matches that involve a dedicated vehicle and vice versa.

Table Table 2.7 shows the results of these experiments. For the reader's convenience, we normalize the costs of the default latest commit strategy to 100. As expected, the results show that the latest-latest commitment strategy outperforms all other strategies in terms of the total costs and number of tasks matched. On the other end of the spectrum, the earliest-earliest commitment strategy performs worst. Intuitively, we see that it is worse to immediately commit to a dedicated vehicle match than to a match with an ad-hoc driver.

Table 2.7: Comparison of commitment strategies, g2: few-to-many, lead-time = 90 min, stop willingness = 2, 100 tasks: 100 drivers, averaged over 5 instances

Commit ad-hoc, dedicated	Costs	Tasks matched (%)	Drivers matched	# dedicated vehicles
Late, Late	100.0	42.2	36.8	4.8
Early, Late	101.2	37.8	34.4	5.2
Late, Early	121.6	28.8	27.2	4.8
Early, Early	122.6	26.2	25.2	5.2

Latest-Latest performs best as it waits as long as possible before committing to tentative matches. This also means that the problems that are solved in the optimization runs in our rolling horizon are likely to be larger for this commitment strategy than for the other strategies. Figure Figure 2.6 provides a box plot of the number of active tasks per optimization run for different commitment strategies and instance types. The results indeed show that the latest-latest strategy has the highest numbers of tasks per optimization run and the earliest-earliest the lowest numbers of tasks.

Interestingly, we see that geography 1 is associated with the largest number of tasks per optimization run, both on average and in terms of the maximum problem size. The main reason for this is that in geography 1 all tasks and drivers originate from one and the same location. This means that an ad-hoc driver or dedicated vehicles does not need to drive to the pickup location to pickup a task before it can start service. As a consequence, there is more time to wait before having to commit to a tentative match.

Figure 2.6: Number of tasks per optimization run for different instance types and commitment strategies, lead-time = 90 min, stop willingness = 2, 100 tasks: 100 drivers

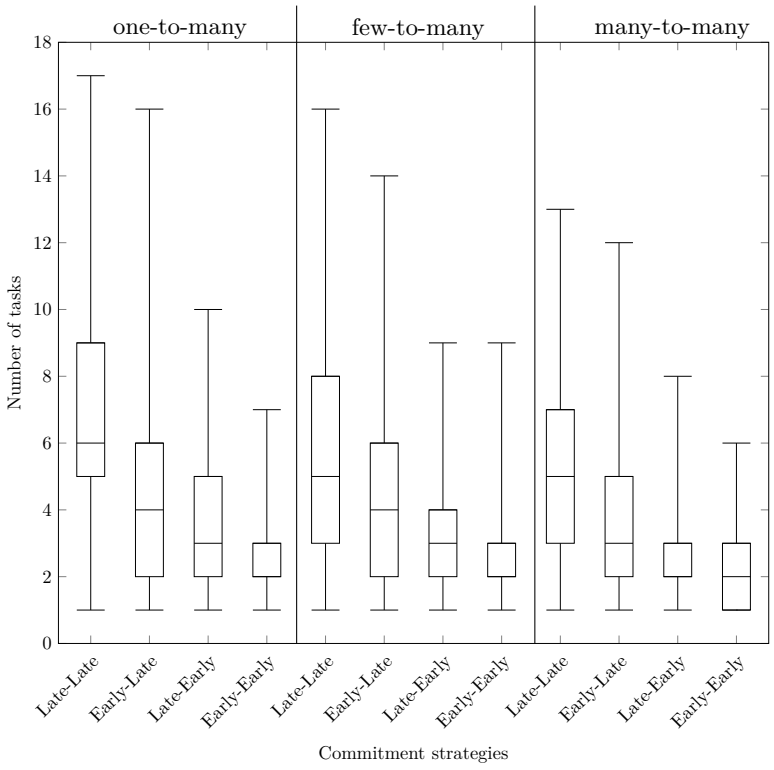
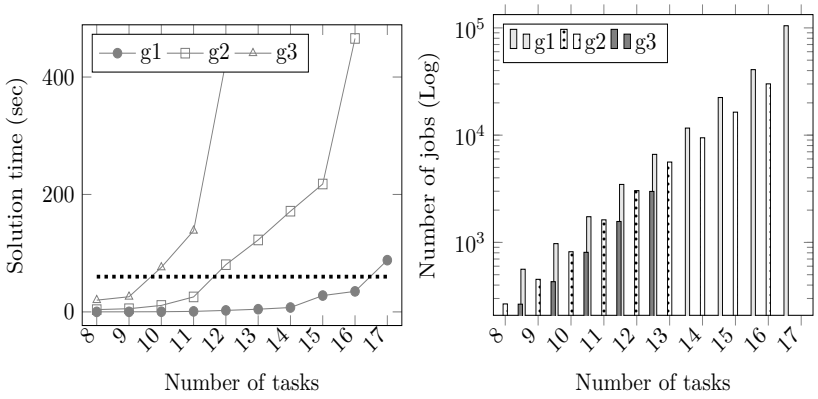


Figure 2.7: Time performance, lead-time = 90 min, stop willingness = 2



(a) Solution time in seconds **(b)** Number of jobs

2.7.5 Impact of the problem size on the solution time

To provide some more insight into the maximum problem sizes that we can solve to optimality within reasonable time, Figure Figure 2.7 presents the solution times for different problem sizes in the rolling horizon. In particular, Figure Figure 2.7a presents the average solution times per iteration for different numbers of active tasks for each geography. Figure Figure 2.7b presents the average number of jobs associated with these different optimization runs. The dashed line in Figure Figure 2.7a denotes a solution time of 60 seconds which we believe is the maximum time available for each optimization run in a dynamic application.

Overall, the graphs show that the solution times and the number of jobs increase with the number of active tasks for all instances. However, we also see that there are clear differences between the different geographies. While we can solve problems of up to 16 tasks in less than a minute for one-to-many, the number of tasks we can handle in such short time decreases to 11 and 9 for few-to-many and many-to-many, respectively. The main reason for this difference is the underlying routing problem associated with finding the route to serve all tasks in a job. While this routing problem is a Travelling Salesman Problem with Time Windows (TSPTW) for

instance type one-to-many, it is a TSPTW with Precedence Constraints for the other instance types.

The number of jobs in Figure Figure 2.7 also shows that the number of feasible jobs decreases from g1 to g2 to g3 since the number of the nodes that has to be visited for g1 to to g2 to g3 increases. This means that finding a feasible route in many to many case (g3) is harder than finding in the one-to-many case.

2.8 Conclusions

In this study, we introduce a variant of the dynamic pickup and delivery problem that aims to utilize the excess capacity of the existing traffic flow in urban areas. We consider a fleet of dedicated vehicles and a set of dynamically arriving ad-hoc drivers who are willing to make a small detour in exchange for a small compensation. We formulate the associated problem as a matching problem with side constraints. To handle real-time updates, we propose a rolling horizon framework that re-optimizes the system whenever new information becomes available.

We also investigate the viability of the crowdsourced delivery concept under the setting of a peer to peer platform. We test the performance of the platform with a simulation study based on three different instance types: a single origin, multiple origins and random origins. As expected, the time flexibility and the stop willingness of ad-hoc drivers have a strong impact on the performance of the system. Also, we compare the performance of the crowdsourced delivery system with a delivery system where all tasks are served by dedicated drivers. The results indicate that there are clear benefits of using ad-hoc drivers in addition to a fleet of dedicated vehicles.

Our results suggest that a setting in which store customers serve delivery tasks that originate from that store may be most suitable for crowd-delivery. In this setting, we see most benefits from the use of ad-hoc drivers and require the smallest number of dedicated vehicles to serve all tasks. Moreover, the planning problems in this setting are easier to solve.

As this is one of the first papers that focuses on crowdsourced delivery, we see several directions for future research. One relevant area for new research is the

development of fast heuristics to solve larger instances in reasonable time, especially in relation to the use of dedicated vehicles together with crowd drivers. Another research direction is to investigate the impact of allowing the parcel transfer between drivers. The repositioning of the parcels according to the trips of the ad-hoc drivers will help to increase utilization of those drivers, which means less system-wide distance.

Another interesting area of future research is to explore the impact of different payment, pricing and incentive schemes to support crowdsourced delivery.

2.9 Appendix

2.9.1 MIP formulation TSP-TWPC

We can formulate the TSP-TWPC for a single driver k as a mixed integer problem. Recall that o_p, o_d and d_p, d_k represent origin and destination nodes for task p and driver k, respectively. The route always starts from the origin of the driver and ends at his destination. Let N^P be a set of nodes that correspond to the origins and destinations of the tasks in P, and let N be set of nodes including the origin and the destination of the driver. Let N^{P^+} and N^{P^-} denote the nodes associates with the origins and the destinations, respectively.

Let x_{ij} be a binary decision variable that is equal to 1 if the driver uses arc $(i, j), i, j \in N$ and 0 otherwise. Let c_{ij} be the cost of using arc (i, j). The continuous variable $B_i, i \in N$ represents the arrival time of the driver to node i.

Then, the mixed integer problem can be formulated as follows:

$$\min \sum_{i,j \in N} c_{ij} x_{ij} - c_{o_k,d_k} \tag{2.5}$$

subject to

$$\sum_{j \in N} x_{i,j} = 1 \qquad \forall i \in N^P, \tag{2.6}$$

$$\sum_{j \in N^p} x_{o_k,j} = 1 \tag{2.7}$$

$$\sum_{i \in N^p} x_{i,d_k} = 1 \tag{2.8}$$

$$\sum_{j \in N} x_{ij} - \sum_{j \in N} x_{ji} = 0 \qquad \forall i \in N^p, \tag{2.9}$$

$$\sum_{i,j \in N} I_{i,j} x_{ij} \leq Q_k \tag{2.10}$$

$$B_{d_p} \geq t_{o_p,d_p} + B_{o_p} \qquad \forall p \in P, \tag{2.11}$$

$$B_j \geq B_i + t_{ij} - M(1 - x_{ij}) \qquad \forall i,j \in N, \tag{2.12}$$

$$e_i \leq B_i \leq l_i \qquad \forall i \in N, \tag{2.13}$$

$$x_{ij} \in \{0,1\} \qquad \forall i,j \in N$$

$$B_i \geq 0 \qquad \forall i \in N,$$

The objective (2.5) is to minimize the total travel costs to serve all delivery tasks by the driver. Constraint (2.6) ensures that each task is served exactly once. Constraints (2.7) and (2.8) make sure that the driver starts at his origin and ends at his destination. Equations (2.9) represent the flow conservation constraints. Constraint (2.10) ensures the maximum number of stops per driver. In constraint (2.10), the indicator function $I_{i,j}$ checks whether node i and j represent the same physical location or not. If yes, then $I_{i,j}$ takes value 1, otherwise it takes 0. Constraints (2.11) ensure the precedence relations between pickup and delivery points. Constraints (2.12) and (2.13) represent the time window constraints.

2.9.2 Matching formulation hindsight benchmark

Note that a dispatch node is not associated with a particular backup vehicle but only denotes a possible dispatch at a certain time from a certain location.

Let A be the set of all feasible arcs. Let $J_k,\ \ k \in D$ denote the collection of jobs that driver k can serve, and $J_p,\ \ p \in P$ denote the set of jobs that contains task p. Let x_{kj} be the binary decision variable that indicates whether the arc between driver k and job j is in the solution ($x_{kj} = 1$) or not ($x_{kj} = 0$) and let y_j be the binary decision variable that indicates whether job j is served by a backup vehicle ($y_j = 1$) or not ($y_j = 0$). The coefficient c_{kj} represents the weight of the arc (k, j), which denotes the cost if driver k is assigned to job j (where c_j is the cost when job j is served by a backup vehicle.) Then, the problem that aims to minimize the total cost can be formulated as follows:

$$\min \sum_{(k,j)\in A} c_{kj}x_{kj} + \sum_{j\in J} c_j y_j \tag{2.14}$$

$$\text{s.t} \sum_{j\in J_k} x_{kj} \leq 1 \ \ \forall k \in D, \tag{2.15}$$

$$\sum_{j\in J_p} \sum_{k\in D} (x_{kj} + y_j) = 1 \ \ \forall p \in P, \tag{2.16}$$

$$x_{kj} \in \{0, 1\} \ \ \forall (k, j) \in A.$$

Equation (2.14) is the objective function that aims to minimize the sum of the costs of the matches and the backup vehicles. Constraints (2.15) makes sure that each driver is assigned to at most one job. Note that this constraint only applies to the ad-hoc drivers since we do not explicitly restrict the number of backup vehicles in this formulation. Constraints (2.16) make sure that each job is assigned to one of the drivers or a backup vehicle.

2.9.3 Determining the number of backup vehicles

The solution to the above problem formulation provides the matches between the jobs and the ad-hoc drivers and the selected backup dispatches. The subset

of backup dispatches that is selected in the solution is denoted by $B' \subseteq B$. To determine the minimum number of required backup vehicles to cover all dispatches, we need to solve a problem that resembles an interval scheduling problem.

We model this problem on a directed graph $G = (V, A)$, where $V = \{s, B', t\}$ includes a node for each dispatch $b \in B'$, a source node s and sink node t. Let A be the set of arcs. We create an arc from s to every dispatch in B' and from every dispatch in B' to t and an arc between every pair of dispatches that could potentially be served by the same vehicle. In particular, we only add an arc (i, j) between dispatch i and j if: $l_j \geq e_i + \tau_i$, where $[e_i, l_i]$ is the dispatch time window, and τ_i is the duration of dispatch i. Let x_{ij} be a binary variable that is equal to 1 if arc $(i, j) \in A$ is selected, and 0 otherwise. Let t_b^d be the actual dispatch time of dispatch b. This gives the following formulation:

$$\min \sum_{i \in B'} x_{si} \tag{2.17}$$

$$\text{s.t} \sum_{i \in V \setminus \{t\}} x_{ij} = 1, \quad \forall j \in B' \tag{2.18}$$

$$\sum_{i \in V \setminus \{t\}} x_{ib} - \sum_{j \in V \setminus \{s\}} x_{bj} = 0, \quad \forall b \in B' \tag{2.19}$$

$$t_j^d \geq (t_i^d + \tau_i) x_{ij}, \quad \forall i \in B', j \in B' \tag{2.20}$$

$$e_i \leq t_i^d \leq l_i, \quad \forall i \in B' \tag{2.21}$$

$$x_{ij} \in \{0, 1\}, \quad (i, j) \in A$$

The objective function (2.17) minimizes the outflow from the source node, which is equivalent to minimizing the number of backup vehicles. Constraint (2.18) ensures that each dispatch is covered by exactly one vehicle. Constraint (2.19) guarantees

that the inflow and outflow of each node are consistent. Constraints (2.20) and (2.21) ensure that the dispatch time windows are satisfied.

2.9.4 Proof of Observation 3

This proof is constructive and provides a procedure to transform an arbitrary route into a *clustered* route.

We start with an arbitrary route r which is fully characterised by the ordered sequence of pick-ups and deliveries. The pick-up location of task j reads p_j, and the delivery location of task j reads d_j. We write that the route will start with k pick-ups, where $k \geq 1$, and then l deliveries will follow, where $l \leq k$, before another pick-up (of task $k+1$) will follow. The sequence of the first $k+l+1$ pick-ups and deliveries can be written as

$$p_1, \ldots, p_k, d_{j_1}, \ldots, d_{j_l}, p_{k+1}.$$

Here the deliveries correspond with a subset of the tasks of the pick-ups, i.e., $\{j_1, \ldots, j_l\} \subseteq \{1, \ldots, k\}$; an item can only be delivered once it has been picked up.

In the case when $k = l$, the first k pick-ups have all been delivered, so the vehicle returns to the next pick-up p_{k+1} empty, and it is also possible that the job has been completed, so that there is no next pick-up $k+1$. In that case, we are finished. In the case that there is another pick-up remaining, we can proceed with the analysis as above with the first pick-up p_{k+1}.

In the case when $k > l$, the first k pick-ups have not yet all been delivered, so the vehicle is not empty when it returns to pick-up task $k+1$. We need to expand the sequence of pick-ups and deliveries further. The route r can be written as

$$p_1, \ldots, p_k, d_{j_1}, \ldots, d_{j_l}, p_{k+1}, \ldots, p_{k+r}, d_{j_{l+1}}, \ldots, d_{j_{l+s}}, p_{k+r+1}, \ldots,$$

where $r \geq 1$ and $l + s \leq k + r$.

Consider p_i, with $1 \leq i \leq k$. In case $i \notin \{j_1, \ldots, j_l\}$, task i is carried back to the pick-up location of tasks $k+1, \ldots, k+r$. We need to consider the following two cases:

1. The pick-up location of task i is not revisited while picking up tasks $k+1, \ldots, k+r$. This implies that this task does not violate the clustered route property during the first $k+r$ pick-ups.
2. The pick-up location of task i is revisited while picking up tasks $k+1, \ldots, k+r$; let task $j \in \{k+1, \ldots, k+r\}$ be the first task in the sequence which has the same pick-up location as i. Then we change the order of pick-ups and deliveries in the route by moving the pick-up of task i right in front of the pick-up of task j. The total distance travelled en route will not increase, and the pick-up of task i will simply be delayed until the moment that task j will be picked up at the same location. The pick-up of task i still occurs before task i is delivered.

This procedure can be followed for all tasks $i \notin \{j_1, \ldots, j_l\}$. The order in which these tasks are considered may have an impact on the order in which tasks are picked up at the same location, but this is not relevant.

We have now arrived at a sequence of pick-ups and deliveries where the first subsequence of pick-ups $\{p_i\}$ and the first subsequence of deliveries $\{d_j\}$ satisfy the clustered route property. We now need to proceed to analyse the remaining sequence of pick-ups and deliveries in a similar manner. Note that some of the tasks, which have been picked up in the first subsequence, may not have been delivered yet in the first subsequence of deliveries. They should be incorporated in the further analysis as well, but this does not change the line of argument.

The transformation of the route by pushing forward deliveries as described above is completed after a finite number of steps. The resulting route has the clustered route property.

In the constructive proof above, no time infeasibilities are introduced. In the following, we can also see an observation about the tasks with compatible time windows and with similar pickup (delivery) locations, could be picked up (delivered) simultaneously.

- If two tasks i_1 and i_2 where p_{i_1} and p_{i_2} satisfy $p_{i_1} = p_{i_2}$ but they have incompatible time windows, then they can not be picked up by the same vehicle at any time.

- If two tasks i_1 and i_2 where d_{i_1} and d_{i_2} satisfy $d_{i_1} = d_{i_2}$ but they have incompatible time windows, then they can not be delivered by the same vehicle at any time.

Proof by contradiction. Say that for a feasible route, tasks i_1 and i_2 are together in a same vehicle at the same time

1. If $p_{i_1} = p_{i_2}$, then i_1 and i_2 could have been picked up together while respecting to the earliest pick-up times
2. If $d_{i_1} = d_{i_2}$, then i_1 and i_2 can be delivered together while respecting to the latest delivery times.

□

3 On-demand crowdshipping with store transfers

3.1 Introduction

On-demand delivery is expected to account for 15% of the last-mile delivery market by 2020 (Joerss et al., 2016). Retailers such as Amazon now offer home delivery of goods purchased online in as little as one hour. However, short delivery lead-times make it difficult to combine multiple deliveries in a single route. Therefore, on-demand delivery requires frequent and inefficient small vehicle dispatches. To make this type of delivery feasible without charging high fees to customers, service providers are looking for new operating models to enable faster and more cost-efficient delivery.

One way to increase efficiency is by using idle capacity in existing traffic flows (Sampaio et al., 2019; Savelsbergh & Van Woensel, 2016). This idea, that is often referred to as 'crowdshipping', involves regular people (which we refer to as crowd drivers) who make deliveries along the route to their destinations. A number of start-ups such as Roody and Deliv use crowd drivers for parcel delivery. Furthermore, Walmart started experimenting with using in-store customers and/or store employees to serve online customers on their way home from the store (Bhattarai, 2017). This is much more convenient for the crowd drivers than a system in which they first have to pickup packages from a fulfillment location.

Crowd drivers deliver parcels to their destinations by making small detours from their planned journeys. Therefore, we assume that delivering an online order with a crowd driver is cheaper than using a regular employee. We employ a compensation

scheme for participating drivers in which a driver receives a small financial incentive based on the detour distance that he or she makes for the delivery (see similar compensation schemes Archetti et al. (2016), Arslan et al. (2019a), and Dayarian & Savelsbergh (2017))).

One of the distinguishing features of a crowd-based delivery system is that crowd drivers act voluntarily, and are not solely motivated by financial incentives. This means that it is important to explicitly take into account the planned itinerary and time schedule of the crowd drivers. Miller et al. (2017), for example, show that crowd drivers require smaller compensations to accept tasks that require only small deviations from their planned trips. A crowd delivery experiment for library books in Finland shows that the majority of crowd deliveries (over 80%) take place within a five kilometer range Paloheimo et al. (2016). This suggest that crowd drivers are more favourable towards shorter detours.

As a company has less control on the crowd drivers than regular employees, it is challenging to serve all delivery tasks by only using crowd drivers, particularly in on-demand delivery services in which delivery lead-times are short. In the context of ride-sharing, Agatz et al. (2011) show that it is unlikely to guarantee service, even with a large number of crowd drivers. Therefore, we see that crowd delivery systems typically use regular drivers as a fallback option (see e.g., Lee & Savelsbergh (2015)).

Our stylized model captures the most important features of an online shopping platform that uses store-based crowd drivers to make deliveries. Multi-store shopping platforms, such as Google Express and Postmates, provide online shopping and on-demand home delivery services from different member stores to online customers. The system centrally coordinates delivery tasks and crowd drivers, and automatically assigns tasks to drivers.

To make best use of the available supply of crowd drivers, two possible ways can be considered: multiple delivery tasks can be delivered by a crowd driver, or multiple drivers can make a single task delivery. In this paper, we analyze the latter one, in which a parcel can be transferred to another crowd driver on its way to the destination. To reduce time synchronization issues, we only consider transfers at stores where a parcel can be temporarily stored so that drivers do not have to

necessarily be a the transfer point at the same time. This eliminates inconvenient waiting times at the transfer location for the drivers.

The contribution of this paper can be summarized as follows. First, we introduce a new on-demand crowdshipping problem that involves store transfers. Second, we propose a solution approach and different dynamic dispatching strategies that explicitly take store transfers into account. Finally, we conduct an extensive computational study to quantify the merits of store transfers for different parameter values.

The remainder of this paper is organized as follows. We discuss the related literature in Section 3.2. Section 3.3 provides the description of the problem. We present our solution approach in Section 3.4. In Section 3.5, we present our numerical experiments. Finally, in Section 3.6, we provide some concluding remarks and directions for future research.

3.2 Literature Review

Our work fits within the emerging stream of research on shared mobility. Most of the early research in this area focused on ride-sharing systems; see Agatz et al. (2012b) and Furuhata et al. (2013b) for reviews. More recently, researchers started to explore using the *crowd drivers* to support freight transportation.

Sadilek et al. (2013) and Suh et al. (2012) investigate using spare vehicle capacities of regular people to support carrying packages from a to b. Sadilek et al. (2013) tests this idea with a simulation based on commuter data. Suh et al. (2012) consider a conceptual model that aims to leverage of social networks of people, e.g., friends or family, for parcel deliveries to each other. Both studies present that crowd-based delivery systems can potentially contribute to a significant reduction of carbon emissions and system-wide travel costs. A study by Devari et al. (2017) delves further into the idea of exploiting social networks of customers. In the conducted simulation study based on survey and the data of city of Alexandria, Virginia, the USA shows that a crowdsourced delivery model for friends may reduce the total vehicle miles up 57%.

An exploratory study by Rougès & Montreuil (2014) examines the business models in crowdsourced delivery and identify that the typical setting is Business-to-Consumers (B2C) for the delivery of one single parcel at a time. Carbone et al. (2015) and Rougès & Montreuil (2014) classify the types of logistics existing in sharing economy based on an exploratory analysis on business cases. These two papers suggest that the success of crowd logistic depends on a large enough pool of crowd participants.

Several researchers study hybrid systems in which crowd drivers and dedicated employees serve a number of customers in parallel. Archetti et al. (2016) study the use of crowd drivers to complement deliveries of a fleet of dedicated vehicles. Their results show that incorporating crowd drivers can result in significant cost savings. Arslan et al. (2019a) examine a similar system where parcel delivery requests and crowd driver appear within a day dynamically. They present a framework that matches drivers with parcels and routes the dedicated vehicles in real-time.

It is possible to increase the efficiency of a dynamic crowd-based delivery system by considering probabilistic information of future arrivals of delivery tasks and crowd drivers. Dahle et al. (2017) and Dayarian & Savelsbergh (2017) formulate stochastic routing problems and propose dispatching strategies. Both studies show that exploiting the forecasts of participating crowd drivers improves the system performance by 2-3%.

Another way to increase the efficiency of crowd-delivery systems is to introduce parcel transfers. Chen et al. (2017) study a long-haul package transportation problem in which delivery tasks and crowd drivers are known in advance. They formulate a mixed-integer problem that crowd drivers are allowed to exchange parcels between each other. The results show that transfers bring benefits to crowdsourced delivery systems. Kafle et al. (2017) design a hybrid urban delivery system in which dedicated trucks carry parcels to relay points and crowd drivers take assigned parcels from relay points and deliver them to their destinations. Nonetheless, transfers between crowd drivers are not considered in this study.

Some recent papers in ride-sharing aim to incorporate passenger transfers to increase system performance in terms of travel distance. Coltin & Veloso (2014), Masoud & Jayakrishnan (2017a) and Masoud & Jayakrishnan (2017b) analyze dynamic ride-sharing systems with transfers. They show that introducing passenger transfers

can help to reduce the total travel distance and increases the match rate of riders. Different than in our setting, passenger-related constraints as the waiting time at a transfer station are considered.

3.3 Problem Description

We consider an online platform that facilitates on-demand delivery from different store locations by using in-store customers as crowd drivers. Online customers place orders that correspond to delivery tasks. The platform receives n_p delivery tasks and n_k crowd drivers throughout the service period. Let P be the set of delivery tasks, K be the set of crowd drivers and S be the set of stores. Let $P(s) \subset P$ represent the delivery tasks that have to be fulfilled from store $s \in S$. Similarly, $K(s) \subset K$ refers to the crowd drivers that originate from store $s \in S$.

We let N^p, N^k be the sets of destinations of the delivery tasks and the drivers, respectively, and N^s is the set of the store locations. Set $N = N^p \cup N^k \cup N^s$ contains all relevant locations. The travel distance and the travel time between two points, $a, b \in N$, denoted by δ_{ab}, t_{ab}, respectively; both follow the triangle inequality.

Each delivery task $i \in P$ is characterized by an origin $o(i) \in N^s$, a destination $\omega(i) \in N^p$, a placement time a_i, an earliest pickup time e_i ($e_i > a_i$) and a latest delivery time at the destination l_i ($l_i > e_i + t_{o(i),\omega(i)}$). We assume that a task corresponds to a small parcel that fits in a regular car so we ignore capacity considerations. Each delivery task can only be fulfilled from one specific store. The difference between the placement time and the earliest pickup time captures the time required for picking and packing at the store.

A crowd driver $k \in K$ announces a trip at time a_k, with an origin equal to store $s \in S$, an earliest departure time from the store e_k ($e_k > a_k$), a destination $\omega(k) \in N^k$ and a latest arrival time l_k. We make a distinction between the announcement time and the earliest departure time to captures the fact that an in-store customers may announce at the time she is ready to leave or while still shopping at the store. The difference between l_k and $e_k + t_{o(k),\omega(k)}$ denotes the departure time flexibility and detour of driver k. For the convenience, T_k denotes

the maximum amount of time driver k's willingness to make detour. Furthermore, we assume that a driver can make at most one additional stop on his/her way to the destination.

In case a delivery task cannot be served by any of the crowd drivers, a back-up vehicle makes the delivery. We assume that the back-up vehicles are always available for calls, and depart from the store at the latest time, i.e., $l_i - t_{o(i),\omega(i)}$. As a result, the backup vehicle carries only a single delivery task per dispatch.

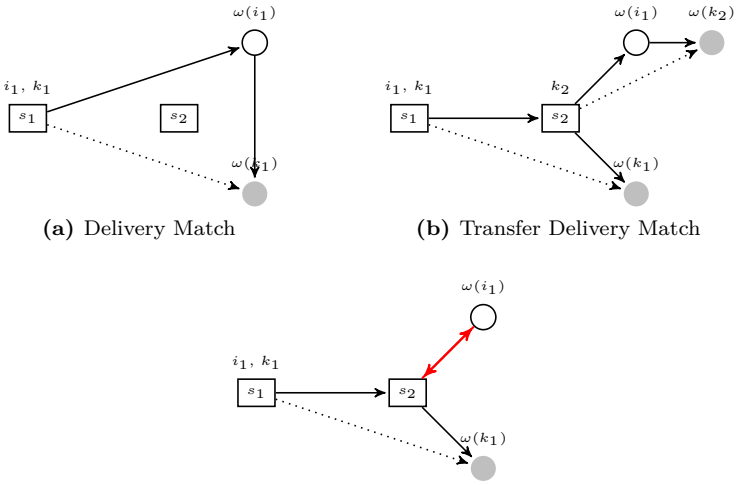
From the perspective of a delivery task, we can distinguish two types of deliveries: (i) a delivery in which a crowd-driver or a backup vehicle deliver directly from the origin store of the task and (ii) a delivery in which a crowd-driver or a backup vehicle deliver from a transfer store. From the crowd driver's perspective, we can distinguish two types of deliveries: (i) a delivery of a single task to the destination and (ii) a delivery of one or more tasks to one of the stores.

Note that the deliveries including transfers involve the synchronization of different crowd drivers and tasks. In most simple example, a package associated with a particular task first has to be delivered to a transfer store before it can be picked up by a second crowd driver for final delivery. To model these interactions, we define the following type of basic match structures: (i) a *delivery match* between one single driver and one single task, (ii) a *transfer delivery match* between one driver from the origin store, one or more tasks and one or more drivers from the transfer store and (iii) a *store delivery match* between one driver from the origin store and one or more tasks.

Figure Figure 3.1 illustrates each match type with an example in which a delivery task i_1 with destination $\omega(i_1)$ and origin $s_1 = o(i_1)$. Figure Figure 3.1a shows a direct delivery in which crowd driver k_1 makes the delivery of task i_1 and continues his/her destination. Figure Figure 3.1b shows a transfer match that task i_1 is delivered via store s_2 by crowd drivers k_1 and k_2. Finally, Figure Figure 3.1c represents the store delivery match.

While a delivery match always involves only one task, transfer and store matches can involve multiple tasks as a crowd driver who moves between two stores can carry multiple items without increasing her detour or the number of stops. A crowd driver $k \in K$ can carry up to b_k parcels between member stores.

Figure 3.1: Delivery options: travelling from the store locations (squares) to destinations (circles)



(a) Delivery Match **(b)** Transfer Delivery Match

(c) Store Delivery Match

$\rightarrow$ delivery trip, $\dashrightarrow$ original trip, $\rightarrow$ back-up delivery

The system serves all delivery tasks either by using the crowd drivers or the backup vehicles. Each crowd driver who carries a parcel gets a compensation that is proportional to the length of detour from the planned trip. The cost of using a backup vehicle is also proportional to the distance of the delivery. The cost per distance-unit of the backup vehicle is β times more expensive than that of a crowd driver. The objective of platform is to minimize the total delivery costs by determining tasks and crowd drivers pairs.

In this study, we only allow transfers at the stores and we assume that there are no cost associated with a store transfer. It is, however, possible to introduce a transfer fee in our model. Furthermore, we do not consider any specific capacity limits at the stores for transfers.

3.4 Solution Approach

The system receives delivery tasks and crowd drivers continuously; therefore, we must determine potential delivery matches at many times throughout the service period. To cope with these dynamics, we use an event-based rolling horizon approach, in which delivery plans are repeatedly made using all known information at the time of an arrival of a task or a crowd driver. Therefore, at each optimization run, our algorithm first identifies optimal pairs by solving a matching problem and then schedules execution times for crowd drivers and back-up vehicles.

In this section, we present our dynamic solution approach. In Section Section 3.4.1, we present a model to determine tentative matches in each optimization run within the rolling horizon. In Section Section 3.4.2, we present the dispatching strategies that determine when to execute the tentative matches.

3.4.1 Offline matching problem

As in Stiglic et al. (2015), we model a bipartite matching problem with side constraints to determine the set of matches that minimizes the total cost to serve all tasks. This matching problem pairs task-sets and driver-sets as defined follows. A task-set j is a collection of delivery tasks in P and a driver-set is a collection of crowd drivers in K. If a task-set (driver-set) j (d) consists of delivery tasks (drivers) that only originate from store $s \in S$, we denote it as j^s (d^s), for the sake of convenience. Let J be the set of all task-sets and let D be the set of all driver-sets. We create a node for each task-set $j \in J$ and a node for each driver-set $d \in D$.

An arc between node j and node d implies that drivers in d are able to deliver the delivery tasks in j on time. We create an arc for an feasible pair (j, d) if the task-set j and driver-set d hold the following conditions:

- *Time-schedule constraint:* a crowd driver $k \in d$ can only depart after its earliest departure time e_k and should arrive at its destination before l_k. Similarly, each task $i \in j$ cannot be picked up earlier than e_i and should be delivered no later than l_i.

- *Detour constraint:* the total time deviation of a crowd driver $k \in d$ from her original trip duration should be less than the maximum detour willingness T_k.
- *Capacity constraint:* a crowd driver k cannot carry more than b_k tasks between two stores.
- *Synchronization constraint:* the departure time of a crowd driver k from transfer store s who is assigned to complete a delivery task i that is transferred to the store s can not be earlier than task i's earliest arrival time to store s.

The weight of arc (j, d) denotes the shipment cost of delivery tasks in j by driver-set d. Let x_{jd} be a binary decision variable that indicates whether the arc between task-set j and driver-set d is in the solution. The coefficient c_{jd} represents the weight of the arc (j, d). Then, the problem that aims to minimize the total cost can be formulated as follows:

$$\min z = \sum_{j \in J, d \in D} c_{jd} x_{jd} \tag{3.1}$$

s.t

$$\sum_{j \in J_i, d \in D} x_{jd} = 1 \quad \forall i \in P, \tag{3.2}$$

$$\sum_{d \in D_k, j \in J} x_{jd} \leq 1 \quad \forall k \in D, \tag{3.3}$$

$$x_{jd} \in \{0, 1\}.$$

Equation (3.1) is the objective function that minimizes the cost to serve all tasks. Constraints (3.2) ensure that each task is served exactly once. Constraints (3.3) guarantee that each crowd-set serves at most one task-set.

The matching problem defined in (3.1) - (3.3) can be solved quickly using a commercial IP solver for a given set of feasible matches. While, the number of potential matches grows exponentially with the number of tasks and drivers, many sets of tasks and drivers cannot be part of a feasible match due to the capacity, time and detour restrictions. Therefore, we efficiently generate the different sets in the following three steps.

First, we generate all direct matches by enumerating all combinations. We can assess the feasibility of a potential *direct delivery match* by checking the time and detour restrictions. The cost of a direct delivery match (j, d), c_{jd}, is proportional to the detour of driver $k \in d$ to deliver parcel $i \in j$, i.e., $c_{dj} := \delta_{o(i)\omega(i)} + \delta_{\omega(i)\omega(k)} - \delta_{o(i)\omega(k)}$.

Next, we generate all store and transfer matches that involve a single task or singleton task-set. There, we make use of the following observation.

Observation 1: At time t, if the travel time from the origin $o(i)$ to the destination $\omega(i)$ of task i via store s takes more time than the remaining delivery lead-time $l_i - t$, it is not feasible to use store s as a transfer store for this task.

For each task $i \in P$; *i.e.* $j = \{i\}$, we then go over all crowd drivers $K(o(i))$ to check for possible transfers to each feasible transfer store $s \in S$. We consider a 'store transfer' to be feasible for a task i if a crowd driver at the origin store $o(i)$ can carry the task to store $s \neq o(i)$ such that the task can be delivered from store s to its destination on time. The matches represent the 'store delivery' matches.

A store delivery match of a singleton task-set $j = \{i\}$ by driver $d = \{k\}$ by definition involves a store $s \in S$. However, we can eliminate this match's dependency to a store s easily filtering out feasible deliveries via stores that are economically not attractive. In other words, we only consider a single store match for a pair (j, d) such that minimizes the delivery cost; *i.e.* $s^* = \arg\min_{s \in S} \delta_{o(i)s} + \delta_{s\omega(k)} - \delta_{o(i)\omega(k)} + 2\beta\delta_{s,\omega(i)}$. Furthermore, the delivery cost of using s^* match is also equal to the cost of this match.

We can then generate all *transfer delivery matches* with singleton task-set $j = \{i\}$ by checking for each store match with task i whether or not there is a crowd driver at the transfer store that can make the final delivery within the available time. Precisely, to generate transfer matches from a store delivery match; initially we fix the store delivery match of tasks i at store s by driver k as a base. Then, we compute the earliest time that task i arrives to store s, which is the earliest time that crowd driver $k' \in K(s)$ can pick it up. Last, we browse among crowd drivers in $K(s)$. For each $k' \in K(s)$, who can make task i's delivery with respect to its time-schedule and detour constraints is stored. The drivers k and k' form the driver-set d, which is a element of transfer match of (j, d). The cost of this match is equal to the detour distance of the two crowd drivers k and $k' \in d$.

After all direct, store and transfer matches that involve a singleton task-set are generated, we generate task-sets of multiple tasks by using store delivery matches recursively based on the following observation:

Observation 2: A store match between a driver k and a set of tasks $j \subset P(o(i))$ via store $s \neq o(i) : i \in j$ with $|j| \geq 2$ is time feasible only if the match between driver k and subset of tasks $j^{'} \subset j$ is time feasible for all $j^{'} \subset j$ via store s.

This observation follows from observation 1 in Stiglic et al. (2015). This means that a store delivery match with two tasks, a crowd driver and transfer store is only feasible if both tasks are individually feasible for this driver and store for a transfer. We use this property to reduce the number of jobs to be considered in our recursive algorithm.

To generate transfer matches for bundled task-sets, we follow a similar idea as we use to form transfer matches for single tasks. A store match of task-set j, transfer store s and driver k who carries the task-set to store transfer s is chosen as a base of the transfer matches. This time, we screen as many drivers as the cardinality of task-set j. When the drivers d^s who are able to complete deliveries of tasks in j, the driver set $d := \{k\} \cup d^s$ is formed. The pair (j, d) is stored as one of the feasible transfer match.

As a feature of generating bundled transfer matches, we can save the matches that some but not all of tasks in the task-set that are delivered from transfer store to their destinations. To classify these delivery types, we split this type matches into two matches such that one part is a store match and the other one is a transfer match.

3.4.2 Rolling horizon framework

Within the rolling horizon, we repeatedly solve an optimization problem. An optimization run results in a set of *tentative* matches. A tentative match has an earliest and latest departure time based on the time schedules of the associated tasks and drivers. By default, we use a 'latest dispatch' strategy. This means that the tentative match is executed at the latest possible time unless an event (arrival

of a delivery task or crowd driver) occurs before this time. In that case, we run a new optimization problem including all active tasks and drivers.

We assume that it is not possible to change an assignment of a crowd driver after he or she started driving towards the delivery location. For the transfer match, however, there is still some flexibility to adjust the second leg of the journey of a task later. That is, while moving the task to the transfer store, new arrivals may create better matches for either task or driver. This gives rise to two dispatching strategies: (i) fix: commit to the whole transfer match at the latest departure time of the first leg (ii) flex: only commit to the first leg of the transfer match at the latest departure time of the first leg. The second leg of a transfer trip can be updated until the latest departure time of that leg.

Figure Figure 3.2 provides an example of a situation in which it is advantageous to update the match after the first leg. At time t, optimization run q, we create a tentative transfer match in which task i is moved to store s by driver k_1 and then moved to the delivery location by k_2. The dashed lines represent the feasible commitment time intervals for the different crowd driver. In this example, we execute the transfer match at the latest feasible time. During the transfer, represented with a thick red arrow, another driver k_3, who has a smaller detour than the designated driver k_2, announces her feasibility for a possible pickup from the transfer store. To use this opportunity, we break the transfer match with k_2 and create a new match for the last part of the trip with k_3.

Figure 3.2: An illustrative example for a transfer job $j = (i, \{k_1, k_2\})$ and the possibility of updating task-driver pair at the transfer store due to the new arrival

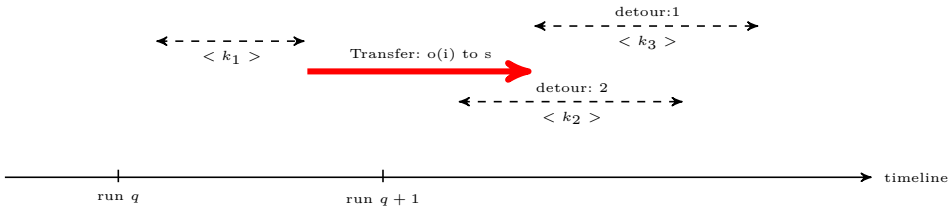
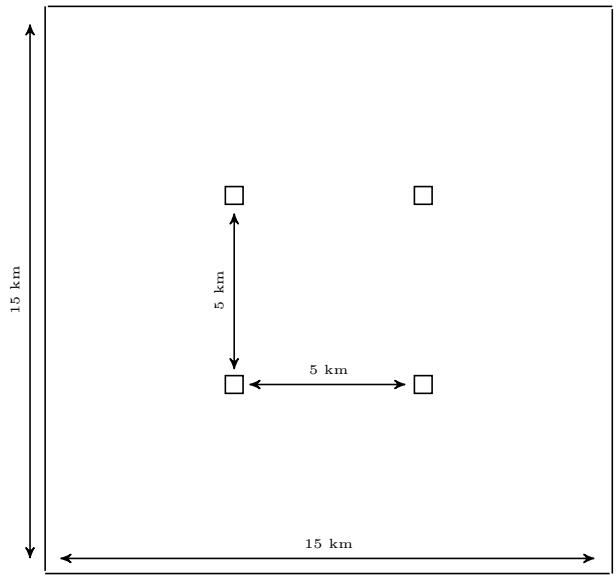


Figure 3.3: The visualization of the plane



3.5 A Computational Study

The numerical experiments are implemented in C++ and ran on a computer with 2,7 GHz Intel Core i5 processor. Gurobi (2016) is used as the MIP solver. In the following, we report the details of our instances and the results of our computational study.

3.5.1 Instance generation

Similar to Arslan et al. (2019a), we generate instances on square plane with length 15km. There are four stores located on the corners of a five square plane with length 5km inside the large plane (see Figure Figure 3.3). We use euclidean distances to compute travel distances between two points, and we assume a constant vehicle speed of 60 km/h.

The announcement time of a task or a crowd driver is drawn uniformly within the operating period of 480 minutes. All tasks can be picked up ten minutes after

their announcement and should be served within 60 minutes. Similarly, each crowd driver is ready to leave the store ten minutes after her announcement. In the base scenario, we only allow a single transfer per delivery task and a crowd driver can carry at most two parcels between stores. We assume that handling time at a store is the part of the travel time, and the cost per km for backup vehicle twice of the compensation that is paid for a crowd driver per km of detour. Furthermore, we assume that the distance of backup vehicle for a delivery is twice of the between the store and the order's destination, which represents the case in which the vehicle makes a single delivery from the store.

Table 3.1: Characteristics of the base case instances

Definition	Values
No. of delivery tasks	50-100-200
No. of crowd drivers	50-100-200
Delivery lead-time (L)	60 min
Announcement lead-time (u)	10 min
Departure time flexibility	20 min
Maximum transfers per parcel	1
Bundle capacity (b)	2
No. of stores	4
Back-up penalty, β	2

The delivery tasks and crowd drivers all originate from one of the four store locations. The origin of tasks and drivers are randomly chosen among the pickup stores, whereas the destinations are uniformly distributed within the plane. For the base case, each instance has 50, 100 or 200 delivery tasks and crowd drivers with varying combinations. All parameters for the base case analysis can be found in Table Table 3.1.

To effectively assess the impact of proposed delivery types, we contemplate three settings as follows:

- *Direct deliveries*: Crowd drivers are only allowed to make direct deliveries.
- *+ Transfers*: Together with direct deliveries, crowd drivers can make deliveries via store transfers.

- *+ Transfers and store deliveries*: In previous setting, it is only allowed that the transferred tasks are completed by crowd-drivers. Nevertheless, in this setting parcels that are transferred to store can be delivered to their destinations by back-up vehicles.

3.5.2 Base results

We first present results for the 'hindsight' benchmark in which all information for a service period is available. We consider three densities in terms of delivery tasks and crowd drivers, *i.e.*, 50, 100, 200. Table Table 3.2 presents the total delivery costs, the costs per task and the fraction of tasks served fully by crowd drivers, *i.e.* matched. To facilitate an easy comparison, we normalize all the costs by dividing by the average of $100 - 100$ density.

Table 3.2: Hindsight results, $T_k = 20$ **mins** , $L = 60$ **mins**

$n_p - n_k$	Direct	+ Transfers	+ Trans & Store
50 – 50			
Total cost	70.6	50.5	43.4
Cost per task	12.7	9.1	7.8
Matched (%)	60.4	70.4	64.2
100 – 100			
Total cost	100	75.5	62.7
Cost per task	9.0	6.8	5.6
Matched (%)	75.7	78.7	71.5
200 – 200			
Total cost	133.4	112.8	92.2
Cost per task	6.0	5.1	4.1
Matched (%)	87.1	85.3	79.2

The results show a substantial cost reduction in the integrated system. For the 100-100 case, allowing transfers leads to 24.5% savings. The cost saving reaches to 37.3% in more integrated setting in which store drops are allowed on top of store transfers. A similar cost saving benefits of store transfers hold for other densities. We observe delivery tasks served by crowd drivers slightly increases. The percentage of matched tasks jumps from 75.7% to 78.7% in the 100:100 case.

Moreover, we observe the economies of scale in the system. The average delivery cost of a task decreases by 55.8% for the system that only direct deliveries are

allowed when the density increases from 50:50 to 200:200. The average delivery cost per task drops by 47.7% when store transfers are allowed. Interestingly, store transfers are more beneficial when the system has less density. The reason of this finding is that when the system has more crowd drivers and tasks, the possibility of finding economical direct deliveries is more likely.

Figure 3.4: The break-down of matched Tasks: $n_p - n_k : 100 - 100$**,** $T_k : 20$ **mins**

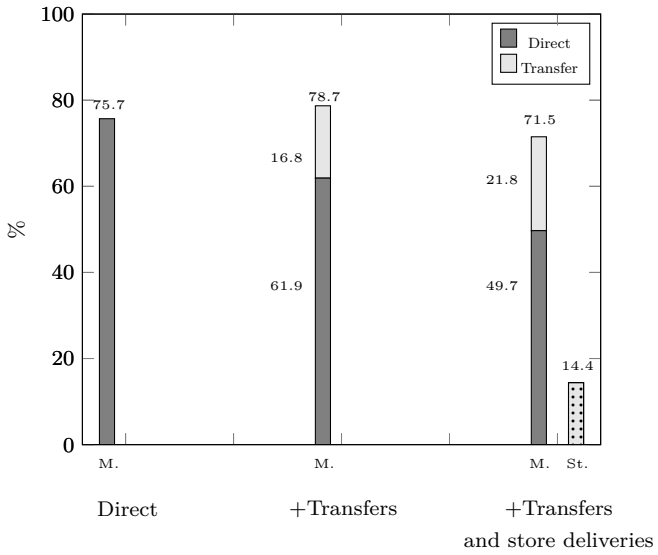


Figure Table 3.2 presents the ratio of delivery types for 100-100 density. Columns entitled 'M' and 'St.' represent the proportions of tasks are matched (delivered to their destinations), or delivered to stores, respectively by crowd drivers. Matched ratio contains the ratios of direct deliveries and transfer deliveries. Nevertheless, store deliveries represent the shipments that crowd drivers carry tasks to stores, but the deliveries are completed by backup vehicles. The labels below the horizontal axis represent which types of settings are considered into the experiments. For instance, the first column shows the system that only direct deliveries are allowed but in the second column represents the result for the system in which direct deliveries and transfers are allowed.

The results on Figure Table 3.2 show that allowing store transfers increases the matched delivery tasks, *i.e.* the ratio of deliveries to home addresses by crowd drivers. With the presence of transfer option, 16.8% of deliveries are done by using transfers while total home deliveries reaches to 78.7%. This break-down of matched tasks shows that nearly the quarter of deliveries become economically more advantageous with store transfers than direct deliveries. Furthermore, we observe that this pattern continuous in more integrated systems that transfers and store deliveries are allowed. The proportion of direct deliveries drops to 49.7% from 61.9%.

3.5.3 Impact of time flexibility, lead time and crowd drivers ratio

Table Table 3.3 shows the impact of crowd drivers' time flexibility on the system performance. The table presents the total cost and the matched tasks numbers for 10, 20, and 30 minutes of crowd drivers' departure and detour time flexibility. All results are normalized based on default 20 minutes time-flexibility and 100:100 density. The last column $\triangle$ quantifies the benefit of store transfers. As expected, more time-flexibility leads to lower system-wide costs; however, we observe that less time-flexibility leads to larger benefits for the store transfers. The results show that while store transfers brings 28.0% of cost savings when the detour flexibility is 10 minutes, the cost saving is 21.7% for 30 minutes detour flexibility. The reason of this pattern is as follows. More time flexibility increases the number of feasible and also economically advantageous direct delivery matches. A similar pattern can be seen for the matched tasks.

Table 3.3: Impact of time flexibility, $n_p - n_k : 100 - 100$

	Total Cost			Matched Tasks	
T_k	Direct	St. Transfer	$\triangle$	Direct	St. Transfer
10 min	108.3	77.7	-28.0	72.7	77.2
20 min	100.0	75.5	-24.5	75.7	78.7
30 min	93.8	73.5	-21.7	77.5	79.4

Table Table 3.4 presents the results for the cost saving that is gained by store transfers for various lead times and densities. The benefits of store transfers

increase along with participating number of crowd drivers, and decreases by longer delivery lead-times. The former pattern is expected since more crowd drivers available, more likely to find economical store transfers. The latter pattern is, however, in-line with the impact of time-flexibility on store transfers. The longer lead times enhances the chances of direct deliveries such that finding efficient direct deliveries is more likely.

Table 3.4: Impact of delivery lead-time and drivers' ratio, T_k : 20, n_p : 100

L \ n_k	50	100	150	200
45 min	6.97	27.32	45.53	59.73
60 min	5.78	24.49	42.94	58.13
90 min	3.97	20.59	40.04	56.02
120 min	3.63	18.23	38.37	54.57

3.5.4 Impact of dynamics

In this section, we present the results for a real-time implementation, in which the platform can take an action at the time of a crowd driver or task arrival. Table Table 3.5 shows the results for the dispatching strategies described in Section Section 3.4.2. We normalized the total cost results with the result that is found for 100:100 density when only direct deliveries are allowed.

Table 3.5: Dynamic results: L : 60 minutes, T_k : 20 minutes

$n_p - n_k$	Direct	Transfers	
50 – 50		Fix	Flex
Total cost	66.3	62.7	61.4
Cost per task	14.3	13.4	13.2
Matched (%)	56.4	55.0	56.8
100 – 100			
Total cost	100	93.5	91.8
Cost per task	10.7	9.9	9.8
Matched (%)	70.1	71.4	70.0
200 – 200			
Total cost	150.5	144.6	139.2
Cost per task	8.0	7.7	7.4
Matched (%)	80.1	74.5	75.1

Table Table 3.5 shows that the cost savings of store transfers in the dynamic environment are substantial; however the saving amount is lower comparing to the savings obtained in the hindsight setting. With store transfers, the system saves up to 9.0%. We also see that more flexibility in dispatching decisions at transfer points enhances the benefits of store transfers. All three densities, flexible dispatching rule generates results that the average cost saving is $1-2\%$ more comparing to the solutions obtained by the fixed dispatching strategy.

The dynamic setting results enable to quantify the value of perfect information. When the system does not know the future arrivals in advance, the average cost per delivery increases from 9.0 to 10.7 for 100-100 density when the system uses direct delivery only. Nevertheless, when store transfers are present, the average cost of a task delivery increases from 5.6 to 9.8. These two results show that the information for future arrivals is more valuable when the system employs the store transfers.

3.5.5 Benefits of bundling

In this section, we examine the impact of crowd drivers' capacity that is used to carry tasks between stores. Table Table 3.6 shows that the cost saving of store transfers for capacities of 1, 2, 3, and 4 task(s). These results are obtained in dynamic setting and the flex dispatching rule is chosen as a dispatch strategy. We normalized the numbers with the total cost of 100-100 direct delivery case.

Table 3.6: Impact of drivers' capacity for transfers; T_k : 20 minutes, L : 60 minutes

	Direct	Bundle Capacity			
$n_p - n_k$	Delivery	1	2	3	4
50-50	66.3	64.2	62.6	62.3	62.2
100-100	100.0	97.5	93.1	92.7	92.1
200-200	150.5	145.1	144.6	141.7	141.4

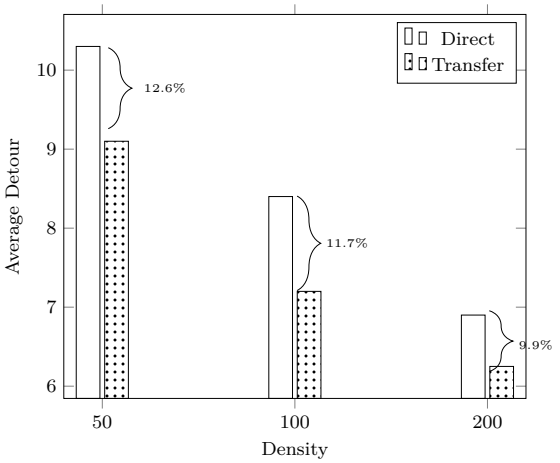
As expected, the larger crowd drivers' capacity for transfers leads to larger cost savings. We also observe that the marginal contribution of an additional driver capacity to the cost saving mitigates with a capacity size. Nevertheless, there is not a clear pattern for the marginal benefit of an extra drivers' capacity with

the density. The benefits of additional capacity seems more effective for 100-100 density than 50-50 and 200-200 densities. This pattern can be explained follows. For lower densities, like 50-50 density, the possibility of bundling tasks is difficult due to temporal and spatial remoteness of tasks. Therefore, an additional capacity for transfers becomes a redundant resource. However, denser cases, like 200-200 case, even though more tasks are suitable for bundling, direct deliveries become economically advantageous and it cancels out the additional transfer capacity benefit.

3.5.6 Detour lengths

One of the premises of store transfers is to reduce the average detour lengths of participating crowd drivers. Figure Figure 3.5 presents the average detours of crowd drivers with and without store transfers in increasing system density in the dynamic setting. The results show that allowing store transfers has a significant impact on the average detour length such that the average detour length of a participating crowd drivers decreases by 11.7%.

Figure 3.5: Impact on average detour lengths, $L = 60$ minutes, T_k : 20 minutes



It should be noted that proportional benefits of store transfers on average detour lengths is more than cost savings. While the cost saving based on store transfers

is 8%, the average detour decreases by 11.7%. This results show that the store transfers is valuable tool for system designer to construct model that increases the drivers' convenience.

3.6 Concluding Remarks

In this study, we analyze a store-based on-demand crowd delivery system that allows store transfers between crowd drivers. To examine the proposed system, we consider an on-demand shopping and delivery platform in which multiple retail stores sell their products, and the platform organizes crowd drivers of the member stores to make home deliveries. To handle the continuous delivery task and crowd driver arrivals, we propose an event-based rolling horizon framework and introduce two different dispatching strategies that take store transfers explicitly into consideration.

We present that the integrated system leads to not only significant cost saving but also in increase in the number of tasks served by crowd drivers. The total delivery cost savings reach up to 39% in hindsight setting and 9.1% in real-time setting. Furthermore, we observe that the benefits of store transfers is more for less dense systems, and the cost savings from store transfers are significantly more when crowd drivers are tight in terms of time availability. As such, the benefit of store transfers reach its maximum when crowd drivers are less flexible for their detours and the delivery lead times is short. In addition, we show that store transfers reduce on average detour length of crowd drivers, which we believe that increases the convenience of crowd drivers, and ultimately enhances the motivation of willing people to participate crowd-delivery platforms as drivers.

In this paper, we did not consider the future information neither for delivery tasks nor for crowd drivers. Therefore, an integration of future information into this model helps to increase the performance of the real-time implementation. Also, we simplified the back-up vehicle model to better quantify the benefit of store transfers. Thus, an efficient usage of back-up vehicles will increase the efficiency of the delivery platform.

4 Splitting shopping and delivery tasks in an on-demand service

This chapter has appeared as ERIM Working Report Series (Arslan et al., 2019b) and it is currently under review in European Journal of Operational Research.
Co-authors: Niels Agatz and Mathias Klapp

4.1 Introduction

Online retailers continuously seek to offer faster delivery services to satisfy the customers' need for instant gratification. Now, online shopping services such as Amazon Prime Now (Holsenbeck (2018)) offer one or two-hour deliveries in selected US cities.

To compete with online retailers, brick and mortar stores have associated with last-mile logistics providers offer Personal Shopper (PS) services. Such services act as intermediaries receiving online requests placed by customers using a mobile app, who request delivery of items available at affiliated brick and mortar stores. Each customer request specifies a shopping list, a delivery location and a delivery deadline. These services are increasingly popular in the delivery of groceries, since they integrate the convenience of online shopping with product availability at grocery stores.

Grocery delivery platform Instacart is now operating in 20 states across the US and has raised more than $600 million from investors. Postmates, Deliv and Google Express are other examples of this asset-light business model to grocery delivery.

Some personal shopper services provide the customer to schedule the delivery for a specific day and time in advance, and others guarantee delivery within a short lead time, *e.g.* 60-120 minutes. In the later, an automated dispatcher works simultaneously to the request arrival process and chooses whether or not to accept each request. Request rejections can be made implicitly by temporarily not offering the service in specific regions. All accepted requests must be purchased at specific stores and delivered to the customer on time. To perform this operation, each accepted request is assigned to a *personal shopper* within an available fleet. A shopper is a delivery driver who, besides performing pickup and delivery operations, also executes shopping operations, *e.g.* parking, walking, going through the store, purchasing and collecting the items.

The personal shopper business model is somewhat similar to on-demand meal delivery services such as Grubhub or Uber Eats. Personal shoppers, however, also execute shopping operations, and each shopping request can involve collecting items from multiple stores. Google Express, for example, has signed on over 50 merchants including Costco, Target and Walgreens.

To simplify planning, personal shopper service providers may operate a sequential delivery policy, in which shoppers serve a single customer request at a time. However, it might be operationally advantageous to combine multiple requests involving common shopping locations in one shopper trip to prorate fixed store shopping times. Unfortunately, a delivery service with such a tight delivery deadline offers limited consolidation opportunities.

In this paper, we explore different operating strategies for personal shopper services. In particular, we study the benefits of splitting the service of a customer request involving shopping at multiple stores into separate tasks served by different shoppers. Also, we study the additional consolidation opportunities that arise from splitting requests.

More shopper scheduling options are available when requests are split into tasks served by different shoppers. This may help to increase the fleet's time and capacity utilization, which is particularly relevant when shoppers have limited capacity and tight delivery deadlines. We refer to this improvement as potential *packing benefit*. Figure 4.1 illustrates an example in which a customer request r demands delivery

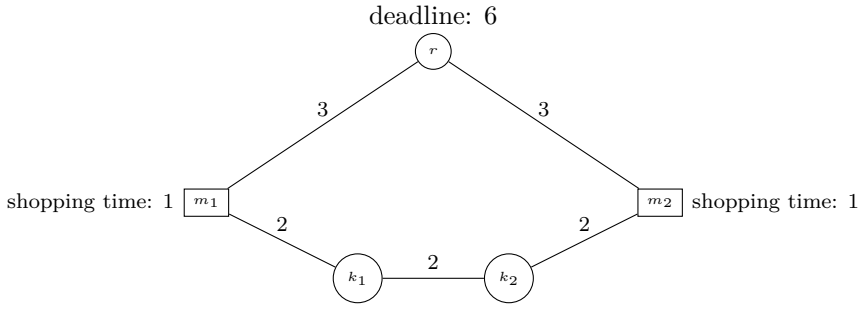


Figure 4.1: Example of a personal shopper service instance

from two stores m_1 and m_2. Two shoppers are located at k_1 and k_2. Travel and shopping times are displayed above the arcs and store nodes, respectively. Suppose that service must occur within six time units. An on-time service is infeasible when one shopper must serve the request alone. However, if the request is split into two separate store-tasks, then both shoppers can deliver to the customer by $t = 6$.

When splitting is allowed, the platform has a larger set of feasible routing options; we call it the potential *routing benefit* of splitting requests. The example in Figure 4.1 shows routing time reductions as a single shopper must travel at least 11 time units to serve r, while two shoppers spend 10 time units. Also, one could save shopping time if multiple tasks originating from a common store are assigned to a single shopper. However, consolidation opportunities are not limited to splitting requests.

In this study, we introduce the Personal Shopper Problem (PSP), which models an online shopping service dynamically receiving, accepting and serving same-day delivery requests. The objective of the PSP is to maximize the number of requests served on-time subject to a limited fleet of shoppers. All accepted requests must be assigned to shoppers and fully served before a pre-established delivery deadline.

Our main contributions are summarized as follows: (i) We identify three types of potential benefits gained by splitting customer delivery requests between different shoppers in a Personal Shopper service: packing, routing and shopping benefits. (ii) We develop an efficient solution to an online personal shopper optimization problem, which integrates request acceptance, order splitting, task to shopper assignment, and routing problems. (iii) We assess our solution quality and confirm

the benefits of splitting requests with computational experiments over a different instance settings.

The remainder of this article is organized as follows. A literature review is presented Section 4.2. We describe the PSP in Section 4.3 and propose a solution for it in Section 4.4. Finally, Section 4.5 outlines results of a computational study, and we conclude in Section 4.6.

4.2 Literature Review

The Personal Shopper Problem (PSP) may be classified as a pick-up and delivery routing problem (PDP). The general PDP aims to find minimum costs routes to serve a set of transportation requests, each with known origin and destination points (Savelsbergh & Sol, 1995). The PSP generalizes the PDP, as a single transportation request can have multiple pick-up locations, *i.e.* stores.

When request splitting is allowed, a personal shopper operation also shares some features with multi-source fulfillment problems and multi-depot delivery problems, in which different parts of a request can be fulfilled from different locations, see Acimovic & Graves (2014); Renaud et al. (1996); Xu et al. (2009). The option of splitting a customer request in multiple deliveries is also studied in Archetti & Speranza (2008), who formulate The Split Delivery VRP. Archetti et al. (2008) shows that splitting deliveries can be beneficial when vehicle capacity is restrictive. Similar benefits are shown for Pickup and Delivery problems in Nowak et al. (2008, 2009).

A personal shopper operation is also *dynamic* as new requests continuously arrive over time; see (Pillac et al., 2013) for a survey on dynamic vehicle routing problems. Recent work on dynamic routing problems has focused on same-day and on-demand delivery services, *e.g.* Arslan et al. (2019a); Klapp et al. (2018b,a); Voccia et al. (2017). Indeed, a PSP is a SDD problem with multiple pickup locations.

The PSP is similar to the dynamic meal delivery problem (DMDP) studied in Reyes et al. (2018); Ulmer et al. (2017); Yildiz & Savelsbergh (2017). Compared to a meal delivery service, personal shopper operations work with drivers who

also pick and purchase the items in the store. As such, personal shoppers spend more time at the pickup location than in a typically restaurant delivery setting. Food freshness requirements in meal delivery also impose limited dispatch-to-door times,i.e., 10 minutes or even less, and these problems are relatively more operationally constrained ending up in simpler routing problems. This difference made by the particular features of meal delivery, also creates structural differences between both problems' solutions. Typically, shoppers in a PSP solution pick-up and carry more tasks than in DMDP. In this sense, meal delivery is relatively more operationally constrained and produces a simpler routing problem.

Recently, Steever et al. (2019) considered a dynamic meal delivery problem with potential spitting of requests from multiple restaurants. Different from our paper, they focus on a setting with soft time windows without in-store shopping related to the request pickups. Their experiments suggest that splitting is not that beneficial in their specific setting.

4.3 Problem Description

In this section, we formally define the Personal Shopper Problem that considers an online platform that dynamically decides which requests to accept and how to best serve the accepted requests. First, we describe the PSP inputs and notation and then we formulate it as a sequential decision model.

4.3.1 Problem Inputs, Notation, and Decisions

We consider a platform that dynamically receives n_R shopping requests throughout a service period $[0, T]$. Each request $r \in \{1, \ldots, n_R\}$ arrives at time e_r and is composed of a set of tasks S_r. Each task $s^r \in S_r$ requires shopping a basket of items at a specific store m_{s^r} and, later, deliver it to the request's destination d_r before $l_r := e_r + L$, a delivery deadline offered to the customer depending on the service lead-time L, *e.g.*, 60 or 120 minutes. Let M and N^R represent the set of relevant stores and customer locations, respectively. Furthermore, let S be the set of all tasks; *i.e.*, $S = \cup_{r \in R} S_r$.

To execute shopping and delivery operations, the personal shopper platform employs a homogeneous fleet $K = \{1, \cdots, n_K\}$ of *shoppers*, each with a capacity of Q tasks. Each shopper $k \in K$ starts and ends its daily operation at its personal location $i \in N^K$, *i.e.*, shopper's k depot.

A shopper visiting a store $m \in M$ spends a time c_s^p to pick up each task s and a fixed time c_m^f in activities such as parking, walking, going through the store, purchasing and collecting the items. Therefore, the shopping time required at store m to collect a set S_m is

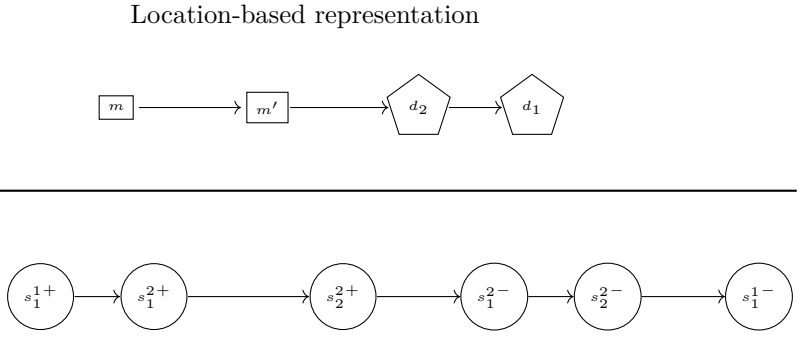
$$c(S_m) := \alpha \cdot c_m^f + (1 - \alpha) \cdot \sum_{s \in S_m} c_s^p, \tag{4.1}$$

where $\alpha \in [0, 1]$ is a parameter representing the relative weight of fixed shopping times.

It is possible to model the PSP on a graph where its nodes represent physical locations, *i.e.*, customers, stores and shopper depots, while its arcs act as the movements between two locations. However, a store location should be associated to a number of different tasks and requests and a physical model makes it difficult to set task-dependent shopping times. Therefore, we instead consider a task-based graph, in which two nodes ${s_m^r}^+$ and ${s_m^r}^-$ are created for each task s^r representing a pickup at store m_s and a delivery to d_r, respectively. As a result, we model the PSP on a graph $G = (N, A)$, where the set of nodes N is the union of these task-based nodes over S and shopper depots N^K. Figure 4.2 illustrates the difference between a location-based and a task-based graph with an example with one shopper, two stores and two requests. Request 1 consists of one task that involves shopping at store m and delivering to customer d_1. Request 2 consists of two tasks, requiring shopping from store m and m' respectively and delivering to customer d_2. The path displayed above the line, physically represents a shopper trip visiting stores (rectangles) and customer locations (pentagon). Below the line, that shopper trip is represented in a task-based graph, where each task shopping operation and each task delivery is represented by a node. Here, s_1^{1+} represents shopping task 1 within request 1 from store m, s_1^{2+} represents shopping task 1 within request 2 from store m, and s_1^{2+} represents shopping task 2 within request 2 from store m'. Analogously, s_1^{2-}, s_2^{2-} and s_1^{1-} represent task deliveries to customer locations.

The set of arcs A in graph G connect each task-based node $i \in N$ to another node $j \in N$. We define c_{ij} as the time it takes to move from node i to node j. If i and j relate to different locations, then $c_{i,j}$ includes the travel time between both locations. Also, if j is associated with a store, then $c_{i,j}$ also includes shopping times at that location. Appendix Section 4.7.1 presents the formal definition of arc cost.

Figure 4.2: Location-based and task-based graphs for a particular shopper's route. Pentagons and squares indicate customer and store locations, respectively. Circles represent shopping and delivery nodes.



Upon each request's arrival, the platform decides whether or not to accept the request for service. In this study, we assume that the platform has no prior nor probabilistic information regarding future requests. Therefore, we always accept a customer request as long as it is feasible to serve the resulting set of pending tasks after the acceptance decision. As the platform searches to feasibly serve all tasks related to the newly arriving request, it has to simultaneously (re)assign tasks to shoppers, including tasks within the same request to different shoppers, and (re)sequence planned shopper routes. If a feasible service option is found, the platform accepts it and continues with the operation. In the next section, we present a sequential decision model to efficiently plan these operations.

4.3.2 Problem Formulation

To make acceptance decisions, the platform must keep track of the accepted but not yet served (active) tasks. Let S_e^a be the set of active tasks at time $e \in [0, T]$. Moreover, at anytime e in the service period, the platform maintains a delivery plan $\sigma_e = \{\tau_1, \cdots, \tau_{n_K}\}$ that includes planned trips for each shopper $k \in K$ to cover all tasks in S_e^a. In our task-based graph G, each trip $\tau_k = \{i_0, i_1, \cdots, i_h\}$ is a sequence of shopping and delivery nodes, and i_0 indicates its shopper's position at time e or the next position if the shopper is en-route. Each trip τ_k must be feasible and, therefore, must meet capacity Q and deliver all its tasks on-time. Also, each task within the trip should be shopped at the corresponding store before delivery.

At time e, the platform also tracks the status of each shopper $k \in K$ defined by its current position ω_k, its earliest departure time from that node $e_k \geq e$, which is strictly later then e if the shopper is en route, and its current load $S_k \subset S_e^a : |S_k| \leq Q$ of already collected but not yet delivered tasks. As we do not allow transfers between shoppers, these tasks have to be delivered by shopper k. Let $\Omega_e = \{(\omega_k, e_k, S_k) : k \in K\}$ be the status of all shoppers at time $e \in [0, T]$.

Now, we formulate the PSP as a sequential decision model defining the system's states, decision epochs, actions, rewards and its objective function (see Puterman (2014)). The set of decision epochs is defined by all arrival times $\{e_r : r \in \{1, \ldots, n_R\}\}$. At each decision epoch $e = e_r$, the state of the platform is $z_e = (S_e^a, \sigma_e, \Omega_e)$. The action in state z_{e_r} is to choose a delivery plan within the collection of plans that accommodate tasks S_r and S_e^a, and the action space is defined by $\mathcal{A}(z_r)$. If a feasible delivery plan exists, then the platform receives an immediate reward, *i.e.*, $I_{e_r}(z_{e_r}) = 1$. Else, we set $I_{e_r}(z_{e_r}) = 0$ when $|\mathcal{A}(z_r)| = \emptyset$. Define V_{e_r} as the cumulative reward at decision epoch r as follows;

$$V_{e_r} = V_{e_{r-1}} + I_{e_r}(z_{e_r}). \tag{4.2}$$

The platform's objective is to maximize the number of served requests by the end of the operation;

$$\max V_T = V_{e_{n_R}}. \tag{4.3}$$

4.4 Solution Approach

Now, we describe our solution approach to the PSP problem.

Equation (4.3) refers to an online optimization problem. An optimal deterministic solution to it may be infeasible, since it requires perfect knowledge of future request arrivals. Instead, we focus on finding a good online policy. We regard this problem as a dynamic problem with no future information and propose the following rolling horizon framework.

4.4.1 Rolling Horizon Framework

To deal with the continuous arrival of requests, we use an event-based rolling horizon framework (RH) that solves a deterministic optimization problem each time a new request arrives. Based on the solution to this optimization problem, we decide whether or not to accept the new request and update the delivery plan accordingly. This approach is commonly used in the literature to model these type of systems, see Srour et al. (2016); Arslan et al. (2019a). Specifically, we solve at each decision epoch a *Pickup and Split Delivery Problem with Deadlines* (PsDPd) that aims to identify a feasible delivery plan covering both newly arrived and active tasks. If this plan is found, then the platform accepts the new request. Instead of solving a feasibility problem, we set the objective function to minimize the total duration of all shopper trips to increase shopper's utilization and eventually increase future acceptance capacity.

Algorithm 4 describes the RH procedure. Upon the arrival of request $r \in R$, a routine *Screentheplan()* computes all shoppers' status and a PsDPd is solved with status Ω_{e_r}, active tasks $S^a_{e_r}$ and newly arrived tasks S_r. Function *PsDPd* returns either a delivery plan σ^* or NULL when it is infeasible to serve r. If a solution is found, we increase the reward by 1 and update the delivery plan. Otherwise, we keep the previous delivery plan and reject r.

The routine *Execute()* carries out the delivery plan until the arrival of request $r+1$. This means shoppers follow the sequences of nodes as stated in their trips. A shopper departs from a node immediately when the corresponding operation on

Algorithm 4 Rolling Horizon Framework

Input:
$V \leftarrow 0$, $\sigma = \{\tau_k := \{\omega_k\} : k \in K\}$
Iterations:
while $e_r \leq T$ **do**
    $(S^a_{e_r}, \Omega_{e_r}) \leftarrow$*Screentheplan(*$\sigma$*)*
    $\sigma^* \leftarrow PsDPd(\Omega_{e_r}, S^a_{e_r}, S_r)$
    if $(\sigma^* \neq$ NULL$)$ **then**
        $V \leftarrow V + 1$
        $\sigma \leftarrow \sigma^*$
    end if
    Execute(σ*)*
    $r \leftarrow r + 1$
end while
Output: V

the node is over. A shopper can be located either somewhere on his or her current trip (*i.e.*, busy) or at the final destination of the last completed trip (*i.e.*, idle). All shoppers return to their depots after the service period is over and all assigned tasks are served.

4.4.2 Deterministic Pickup and Split Delivery Problem with Deadlines

At each decision epoch, we solve a deterministic routing problem where customer requests can be split and served by multiple drivers in parallel. We refer to this as the Pickup and Split Delivery Problem with Deadlines (PsDPd)

Let PsDPd$(\Omega_{e_r}, S^a_{e_r}, S_r)$ be the problem solved with the arrival of request r. For the sake of convenience, let's assume that $S^a = S^a_{e_r} \cup S_r$. The PsDPd seeks a set of feasible shopper trips serving all tasks in S^a while minimizing the sum of all trip durations.

In a task-based graph, the PsDPd can be formulated as a Pickup and Delivery Problem with Time Windows (PDPTW). However, we may exploit the fact that we may be able to consolidate shopping operations of multiple requests at a single store. Observation 1 uses this idea to reduce the search space.

Observation 1. *Let* $\tau = \{i_0^-, i_1^+, X, i_1^-, \cdots\}$ *and* $\tau^{'} = \{i_0^-, i_1^+, X^{'}, i_1^-, \cdots\}$ *be trips such that both have the same node sequences except partial sequences* X *and* $X^{'}$*. Also, let* X *and* $X^{'}$ *be the two different permutations of a set of nodes such that each element of the set maps to same location. Then, two trips* τ *and* $\tau^{'}$ *are identical in terms of trip duration.*

4.4.2.1 Exact Approach

This section describes an exact approach to solve the PsDPd that partitions all active tasks among the fleet of shoppers and solves the individual shopper routing problems to optimality. Let $p = \{P_1, \cdots, P_{n_K}\}$ be a partition of the set of active tasks S^a, where $P_k \subset S^a$ is the set of tasks assigned to shopper k. Let $\mathbb{P}$ be the collection of all feasible partitions. The partition p is feasible if each shopper $k \in K$ has a feasible trip covering P_k.

To find an optimal delivery plan, we enumerate all feasible partitions $\mathbb{P}$ and determine the optimal delivery plan for each one as follows. For a given set of shopper tasks P_k, we can find the trip for shopper k that serves all tasks while minimizing the total travel time by solving a Hamiltonian Path Problem with Precedence Constraints and Deadlines. We can certify that the set of optimal trips for all shoppers in a partition p forms an optimal delivery plan for a given partition p. To identify which partition minimizes the total service time, we explore all partitions sequentially. To do so, we store the best partition (delivery plan) and the associated objective value as an incumbent plan as an upper-bound value. For each unexplored partition, we start solving Hamiltonian path problems for each shopper. If a lower bound to the minimum feasible duration of all shoppers' routes exceeds the incumbent, then we stop and discard the partition. If the solution takes less time than the current best solution, we update the incumbent plan. The approach terminates when all partitions are checked. The final incumbent solution is the delivery plan that minimizes the total service time.

Each Hamiltonian Path Problem is solved by a forward labeling algorithm, similar to the one proposed by Tilk & Irnich (2016). In this method, partial shopper trips are constructed and recursively extended via a resource extension function. Let x_i be a partial trip from a source node; *e.g.*, shopper's k current location, to a node

$i \in P_k$ with label (X, c, i), where $X \subset P_k$ is the ordered subset of visited nodes in the partial trip, c is the trip's duration and i is the last node in x_i. The initial label for shopper k is $(X = \{\omega_k\}, 0, \omega_k)$. A label extends to node $j \in P_k$ and produces a new label $(X \cup \{j\}, c, j)$ with the following REFs: $c_j = REF_{ij}(c_i) := c_i + t_{ij}$. An extended (partial) trip can be infeasible, feasible but dominated, or non-dominated. We eliminate all trips that are dominated.

Algorithm 5 describes the forward labeling algorithm for shopper k over tasks P_k. Let $\mathcal{X}_l$ be collection of partial trips with length $l = |X| - 1$. Subroutine $FeasibilityCheck()$ evaluates whether or not an extension of the partial path to node j meets delivery deadlines. It is feasible to extend a partial trip if it is possible to reach node j before the deadline. Routine $BoundingCheck()$ verifies if the total duration of the partial plan for Partition P_k exceeds the incumbent's solution, and if so, stops the algorithm and prunes the partition.

Algorithm 5 Forward labeling algorithm

```
1: Set X_0 = {X = {ω_k}, 0, ω_k}
2: for l = 1 ··· |P_k| − 2 do
3:     for x ∈ X_l do
4:         for j ∈ P_k, j ∉ X do c_j = REF_ij(c_i)
5:             if FeasibilityCheck(c_j) then
6:                 if BoundingCheck(c_j) then
7:                     Add x_j to X_{l+1}
8:                 end if
9:             end if
10:        end for
11:    end for
12:    DominanceRule(X_{l+1}),
13: end for
14: Find a path x with label (P_k, c, ·) such that c is minimal.
```

The dominance rule eliminates partial trips according to the following rule:

Definition 1. *Let x_i and x_i' be two partial trips for the same shopper with labels (X, c, i) and (X', c', i) with common last node i. Partial trip x_i dominates $x_i^{'}$ if $X^{'} \subset X$ and $c < c^{'}$.*

Furthermore, **Observation 1** allows us to eliminate redundant partial paths by forcing a lexicographical order for nodes corresponding to the same location without loss of generality.

4.4.2.2 A Heuristic for the PsDPd-*Planmaker()*

Solving the PsDPd exactly is hard in large instances. Therefore, we propose a adaptive large neighbourhood search (ALNS) heuristic called *Planmaker()* similar to the one proposed by Pisinger & Ropke (2010).

When a request r arrives at time e_r, we try inserting all its tasks into the delivery plan σ. For each unassigned task $s \in S_r$, the procedure determines the cheapest insertion for its pickup (s^+) and delivery tasks (s^-) simultaneously examining all possible options in each shopper trip. Then, the cheapest unassigned task is inserted into the plan. This process continues until there is no unassigned task left to insert (or when it is not feasible to do so).

Later, we run ALNS over neighborhoods defined by removal and repair operators. Removal operators destroy the solution to a predefined level by removing certain tasks. Repair operators reinsert the removed tasks into the delivery plan.

Destroy Operators. For a number of n_s^a tasks and n_r^a requests in the incumbent solution, we set a removal ratio $\rho \in [0, 1]$ and, depending on the following five operators we remove $q_s = \rho \cdot n_s^a$ tasks or $q_r = \rho \cdot n_r^a$ requests.

- *Random task removal.* Randomly remove q_s tasks.
- *Random request removal.* Randomly remove q_r requests.
- *Most time-consuming task removal.* Remove the q_s most time-consuming tasks from the current solution. The time-consumption of a task is defined as the time savings generated if the task is ejected from the route.
- *Most time-consuming request removal.* Remove the q_r most time-consuming requests.
- *Shaw Removal.* Randomly remove a task s, and then also remove the $q_s - 1$ closest tasks to s in terms of the euclidean distance.

Repair Operators Once tasks are removed, they are sequentially inserted back into the delivery plan based on the following operators:

- *Cheapest insertion*: Insertion sequence goes in increasing order of insertion cost.
- *Non-split insertion:* All removed tasks from the same request are inserted into the tour of one shopper.
- *Regret insertion.* Tasks are inserted in decreasing order of regret, defined as the difference between the cheapest and the second cheapest cost.

Finally, we choose the repair operators according to weights, which are updated based on the success of each operator.

4.5 Computational Study

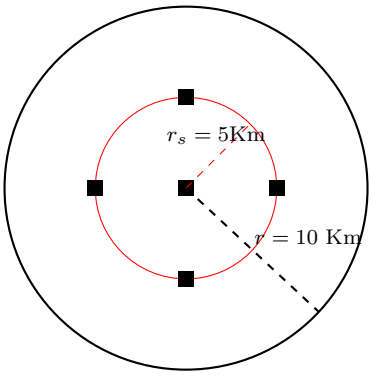
In this section, we present an extensive set of computational experiments to assess the quality of our solution approach and explore the potential benefits of different operational strategies for the Personal Shopper Service. Next, we describe the experimental setting and instances for the base-case experiments.

4.5.1 Experimental Setup

All our experiments use a circular service area with a 10km radius in which the request destinations are uniformly distributed. We consider five stores in the service region: one in the center, and four located on an inner circle of radius $r_s = 5$km at equal intervals (see Figure 4.3).

The base case experiment considers instances with three shoppers and 80 request arrivals within a ten hour service period. We consider exponential request inter-arrival times and, thus, realizations are uniformly distributed within the service period. However, this probabilistic information is assumed unknown to the platform. Also, we assume that each request requires to shop three tasks, each at a different store randomly selected from the five stores available. To simplify the interpretation

Figure 4.3: Layout of the service region used in our experiments



of the results, we also assume that each shopping task consists of the same number of items with the same weight and volume, and that each shopper has capacity $Q = 10$ tasks. Shoppers travel at a speed of 30 km/h. We set α to 0.9 and set all shopping time values equal to ten minutes, meaning that a shopper spends nine minutes for each store visit, and one minute to pick up a task within a store. Table 4.1 summarizes all parameter values used in the base case experiment.

Table 4.1: Base Case Parameters

No. of shoppers	3
Shopper speed	30 km/h
Shopping time coefficients:	
c^f	10 min
c^p	10 min
α	0.9
Deadline	90 min
Shopper capacity	10 tasks

4.5.2 PsDPd Heuristic Validation

Now, we evaluate the performance of our *Planmaker()* heuristic compared to the exact solution of the PsDPd. We collect an instance set for the PsDPd by executing the RH framework over the online problem as described in Section Section 4.4.2.

To solve the deterministic snapshot problems to optimality, we consider a small instance with 40 request arrivals and two shoppers in this validation process. We draw five instances for each sub-problem with between 8 and 15 active tasks.

Table 4.2 presents the average percentage difference in total service time between the heuristic and the exact approach, and the number of times an optimal solution is found by the heuristic. However, more importantly, we show the number of times that PlanMaker() fails to find a feasible solution while the exact approach does find one.

Table 4.2: Comparisson betwween exact and heuristic solutions for the PsDPd

n_s	Av. Opt. gap (%)	# Optimal	△ Feasibility
8	1.3	4/5	0
9	1.5	4/5	0
10	4.0	3/5	0
11	0.8	4/5	0
12	2.1	3/5	0
13	2.8	4/5	0
14	3.7	1/5	0
15	3.1[1]	0/5	1

The results show that the heuristic performs well with an average optimallity gap of less than 4.0%. Although it does not always finds the optimal solution, PlanMaker() identifies a feasible solution and, thus, makes a correct acceptance decision in all but one case.

4.5.3 Operating Policies

For each simulation, we test three different policies computing performance metrics in each one:

> One by One ($1b1$): Each shopper serves one single customer request at a time and cannot start serving a new request before delivering the previous one.

[1] Average over four instances. Our heuristic did not find a feasible solution for one instance

Consolidation (C): A shopper can simultaneously serve multiple requests, *e.g.*, shopping for one request and then delivering another. In this policy, all tasks of a single request are served by one single shopper.

Consolidation & Splitting ($C\&S$): A request can be split into different store tasks that can be served by multiple shoppers in parallel. As in the consolidation strategy, shoppers can simultaneously serve tasks of multiple requests.

4.5.4 Base Case Results

Table 4.3 presents the results for the different policies averaged over 99 random scenarios of request arrivals. We report the following performance indicators: (i) *Served requests*: the number of served requests as a percentage of the total number of requests. (ii) *Requests split*: the number of split requests as a percentage of total served requests. (iii) *Time per request*: The total time worked by all shoppers divided by the number of requests served. We further break this value down in two parts: *Shopping time* and *Travel time* per request. (iv) *Click to door (CtD)*: the average time between a request's arrival and the delivery of its latest task. (v) *Delivery interval*: the average time between the delivery of the first and the last task of a specific request. By definition, the interval is equal to zero for requests that are not split. (vi) *Number of locations visited per request*: The total number of visited locations (stores and customers) over all shoppers divided by the number of requests served.

Table 4.3: Average base case results, **80 Requests, L: 90 minutes, 3 tasks per request, n_K: 3 shoppers**

	$1b1$	C	$C\&S$
Served requests (%)	45.8	77.3	88.1
Request split. (%)	0	0	69.2
Delivery interval (min.)	0	0	23.6
Time per req (min.).	51.2	28.9	25.1
Shopping time per req (min.)	30.0	15.0	10.3
Travel time per req (min.).	21.2	13.9	14.8
# locations visited per req.	4.0	2.3	2.7
CtD (min.)	77.1	78.2	77.3

Our results show that substantial performance improvements can be obtained by consolidating requests. Comparing the consolidation policy C to the $1b1$ policy, the average numbers of served requests increases from 45.8% to 77.3%. This improvement is associated with a reduction in the fulfillment time per request from 51.2 to 28.9 minutes, both in terms of shopping and travel times.

An additional performance increase of 15% is possible when request splitting is allowed, *i.e.*, $C\&S$. Splitting enables more consolidation opportunities by reducing granularity and increasing the *packing benefit*. Moreover, we also observe a reduction in the service time per request, which indicates extra *shopping benefits*. Conversely, there is a slight increase in travel time per request, *i.e.*, from 2.3 to 2.7. This could relate to the fact that splitting means that a request destination is visited multiple times. Overall, the travel time per location visited decreases from 6.0 (13.9/2.3) to 5.4 (14.8/2.7) which indicates that there are *routing benefits*.

When we allow splitting, we see that 69.2% of the accepted requests are split and thus served by more than one shopper. The average time between the first partial delivery and the final partial delivery, *i.e.*, the delivery interval, is 23.5 minutes while the click-to-door (CtD) time is between 77 and 78 minutes for each of the different policies[2]. This suggests that the first partial deliveries occur earlier than the non-split deliveries.

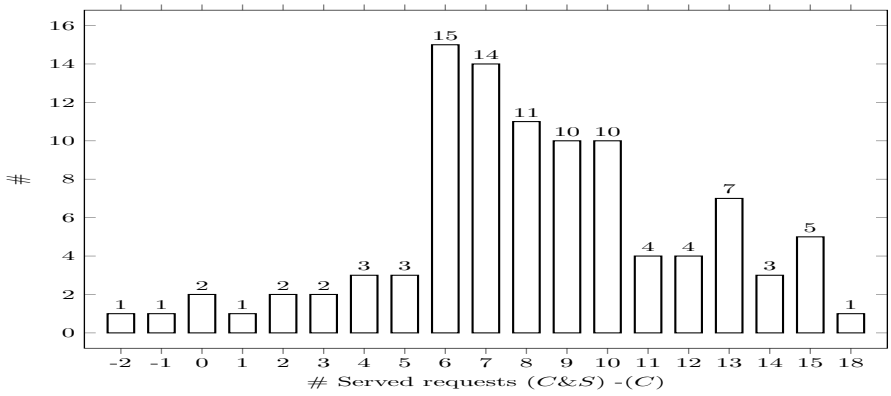
Figure 4.4 presents an histogram of the differences in the number of requests served for policies $C\&S$ and C over all simulated scenarios. We see that the number of accepted customer is no smaller for $C\&S$ than C in all but two instances.

4.5.5 Impact of the Number of Stores per Request

Now, we vary the number of tasks per request between two and five. This also varies the number of different stores involved per request. To allow for comparison, the same stream of requests is used in each of the different scenarios. The only difference is the number of tasks per request. To keep the average number of tasks per shopper the same in the different scenarios, we adapt the number of shoppers accordingly.

[2]See Appendix Section 4.7.2 for more detailed analysis

Figure 4.4: Impact of Request Splits: Distribution of Requests Served over 99 Simulated Instances; **80 Requests, L: 90 minutes, 3 tasks per request, n_K: 3 shoppers**



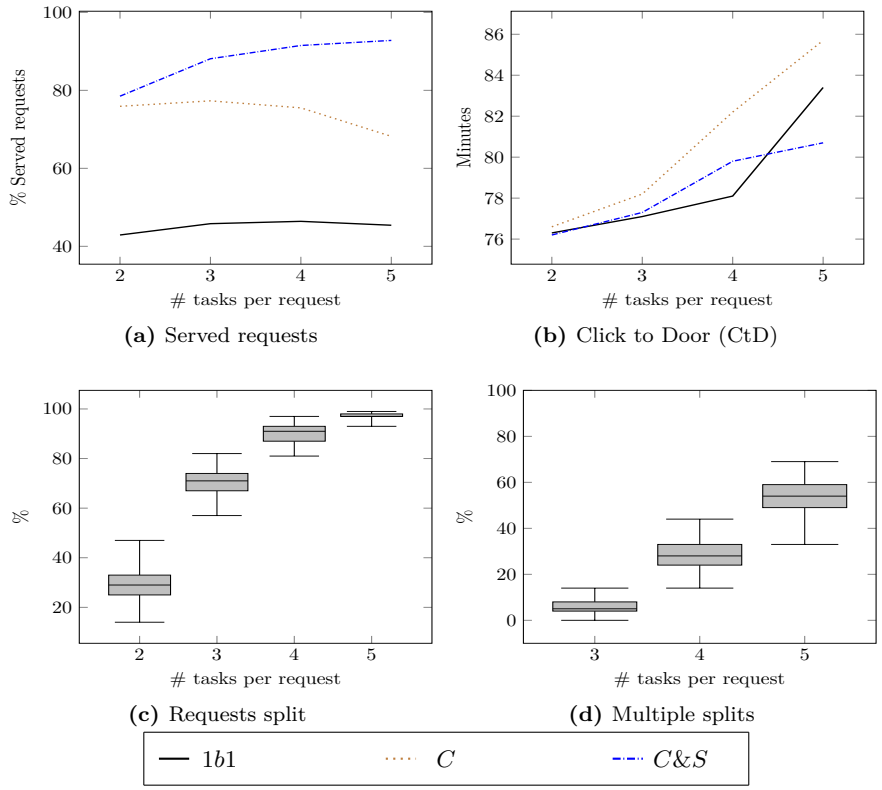
Our results are summarized in Figure 4.5. Figure 4.6a presents the percentage of served requests for the different numbers of stores (tasks) per request and policy. For the consolidation policy (C), we see that the performance deteriorates with more tasks per request. One potential reason for this observation is that it is more difficult to fit larger requests into the shoppers' time schedules. Moreover, it is also more difficult to combine multiple larger request in one shopper to exploit the economies of scale in shopping. However, the consolidating policy (C) consistently outperforms the simple $1b1$ policy, even for requests that involve five stores.

In contrast to policy (C), we see that the performance of the $C\&S$ policy improves with the number of stores per request. The reason for this is that this policy assigns tasks (instead of requests) to shoppers. This fact facilitates that shoppers can collect more tasks at store visits and also traverse fewer stores before delivery locations with policy $C\&S$ comparing to C.

Figure 4.6b reports the average click-to-door (CtD) times for the different numbers of tasks per request. As expected, the CtD time increases with the number of tasks per request. This is because each store visit involves travel time to the store and shopping time in the store. We observe that the policies without splitting are more sensitive to variations in the number of stores per request. This implies that splitting also reduces the fulfillment time per customer.

The box plots in Figure 4.6c and Figure 4.6d provide more insight into the solutions for the $C\&S$ policy. Figure 4.6c shows that the number of requests with at least one split increases with the number of tasks per request. Figure 4.6d show the same for the number of requests with at least two splits. These results are intuitive as the number of splitting options increases with the number of tasks. We see that in the instances with five tasks per request almost all requests are split at least once.

Figure 4.5: Impact of Request Size, **80 Requests,** $\alpha : 0.9$**, L: 90 minutes**



(a) Served requests **(b)** Click to Door (CtD)

(c) Requests split **(d)** Multiple splits

4.5.6 Impact of Shopping Economies of Scale α

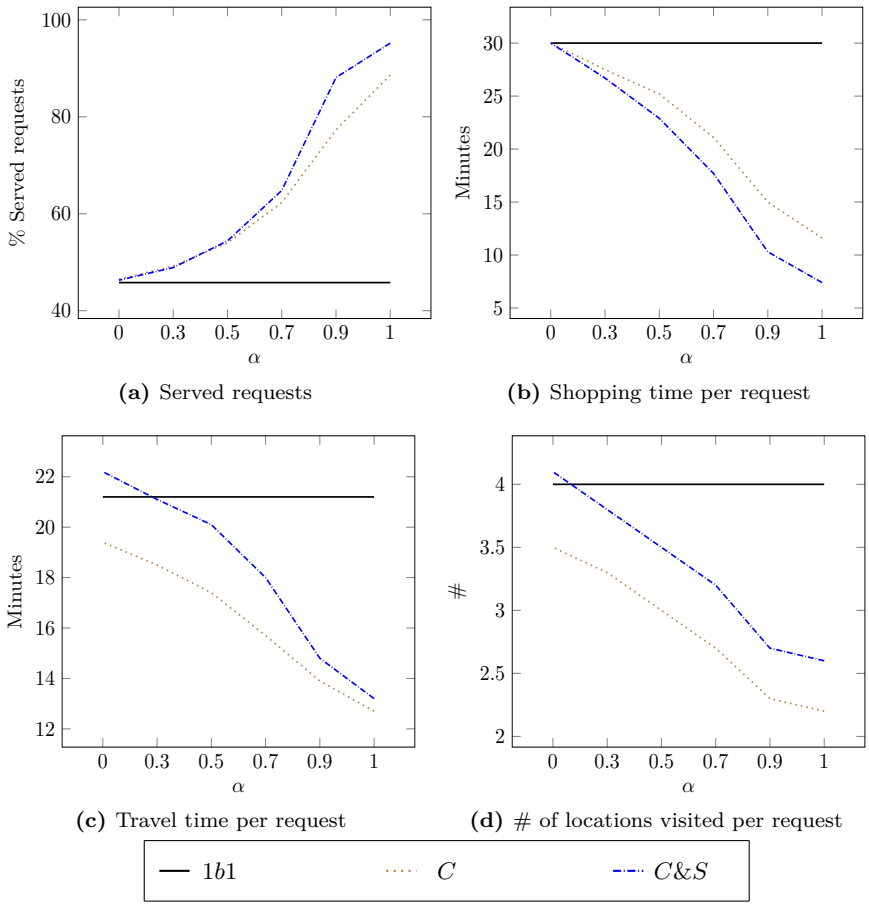
In this section, we study the impact of the relative weight $\alpha \in [0, 1]$ of the fixed part of the shopping time, *i.e.*, 'setup time', on the performance of the different proposed policies. A value of $\alpha = 0$ indicates that there is no setup time and the shopping time at the stores is directly proportional to the number of tasks collected. In this case, there are no economies of scale in the shopping operations. Conversely, a value of $\alpha = 1$ represents a situation in which there is only a setup time and the total shopping times at the store are independent of the number of tasks picked up. In our base experiments, we assume the fixed and the variable part of the shopping time to be equal, so that total shopping time is independent of α when no consolidation occurs.

Figure 6 presents the performance for the different values of α for all policies. Figure 4.7a shows that for policies with consolidation (C and $C\&S$) the percentage of served requests increases with α. This is intuitive as the value of consolidating increases with a higher value of α. As expected, the economies of scale do not effect the performance of the $1b1$ policy as it does not allow consolidations. Comparing C and $C\&S$, we see that the benefits of splitting, *i.e.*, shopping benefits, increase with α. For values $\alpha < 0.5$ there are no benefits as the savings in the shopping time do not offset the additional travel time associated with splitting requests.

Figure 4.7b reports the average shopping time per served request. This illustrates the realized economies of scale in the shopping activities. Without consolidation, each 3-task request involves 30 minutes of shopping. With consolidation (C and $C\&S$), the average shopping time per request decreases as α increases. By simultaneously shopping multiple tasks in a store, it is possible to reduce the shopping time per task; this reduction is higher when requests can be split.

Similarly, we observe in Figure 4.7c and Figure 4.7d that for policies C and $C\&S$ the average travel time and number of locations visited per request reduce as α increases. If there are more incentives for consolidation, then the platform will efficiently pack tasks to reduce the number of nodes visited per request and, thus, travel time. We further observe that the average difference in travel time per request between policies C and $C\&S$ reduce as α increases.

Figure 4.6: Impact of Shopping Consolidation Factor, **80 Requests, L: 90 minutes,** n_K**:3**



(a) Served requests

(b) Shopping time per request

(c) Travel time per request

(d) # of locations visited per request

4.5.7 Impact of the Delivery Deadline on the Packing Benefits

In the previous section, we saw that the consolidation of shopping tasks is an important driver for the benefits of splitting. In this section and the next section, we focus specifically on the packing and routing benefits by setting $\alpha = 0$, which means that there are no economies of scale in the shopping activities.

Figure 4.7: Packing Benefits of Request Split, $\alpha = 0$**, 80 Requests, 3 Shoppers**

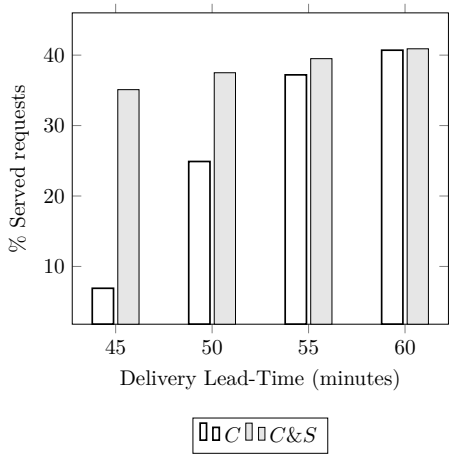
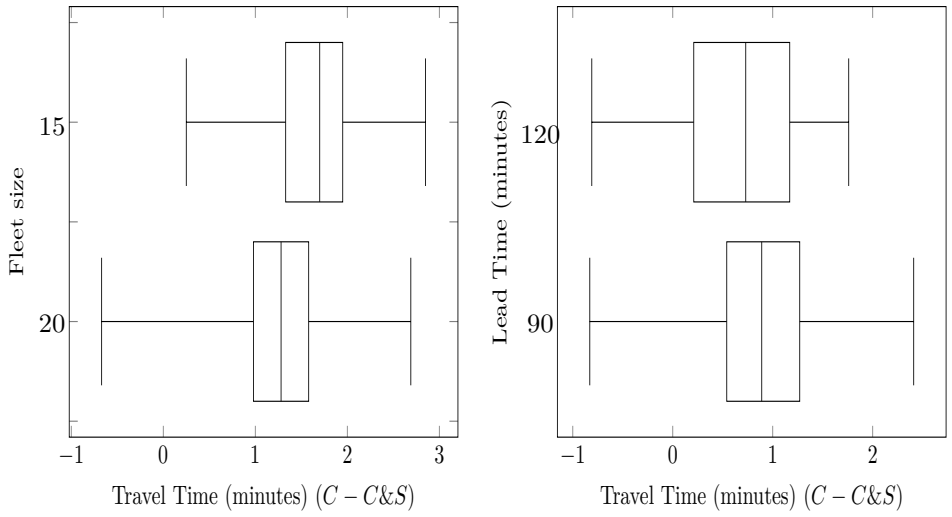


Figure 4.7 presents the average number of requests served for policies C and $C\&S$ for different delivery deadlines ranging from 45 to 60 minutes. We see that the performance of both policies is similar with a 60 minute deadline. However, we see that the policy that allows splits outperforms the consolidation policy when the deadlines become shorter and the instances become more time constrained. That is, the performance of policy C deteriorates significantly with a reduction of the delivery deadline. Request splitting allows for more planning flexibility by creating smaller tasks that can be served in parallel which help to improve the capacity utilization, *i.e.*, the packing benefits. This suggests that request splitting is especially beneficial in systems with short delivery deadlines and tight time constraints.

4.5.8 Routing Benefits

In this section, we focus specifically on the routing benefits of request splitting. Therefore, we consider a setting without shopping benefits ($\alpha = 0$) and no packing benefits, *i.e.*, sufficient capacity. In particular, we increase the number of shoppers so that policy C can serve all requests.

Figure 4.8: Routing Benefits of Request Splits When No Packing Benefits: $\alpha = 0$, **80 Requests**



(a) Lead-Time: 60 mins **(b)** Fleet size: 10 Shoppers

Figure 4.8 presents box-plots for the average difference between the travel time per request in policy C and $C\&S$. We choose two different delivery deadlines (90 and 120 min.) and two different fleet sizes, with 15 and 20 shoppers, respectively. The longer deadlines in combination with the large fleet ensures that all requests can be served in both policies for all the instances.

We see that for all parameter settings, request spliting provides additional routing benefits in most of the instances. Furthermore, we observe that this benefit increases when the fleet size is smaller, or the lead-time is shorter; *i.e.*, the resources are more restricted for a request. The primary reason behind it is that the number of feasible solutions available for policy C are sensitive to the time and fleet restrictions. However, splitting requests allows to partially reassign node visits between shoppers so that the overall delivery plan has more options to minimize travel time.

These results suggest that there are benefits to splitting even in settings with sufficient capacity and time to serve all requests.

4.6 Concluding Remarks

The paper focuses on a personal shopper service that provides same-day delivery from brick and mortar stores. Customers place orders throughout the day requesting delivery from one or more stores. The service provider decides whether or not to accept a request and on how to assign shoppers to request in order to maximize the number of served customer requests. We model the problem as a sequential decision problem and present a rolling horizon approach to solve it. We propose an exact and a heuristic approach to solve the pickup and delivery problems at each optimization interval.

We have conducted an extensive computational study to compare different operating strategies, in terms of number of served request and total travel time. Our results provide the following insights. Consolidating requests increases the number of served requests as compared to a strategy in which all requests are served one by one. Moreover, splitting requests between different shoppers can further enhance the performance of the system. The benefits of splitting increase when the system is more constrained or when there are more economies of scale in shopping.

As this is one of the first papers that studies the personal shopper business model, there are many avenues for future research. One natural extension is to consider a setting in which we have probabilistic information on the customer requests. Here, it would be possible to strategically reserve some delivery options for more profitable future customer requests. Another interesting research avenue involves incorporating both splits and transfers. This means allowing transfers of shopped tasks between shoppers to consolidate delivery operations.

4.7 Appendix

4.7.1 Arc Costs in the Task-Based Graph

In the task-based graph, the cost of an arc consists of multiple components (*e.g.*, parking time, task picking time, and, travel duration between different locations.), as this value depends on the source and sink nodes of the arc.

In equation 4.4, we formally define arc costs on the task-based graph, defined in Section Section 4.3. Let t_{ij} be the average travel time from node i and j. Notice that this value is positive only if these i and j are not located at the same physical point; define the logical operator $i \overset{\text{L}}{\neq} j$ that checks whether i and j are two different physical location.

$$c_{ij} := \mathbb{1}_{\left(i \overset{\text{L}}{\neq} j\right)} t_{ij} + \mathbb{1}_{(j \in S^+)} \left(c_j^p + \mathbb{1}_{\left(i \overset{\text{L}}{\neq} j\right)} c_j^f \right), \tag{4.4}$$

where $\mathbb{1}_{(\cdot)}$ is an indicator function that takes value one if the statement in $(\cdot)$ is true.

4.7.2 Distributions in Base Case for Policy $C\&S$

We provide additional results for the $C\&S$ policy in the base experiments. In Table 4.4, we present the empirical distribution of click-to-door times and delivery intervals. Click-to-door times are displayed for both requests split and non-split in the operation. Delivery intervals are only computed for requests, which were split during service.

These results suggest that click to door times do not significantly change when request splitting is allowed. Table 4.4 also shows that delivery intervals range between a couple of seconds to 85.7 minutes. Nonetheless, the average delivery interval is 23.4 minutes and three out of four cases are less than 34.9 minutes.

Table 4.4: Empirical Distribution of click-to-door time and delivery Interval for the $C\&S$ policy.

	click to door (CtD)			delivery
	non-split	split	All	interval
minimum	15.0	37.8	15.0	0.01
Q1	71.4	72.9	71.7	7.3
Q2	81.7	82.2	81.8	18.3
Q3	87.2	86.8	87.3	34.9
maximum	89.9	89.9	89.9	85.7
average	77.2	78.1	77.3	23.6

5 Conclusions and future outlook

In this thesis, we have explored new last-mile delivery models to examine the challenges in emerging on-demand delivery systems. Most of these express delivery services are introduced recently to online shopping markets by e-commerce players to gain competitive advantages; however, it is very likely that these new business models will be a norm in the near future, and that operational efficiencies will be an important competitive factor. Therefore, we initially identified operational challenges accompanying these services, and then recommend solutions, which make use of sharing principles and connected technologies, to create effective on-demand delivery services.

We mainly focused on two concepts in on-demand delivery systems that aim to utilize *exisiting traffic flows* and *existing retailer infrastructures*. We explored these ideas by introducing crowdsourced delivery, online shopping and delivery platforms. Notably, we investigated these two concepts from an operational point-of-view, assessing their capabilities by applying optimization techniques.

In this concluding chapter, we first summarize the main results of our research, and then discuss several new directions for future research.

5.1 Main results

In the Introduction chapter, we addressed the trends in online shopping for the B2C market, particularly we focus on the operational challenges that arise while aiming for expedited delivery services. We identified that short lead-times enforce logistic providers to reshape their traditional last-mile delivery models. Thus, we proposed systems that exploit the journeys of willing drivers, and retailer stores as fulfillment centers and transshipment points.

In the first two chapters, we examined crowdsourced delivery systems in the on-demand delivery setting to utilize existing traffic flows. To express the dynamic nature of continuous arrivals of delivery requests and crowd drivers, we formulated these problems as variants of dynamic pickup and delivery problems. Within these special variants, we considered the unique characteristics of crowd drivers by specifying their detour and departure flexibilities, willingness to stop, and existing itineraries. Furthermore, to guarantee on-time deliveries, we considered dedicated delivery resources that can fulfill the service in case of crowd driver unavailability. We proposed rolling horizon frameworks, which re-optimize incumbent delivery plans at the arrival of each request or crowd driver. At the core of our solution approach, we constructed feasible delivery plans for each driver and carry these plans over time. We were able to obtain optimal assignment and routing decisions rapidly by using the stored information.

We then assessed benefits of using crowdsourcing in on-demand parcel delivery system on a broad test set of different attributes (*i.e.,* geographies, detours and stop flexibilities of crowd drivers, and dispatching strategies of drivers). We evaluated two leading indicators: the total distance of travel and the number of customers served by crowd drivers.

In Chapter 3, we incorporated the sourcing of existing the retail infrastructures to use stores as transfer points and allowed multiple crowd drivers to deliver a single parcel. In other words, a driver can pass a package to another driver to finish the delivery. The use of stores as transfer points provides the advantage that two drivers are not required to be at the same point simultaneously. Computational experiments present that store transfers lead to significant cost savings and higher service rates by crowd drivers. Equally important, we observed that the average detour duration of crowd drivers who carry parcels reduces with the availability of store transfers and less detour per delivery enhances the convenience of crowd-drivers. Therefore, this chapter demonstrates that the integrated planning of crowd-sourced drivers and existing store locations

Based on these two chapters, we concluded that crowdshipping, or utilizing existing traffic flows, increases the efficiency of on-demand delivery services by decreasing total delivery costs. In particular, the results emprically show us that:

- Crowdsourced delivery is the most beneficial when the origins of delivery requests and crowd drivers' journey are the same.
- The more flexible crowd drivers are, *i.e.,* willing to deviate more or stop more, the more efficient the crowd-delivery system is.
- The crowdshipping savings are higher when the system-wide delivery lead-time is shorter.
- The parcel exchanges, particularly for store transfers, are more beneficial when crowd drivers are less flexible or fewer drivers are available in the system.

These results indicate great potential for using existing traffic flows in the crowd-sourced delivery, which drivers have existing journeys. Mainly, logistic providers reduce their last-mile delivery expenses by designing compensation schemes that give sufficient motivation for willing people to deliver some packages on their way to destinations. Furthermore, we see that even when there are no sufficient crowd drivers in such systems, or crowd drivers are limited in time, we can still make the system work with the help of store transfers.

In Chapter 4, we studied online shopping and delivery platforms to utilize local stores as fullfilment centers. These platforms, similar to well-known restaurant delivery platforms such as UberEats or Grub-hub, play a central role between the end customers and retail grocery stores. They manage a fleet of shoppers who go into stores, collect, purchase, and subsequently deliver customers' orders potentially from multiple locations. In this particular on-demand supply chain design, we explored the benefits of request splitting for the customer requests consisting of shopping at multiple stores. We identified three potential benefits that request splits can provide: *packing benefits*, in which platform can accept more requests due to task granularity, *routing benefits*, in which the platform has more feasible route alternatives for shoppers, and *shopping benefits*, in which the platform organizes the plan such that pickup consolidation can be achieved.

We formulated this problem as a sequential decision-making model. This model provides valuable tools to capture the dynamics of systems, and re-shapes to the ongoing plan at the time of a request arrival. We assume a basic acceptance policy *i.e.,*, each arriving request is accepted as long as shoppers can serve it. We assessed

the benefits of request splits in different sizes of requests, factors of shopping consolidation advantage, and delivery lead times.

The results show that shoppers' utilization increases with request splits, by effectively managing delivery fleet resources. We also observed the following results from the sensitivity analysis:

- The benefits of request splits are greater when customer requests require more stores shopping, or when the delivery lead time is shorter.
- Split benefits correlate highly with the consolidation advantage at stores, *e.g.,* if a store stop has considerably long parking time, it is better to collect more orders as from the store when a shopper arrives.

From the managerial perspective, these results present several insights: First, we see that platforms have gained a surplus from request splits; and hence they have a option to share a part of the excess saving with customers, which may incentivize them to use this service. Second, the platforms can be more strategic to distribute these incentives on specific customer segments, such as customers whose orders consist of shopping in a store with long parking times. Last, these platforms offer a faster service to customers, particularly whose requests contain multiple stores shopping.

5.2 Future outlook

This dissertation explored the potential of using existing traffic flows and existing retail stores as forms of crowdsourcing, online shopping and delivery platforms in the on-demand delivery services. These results reveal not only valuable insights, but also point several exciting aspects to examine. In this section, we discuss possible extensions for the problems introduced in this thesis and other topics for future research. At the end, we briefly discuss trends in the on-demand delivery, particularly within the context of crowdshipping, online shopping and delivery platforms.

Crowdsourced workforce is becoming the major resource of the on-demand delivery services. Potentially, the crowd workforce integration will expand to other services

in last-mile delivery models in the near future, mainly because of its flexibility and minimal capital requirement, which other technological advances such as drones cannot easily provide. Therefore, our analysis in Chapters 2 and 3 provide crucial insights into crowdsourced delivery systems and prescribe efficient operational decision support systems.

One can, however, observe that the current practice of the crowdsourced delivery differs slightly from the focus in this dissertation. Crowd drivers often join platforms to earn extra money in their free time regardless of their movements. Typically, these drivers act as individual agents who have own unique working schedules and vehicles, and mostly aim to maximize their renumeration within their schedules. Therefore, distinguishing the true motives of the crowd who participates in these platforms seems important for an attractive system design. Eventually, crowd platforms heavily depend on crowd driver participation, and hence these platforms should keep the crowd drivers satisfied with their compensation schemes and other benefits.

From the operational perspective of crowdsourced delivery, further in-depth research is needed to investigate the interaction of the crowd workforce with a dedicated delivery workforce. These interactions are crucial since the crowd workforce is not homogenous in terms of working hours or the mode of transportation. As a result, a more stable delivery resource (*e.g.,* company employees or third-party couriers) is required to keep a certain level of service. We mainly focused on these interactions for the crowd drivers with limited time and detour willingness. However, in practice, we see that some of the crowd workforce stays in the system longer times, and they are flexible in spatial dimensions. Nevertheless, their appearance moments and exact duration times are not certain. Therefore, mathematical models that incorporates this type of workforce needs to be developed.

To use existing retailer infrastructures as a beneficial resource, we first explored the potential of using retailer stores as transshipment points in Chapter 3. Several directions can be pursued to extend this idea. One of the first of them might explore the question of how to distribute the created benefits among the stores. This question is critical in case these stores are not branches of the same organization. The second avenue for exploration is more from the operational perspective: In contrast to transfers in ride-sharing, a parcel in crowdshipping can, in theory, be

transferred numerous times as long as it is delivered on time. Therefore, devising efficient algorithms that scale with the number of stores, delivery requests, and crowd drivers is the next step for this research stream.

As an alternative way to exploit existing retailer stores in on-demand delivery services, we investigated online shopping and delivery platforms. While these platforms are still relatively new in the last-mile logistics, many of them have already proven their success. Due to the low investment costs compared to warehouse based supply chains, they seem to maintain growth in the near future, particularly in urban areas where retailer stores' density is high.

In Chapter 4, we have mainly evaluated the benefits of request split operations for online shopping and delivery platforms. However, this emerging business idea with a disruptive supply chain model opens many more research questions. From a strategic point of view, it can be worthwhile to investigate whether or not this supply model would be a viable strategy that can sustain the on-demand deliveries, and/or whether it is a scalable approach for mega cities. Also, incorporating the crowd workforce as shoppers could be a viable strategy and an assessment of how a crowd of drivers and shoppers create benefits in these models is crucial for large-scale implementation. From the operational perspective, these platforms and the corresponding supply models generate various research questions. One of them is to evaluate the potential of task transfers between shoppers. In this way, a shopper can deliver a customer request fully, and platforms can still get benefit from request splits. Another idea is to separate shopping and delivery tasks into only shopping and only delivery tasks for popular stores. As in the traditional supply chain setting, in which pickers in warehouses to prepares orders to pass delivery employees for shipments, in some stores dedicated in-store pickers perform only shopping tasks and make these tasks ready for the dedicated delivery employees.

Bibliography

Acimovic, J. & Graves, S. C. (2014). Making better fulfillment decisions on the fly in an online retail environment. *Manufacturing & Service Operations Management*, 17(1), 34–51.

Agatz, N., Erera, A., Savelsbergh, M., & Wang, X. (2012a). Optimization for dynamic ride-sharing: A review. *European Journal of Operational Research*, 223(2), 295–303.

Agatz, N., Erera, A., Savelsbergh, M., & Wang, X. (2012b). Optimization for dynamic ride-sharing: A review. *European Journal of Operational Research*, 223(2), 295–303.

Agatz, N. A., Erera, A. L., Savelsbergh, M. W., & Wang, X. (2011). Dynamic ride-sharing: A simulation study in metro Atlanta. *Transportation Research Part B: Methodological*, 45(9), 1450–1464.

Ahuja, R. K., Magnanti, T. L., & Orlin, J. B. (1993). *Network flows: theory, algorithms, and applications*. New Jersey: Prentice hall.

Amazon (2018). Primenow. `https://www.amazon.com/primeinsider/tips/prime-now-qa.html`. Accessed: 2019-06-17.

Archetti, C., Feillet, D., & Speranza, M. G. (2015). Complexity of routing problems with release dates. *European Journal of Operational Research*, 247(3), 797–803.

Archetti, C., Savelsbergh, M., & Speranza, M. G. (2016). The vehicle routing problem with occasional drivers. *European Journal of Operational Research*, 254(2), 472–480.

Archetti, C., Savelsbergh, M. W., & Speranza, M. G. (2008). To split or not to split: That is the question. *Transportation Research Part E: Logistics and Transportation Review*, 44(1), 114–123.

Archetti, C. & Speranza, M. G. (2008). The split delivery vehicle routing problem: a survey. In *The vehicle routing problem: Latest advances and new challenges* (pp. 103–122). Springer.

Arslan, A. M., Agatz, N., Kroon, L., & Zuidwijk, R. (2019a). Crowdsourced delivery: A dynamic pickup and delivery problem with ad hoc drivers. *Transportation Science*, 53(1), 222–235.

Arslan, A. M., Niels, A., & Klapp, M. (2019b). Splitting shopping and delivery tasks in an on-demand personal shopper service. *ERIM Report Series*.

Balasubramanian, S. (1998). Mail versus mall: A strategic analysis of competition between direct marketers and conventional retailers. *Marketing Science*, 17(3), 181–195.

Baldacci, R., Maniezzo, V., & Mingozzi, A. (2004). An exact method for the car pooling problem based on lagrangean column generation. *Operations Research*, 52(3), 422–439.

Berbeglia, G., Cordeau, J.-F., Gribkovskaia, I., & Laporte, G. (2007). Static pickup and delivery problems: a classification scheme and survey. *Top*, 15(1), 1–31.

Berbeglia, G., Cordeau, J.-F., & Laporte, G. (2010). Dynamic pickup and delivery problems. *European journal of operational research*, 202(1), 8–15.

Bhattarai, A. (2017). Walmart is asking employees to deliver packages on their way home from work.

Bol.com (2019). Dutch marketplace bol.com starts 2 hour delivery in the netherlands. `https://www.channelengine.com/company/marketplace-blog/dutch-marketplace-bolcom-start` Accessed: 2019-06-17.

Buldeo Rai, H., Verlinde, S., & Macharis, C. (2019). The ?next day, free delivery? myth unravelled: Possibilities for sustainable last mile transport in an omnichannel environment. *International Journal of Retail & Distribution Management*, 47(1), 39–54.

Cachon, G. P., Daniels, K. M., & Lobel, R. (2017). The role of surge pricing on a service platform with self-scheduling capacity. *Manufacturing & Service Operations Management*, 19(3), 368–384.

Carbone, V., Rouquet, A., & Roussat, C. (2015). Carried away by the crowd?: what types of logistics characterise collaborative consumption. In *1st International Workshop on Sharing Economy* (pp. 1–21).

Cattaruzza, D., Absi, N., & Feillet, D. (2016). Vehicle routing problems with multiple trips. *4OR*, 14(3), 223–259.

Chen, M. K. (2016). Dynamic pricing in a labor market: Surge pricing and flexible work on the uber platform. In *Proceedings of the 2016 ACM Conference on Economics and Computation* (pp. 455–455).: ACM.

Chen, W., Mes, M., & Schutten, M. (2017). Multi-hop driver-parcel matching problem with time windows. *Flexible services and manufacturing journal*, (pp. 1–37).

Chen, Z.-L. & Xu, H. (2006). Dynamic column generation for dynamic vehicle routing with time windows. *Transportation Science*, 40(1), 74–88.

Colby, C. & Bell, K. (2016). The on-demand economy is growing, and not just for the young and wealthy. *Harvard Business Review*.

Coltin, B. & Veloso, M. (2014). Ridesharing with passenger transfers. In *Intelligent Robots and Systems (IROS 2014), 2014 IEEE/RSJ International Conference on* (pp. 3278–3283).: IEEE.

Dahle, L., Andersson, H., & Christiansen, M. (2017). The vehicle routing problem with dynamic occasional drivers. In *International Conference on Computational Logistics* (pp. 49–63).: Springer.

Dayarian, I. & Savelsbergh, M. (2017). Crowdshipping and same-day delivery: Employing in-store customers to deliver online orders.

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

Doan, A., Ramakrishnan, R., & Halevy, A. Y. (2011). Crowdsourcing systems on the world-wide web. *Communications of the ACM*, 54(4), 86–96.

Einav, L., Farronato, C., & Levin, J. (2016). Peer-to-peer markets. *Annual Review of Economics*, 8, 615–635.

Fatnassi, E., Chaouachi, J., & Klibi, W. (2015). Planning and operating a shared goods and passengers on-demand rapid transit system for sustainable city-logistics. *Transportation Research Part B: Methodological*, 81, 440–460.

Furuhata, M., Dessouky, M., Ordóñez, F., Brunet, M.-E., Wang, X., & Koenig, S. (2013a). Ridesharing: The state-of-the-art and future directions. *Transportation Research Part B: Methodological*, 57, 28–46.

Furuhata, M., Dessouky, M., Ordóñez, F., Brunet, M.-E., Wang, X., & Koenig, S. (2013b). Ridesharing: The state-of-the-art and future directions. *Transportation Research Part B: Methodological*, 57, 28–46.

Gevaers, R., Van de Voorde, E., & Vanelslander, T. (2011). Characteristics and typology of last-mile logistics from an innovation perspective in an urban context. *City Distribution and Urban Freight Transport: Multiple Perspectives, Edward Elgar Publishing*, (pp. 56–71).

Ghilas, V., Demir, E., & Van Woensel, T. (2013). Integrating passenger and freight transportation: model formulation and insights. In *Beta Working Papers WP 441*. Technische Universiteit Eindhoven.

Gurobi (2016). Inc.: Gurobi optimization (version 7.5.0)[software].

Gurobi Optimiziation, I. (2016). Gurobi optimizer version 6.5.1 (software program).

Holsenbeck, K. (2018). Everything you need to know about prime now. Accessed on 12/12/2018.

Howe, J. (2006). The rise of crowdsourcing. *Wired magazine*, 14(6), 1–4.

Joerss, M., Schröder, J., Neuhaus Florian, Klink, C., & Mann, F. (2016). Parcel delivery. the future of last mile.

Kafle, N., Zou, B., & Lin, J. (2017). Design and modeling of a crowdsource-enabled system for urban parcel relay and delivery. *Transportation research part B: methodological*, 99, 62–82.

Kerrigan, H. (2016). The on-demand economy. http://businessresearcher.sagepub.com/.

Klapp, M. A. (2016). *Dynamic optimization for same-day delivery operations*. PhD thesis, Georgia Institute of Technology.

Klapp, M. A., Erera, A. L., & Toriello, A. (2018a). The dynamic dispatch waves problem for same-day delivery. *European Journal of Operational Research*, 271(2), 519–534.

Klapp, M. A., Erera, A. L., & Toriello, A. (2018b). The one-dimensional dynamic dispatch waves problem. *Transportation Science*, 52(2), 402–415.

Lee, A. & Savelsbergh, M. (2015). Dynamic ridesharing: Is there a role for dedicated drivers? *Transportation Research Part B: Methodological*, 81, Part 2, 483 – 497.

Li, B., Krushinsky, D., Reijers, H. A., & Van Woensel, T. (2014). The share-a-ride problem: People and parcels sharing taxis. *European Journal of Operational Research*, 238(1), 31–40.

Li, B., Krushinsky, D., Van Woensel, T., & Reijers, H. A. (2016). An adaptive large neighborhood search heuristic for the share-a-ride problem. *Computers & Operations Research*, 66, 170–180.

Malandraki, C. & Dial, R. B. (1996). A restricted dynamic programming heuristic algorithm for the time dependent traveling salesman problem. *European Journal of Operational Research*, 90(1), 45–55.

Martin, J., Florian, N., & Jürgen, S. (2016). How customer demands are reshaping last-mile delivery.

Masoud, N. & Jayakrishnan, R. (2017a). A decomposition algorithm to solve the multi-hop peer-to-peer ride-matching problem. *Transportation Research Part B: Methodological*, 99, 1–29.

Masoud, N. & Jayakrishnan, R. (2017b). A real-time algorithm to solve the peer-to-peer ride-matching problem in a flexible ridesharing system. *Transportation Research Part B: Methodological*, 106, 218–236.

Masson, R., Trentini, A., Lehuédé, F., Malhéné, N., Péton, O., & Tlahig, H. (2014). Optimization of a city logistics transportation system with mixed passengers and goods. *EURO Journal on Transportation and Logistics*, (pp. 1–29).

Miller, J., Nie, Y., & Stathopoulos, A. (2017). Crowdsourced urban package delivery: Modeling traveler willingness to work as crowdshippers. *Transportation Research Record: Journal of the Transportation Research Board*, (2610), 67–75.

Mingozzi, A., Bianco, L., & Ricciardelli, S. (1997). Dynamic programming strategies for the traveling salesman problem with time window and precedence constraints. *Operations Research*, 45(3), 365–377.

Morphy, E. (2014). About Walmart's idea to crowdsource its same-day delivery service.

Nowak, M., Ergun, Ö., & White III, C. C. (2008). Pickup and delivery with split loads. *Transportation Science*, 42(1), 32–43.

Nowak, M., Ergun, O., & White III, C. C. (2009). An empirical study on the benefit of split loads with the pickup and delivery problem. *European Journal of Operational Research*, 198(3), 734–740.

Paloheimo, H., Lettenmeier, M., & Waris, H. (2016). Transport reduction by crowdsourced deliveries–a library case in finland. *Journal of Cleaner Production*, 132, 240–251.

Pillac, V., Gendreau, M., Guéret, C., & Medaglia, A. L. (2013). A review of dynamic vehicle routing problems. *European Journal of Operational Research*, 225(1), 1–11.

Pisinger, D. & Ropke, S. (2010). Large neighborhood search. In *Handbook of metaheuristics* (pp. 399–419). Springer.

Popper, B. (2015). Drones could make Amazon's dream of free delivery profitable.

Punel, A., Ermagun, A., & Stathopoulos, A. (2018). Studying determinants of crowd-shipping use. *Travel Behaviour and Society*, 12, 30–40.

Punel, A. & Stathopoulos, A. (2017). Modeling the acceptability of crowdsourced goods deliveries: Role of context and experience effects. *Transportation Research Part E: Logistics and Transportation Review*, 105, 18–38.

Puterman, M. L. (2014). *Markov decision processes: discrete stochastic dynamic programming*. John Wiley & Sons.

Renaud, J., Laporte, G., & Boctor, F. F. (1996). A tabu search heuristic for the multi-depot vehicle routing problem. *Computers & Operations Research*, 23(3), 229–235.

Reyes, D., Erera, A., Savelsbergh, M., Sahasrabudhe, S., & O'Neil, R. (2018). The meal delivery routing problem. *Optimization Online*.

Rougès, J.-F. & Montreuil, B. (2014). Crowdsourcing delivery: New interconnected business models to reinvent delivery. In *1 st International Physical Internet Conference* (pp. 28–30).

Sadilek, A., Krumm, J., & Horvitz, E. (2013). Crowdphysics: Planned and opportunistic crowdsourcing for physical tasks. *SEA*, 21(10,424), 125–620.

Sampaio, A., Savelsbergh, M., Veelenturf, L., & Van Woensel, T. (2019). Crowd-based city logistics. In *Sustainable Transportation and Smart Logistics* (pp. 381–400). Elsevier.

Savelsbergh, M. & Van Woensel, T. (2016). 50th anniversary invited article city logistics: Challenges and opportunities. *Transportation Science*, 50(2), 579–590.

Savelsbergh, M. W. (1985). Local search in routing problems with time windows. *Annals of Operations research*, 4(1), 285–305.

Savelsbergh, M. W. & Sol, M. (1995). The general pickup and delivery problem. *Transportation science*, 29(1), 17–29.

Skrovan, S. (2017). Why most shoppers still choose brick-and-mortar stores over e-commerce. `https://www.retaildive.com/news/why-most-shoppers-still-choose-brick-and-mortar-stores-over-e` Accessed: 2019-06-18.

Solomon, B. (2016). America's most promising company: Instacart, the $2 billion grocery delivery app. url=https://www.forbes.com/sites/briansolomon/2015/01/21/americas-most-promising-company-instacart-the-2-billion-grocery-delivery-app/73f9385742dc. Accessed on 7/02/2019.

Srour, F. J., Agatz, N., & Oppen, J. (2016). Strategies for handling temporal uncertainty in pickup and delivery problems with time windows. *Transportation Science*, 52(1), 3–19.

Steever, Z., Karwan, M., & Murray, C. (2019). Dynamic courier routing for a food delivery service. *Computers & Operations Research*, 107, 173–188.

Stiglic, M., Agatz, N., Savelsbergh, M., & Gradisar, M. (2015). The benefits of meeting points in ride-sharing systems. *Transportation Research Part B: Methodological*, 82, 36 – 53.

Suh, K., Smith, T., & Linhoff, M. (2012). Leveraging socially networked mobile ict platforms for the last-mile delivery problem. *Environmental science & technology*, 46(17), 9481–9490.

Tang, C. S., Bai, J., So, K. C., Chen, X. M., & Wang, H. (2016). *Coordinating supply and demand on an on-demand platform: Price, wage, and payout ratio*. Technical report.

Tilk, C. & Irnich, S. (2016). Dynamic programming for the minimum tour duration problem. *Transportation Science*, 51(2), 549–565.

Ulmer, M. W., Thomas, B. W., Campbell, A. M., & Woyak, N. (2017). The restaurant meal delivery problem: Dynamic pick-up and delivery with deadlines and random ready times.

Voccia, S. A., Campbell, A. M., & Thomas, B. W. (2017). The same-day delivery problem for online purchases. *Transportation Science*, 53(1), 167–184.

Xu, P. J., Allgor, R., & Graves, S. C. (2009). Benefits of reevaluating real-time order fulfillment decisions. *Manufacturing & Service Operations Management*, 11(2), 340–355.

Yildiz, B. & Savelsbergh, M. (2017). Provably high-quality solutions for the meal delivery routing problem. *Georgia Institute of Technology*.

About the author

Alp Arslan was born in Istanbul, Turkey, on March 20, 1987. He studied Industrial Engineering at Istanbul Technical University and Sabanci University, in Turkey. In 2014, he obtained his M.Sc. degree in Operations Research and Statistics at University North Carolina Chapel Hill in the USA with a thesis on Flexible Products in Revenue Management. In 2014, Alp became a Ph.D. candidate at the Technology and Operations Management Department of Rotterdam School of Management at the Erasmus University Rotterdam under the supervision of Professor Rob Zuidwijk and Dr. Niels Agatz.

Alp's research focusses on the operational problems arises in the on-demand delivery services and how to improve these services with help of the sharing economy and online platforms. Alp has published in the Transportation Science and presented his work at several international conferences. He has also served as an ad-hoc reviewer of various journals including Transportation Science, Transportation Research Part C. Alp currently works as a Research Scientist of School of Information Systems at the Singapore Management University.

Summary

Arguably, on-demand delivery is the ultimate challenge in the whole supply chain; particularly in the last-mile logistics. Typically, small-sized parcels have to be shipped to geographically dispersed locations as little as in an hour. Also, delivery vehicles, in this service, are required to visit the stocking locations or depots multiple times to collect arriving customer orders while they are on delivery. Returning to depots often within the service period, however, increases the marginal mileage per order, and, consequently more resources are required to serve the same amount of customers as compared to *e.g,* a next-day delivery service. In short, offering expedited delivery is costly for providers.

Service providers, however, are unable to fully transfer the cost of the on-demand delivery to online shoppers due to the consumers' resistance to delivery fees. Empirical studies show that consumers are highly price-sensitive towards delivery fees, and they are often unwilling to pay for faster deliveries. Therefore, service providers must absorb most of the costs to stay competitive in the market. At a consequence, they require to rethink two fundamental questions in order to create cost-effective on-demand delivery services:

1. From what locations to serve customer demand?
2. How to efficiently organize the last-mile to customers' addresses?

This dissertation focused on examining innovative concepts to answer these questions above. In particular, we contemplated to utilize existing traffic flows and retail infrastructures by studying *crowdsourced delivery* and *online shopping and delivery platforms* in on-demand delivery. Specifically, we contribute decision support tools of such on-demand delivery providers which contemplate using or currently use one of these two ideas.

In Chapter 2, we introduced crowdsourced delivery as a dynamic pickup delivery problem. We considered an online peer to peer platform organizes willing ad-hoc drivers to make the on-demand parcel deliveries on their way to their destinations. In this model, dedicated employees also guarantee the on-time deliveries in case no crowd drivers are available momentarily. We devised an event based rolling horizon framework that solves a matching problem to disclose the most updated assignment solution at an arrival of online request or ad-hoc driver. The computational experiments present the benefits of crowd drivers in an on-demand delivery service. The results show that not only the total mileage reduces with the presence of ad-hoc crowd drivers but also the required number of dedicated employees. Furthermore, the sensitivity analysis unveils that the overall benefits of crowd drivers depends on their willingness to stop more and/or incur longer detours. Not but not the least, we see that crowd driver integration into delivery systems is most beneficial when drivers and requests origins coincides.

We elaborated the dynamic crowd shipping implementation with store transfers in Chapter 3. In this study we solely consider in-store customers as crowd drivers and introduced store transfers, in which two crowd drivers can transfer packages to another to finalize a single parcel delivery. The advantageous feature of the store transfers is removing the necessity of restricted time syncronizaty of drivers. Similar to the previous study, we propose a matching framework that is integrated into the rolling horizon mechanism. The difference comparing to the matching problem in Chapter 2 is that constructing feasible driver(s), task(s) pairs must follow the time and space compatibility. We test the value of store transfers with a synthetic dataset. The numerical results reveal that store transfers not only decrease the total cost of the system, but also reduces the detour amount of individual drivers on average.

In Chapter 4, we particularly looked at the opportunities to utilize existing retail stores. To do so, we envisioned an online marketplace that provides a personal shopping and on-demand delivery service. In this model, a customer places an order potentially requiring shopping from multiple stores and shoppers of the platform visit these stores, purchase products, and deliver to customers? desired addresses. We test several shopping and delivery strategies including request consolidating, which a shopper serves multiple customers at a time; request splitting, which

a customer is allowed to be served by multiple shoppers. We formulate this problem under the dynamic sequential decision making and solve the pickup and split delivery problem as a subproblem. The numerical experiments present that allowing requests consolidation and splitting increases the capacity utilization of shoppers significantly. Furthermore, we observe that request splitting does not increase the click to delivery time.

In conclusion, we explored two asset-light opportunities in on-demand delivery services. In both ideas, we considered the delivery locations in the close vicinity of customers? addresses, and therefore willing crowd drivers can easily and effectively participate crowd shipping platforms. Also, using existing stores as fulfillment centers eliminates the necessity of large-scale warehouse investments near urban areas. Subsequently, we investigated the benefits of two aforementioned ideas in on-demand delivery operations. First, we observed that organizing crowd drivers effectively reduces the last-mile delivery costs of on-demand services substantially. Second, we present the effective shopping and delivery operation strategies when customer request containing shopping from multiple local stores.

Samenvatting (Summary in Dutch)

Onbetwistbaar leveren is ongetwijfeld de ultieme uitdaging in de hele supply chain; vooral in de laatste mijl logistiek. Kleine pakketten moeten doorgaans binnen een uur naar geografisch verspreide locaties worden verzonden. Ook moeten bezorgvoertuigen in deze service meerdere keren de stocklocaties of depots bezoeken om aankomende klantbestellingen te verzamelen terwijl ze worden afgeleverd. Terugkeren naar depots vaak binnen de serviceperiode verhoogt echter de marginale kilometerstand per bestelling, en bijgevolg zijn meer middelen nodig om hetzelfde aantal klanten te bedienen in vergelijking met textit bijv. Een bezorgservice op de volgende dag. Kortom, het aanbieden van versnelde levering is kostbaar voor providers.

Serviceproviders zijn echter niet in staat om de kosten van de on-demand levering volledig over te dragen aan online shoppers vanwege de weerstand van de consument tegen bezorgkosten. Empirische studies tonen aan dat consumenten zeer prijsgevoelig zijn voor bezorgkosten en dat ze vaak niet willen betalen voor snellere leveringen. Daarom moeten dienstverleners de meeste kosten op zich nemen om concurrerend te blijven in de markt. Bijgevolg moeten ze twee fundamentele vragen heroverwegen om kosteneffectieve bezorgdiensten op afroep te creëren:

1. Van welke locaties om de klantvraag te bedienen?
2. Hoe de last-mile naar de adressen van klanten efficiënt organiseren?

Dit proefschrift was gericht op het onderzoeken van innovatieve concepten om deze vragen hierboven te beantwoorden. We hebben met name overwogen om bestaande verkeersstromen en retailinfrastructuren te gebruiken door textit crowdsourced delivery en textit online shopping and delivery platforms te bestuderen in on-demand levering. Specifiek dragen we bij aan beslissingsondersteunende tools van

dergelijke on-demand bezorgproviders die overwegen een van deze twee ideeën te gebruiken of die momenteel gebruiken.

In hoofdstuk 2 hebben we crowdsourced-bezorging geïntroduceerd als een probleem met dynamische bezorging bij afhalen. We hebben overwogen dat een online peer-to-peer-platform bereidwillige ad-hocchauffeurs organiseert om de pakketbezorging op aanvraag op weg naar hun bestemming te maken. In dit model garanderen toegewijde medewerkers ook de tijdige leveringen voor het geval er tijdelijk geen crowd-drivers beschikbaar zijn. We hebben een event-gebaseerd rolling horizon-raamwerk ontwikkeld dat een matchingprobleem oplost om de meest bijgewerkte toewijzingsoplossing te onthullen bij een online aanvraag of ad-hocstuurprogramma. De computationele experimenten presenteren de voordelen van crowd drivers in een on-demand bezorgservice. De resultaten laten zien dat niet alleen het totale aantal kilometers afneemt met de aanwezigheid van ad-hoc crowd drivers, maar ook het vereiste aantal toegewijde medewerkers. Bovendien onthult de gevoeligheidsanalyse dat de algemene voordelen van crowd drivers afhangen van hun bereidheid om meer te stoppen en / of langere omwegen te maken. Niet alleen, maar niet minder belangrijk, zien we dat de integratie van crowd-drivers in afleversystemen het meest voordelig is wanneer chauffeurs en verzoeken van oorsprong samenvallen.

In hoofdstuk 3 hebben we de implementatie van dynamische crowd-shipping met winkeltransfers uitgewerkt. In dit onderzoek beschouwen we klanten in de winkel alleen als crowd-drivers en hebben we winkeltransfers geïntroduceerd, waarbij twee crowd-chauffeurs pakketten naar een andere kunnen overbrengen om een ??enkele pakketbezorging te voltooien. Het voordelige kenmerk van de winkeltransfers is het verwijderen van de noodzaak van beperkte tijdsynchronisatie van stuurprogramma's. Net als in de vorige studie stellen we een passend kader voor dat is geïntegreerd in het rolling horizon-mechanisme. Het verschil met het matching probleem in hoofdstuk 2 is dat het construeren van haalbare driver (s), taak (en) paren de tijd- en ruimtecompatibiliteit moet volgen. We testen de waarde van winkeltransfers met een synthetische dataset. De numerieke resultaten onthullen dat winkeloverdrachten niet alleen de totale kosten van het systeem verlagen, maar ook gemiddeld het aantal omleidingen van individuele chauffeurs verminderen.

In hoofdstuk 4 hebben we met name gekeken naar de mogelijkheden om bestaande winkels te gebruiken. Om dit te doen, hadden we een online marktplaats voor

ogen met een persoonlijke winkel en on-demand bezorgservice. In dit model plaatst een klant een bestelling waarvoor mogelijk winkelen bij meerdere winkels nodig is en shoppers van het platform bezoeken deze winkels, kopen producten en leveren aan de gewenste adressen van klanten. We testen verschillende winkel- en bezorgstrategieën, waaronder het consolideren van aanvragen, waarbij een klant meerdere klanten tegelijk bedient; verzoeksplitsing, waarbij een klant door meerdere klanten mag worden bediend. We formuleren dit probleem onder de dynamische sequentiële besluitvorming en lossen het ophaal- en gesplitste bezorgprobleem op als een subprobleem. De numerieke experimenten presenteren dat het toestaan ??van consolidatie en splitsen van aanvragen de bezettingsgraad van shoppers aanzienlijk verhoogt. Verder zien we dat het splitsen van aanvragen de klik naar levertijd niet verhoogt.

Concluderend hebben we twee mogelijkheden voor activabewustzijn in bezorgdiensten op aanvraag onderzocht. In beide ideeën hebben we rekening gehouden met de afleverlocaties in de buurt van de adressen van klanten, en daarom kunnen bereidwillige crowd drivers eenvoudig en effectief deelnemen aan crowd shipping platforms. Het gebruik van bestaande winkels als fulfilment centra elimineert ook de noodzaak van grootschalige magazijninvesteringen in de buurt van stedelijke gebieden. Vervolgens hebben we de voordelen van de twee bovengenoemde ideeën in on-demand leveringsactiviteiten onderzocht. Ten eerste hebben we geconstateerd dat het organiseren van crowd drivers effectief de last-mile bezorgkosten van on-demand services aanzienlijk vermindert. Ten tweede presenteren we de effectieve winkel- en bezorgingsstrategieën bij verzoeken van klanten met winkelen bij meerdere lokale winkels.

ERIM Ph.D. Series Research in Management

The ERIM PhD Series contains PhD dissertations in the field of Research in Management defended at Erasmus University Rotterdam and supervised by senior researchers affiliated to the Erasmus Research Institute of Management (ERIM). All dissertations in the ERIM PhD Series are available in full text through the ERIM Electronic Series Portal: `http://repub.eur.nl/pub`. ERIM is the joint research institute of the Rotterdam School of Management (RSM) and the Erasmus School of Economics at the Erasmus University Rotterdam (EUR).

Dissertations last four years

Ahmadi, S., A motivational perspective to decision-making and behavior in organizations, Promotors: Prof. J.J.P. Jansen & Prof. T.J.M. Mom, EPS-2019-477-S&E, https://repub.eur.nl/pub/116727

Akemu, O., Corporate Responses to Social Issues: Essays in Social Entrepreneurship and Corporate Social Responsibility, Promotors: Prof. G.M. Whiteman & Dr. S.P. Kennedy, EPS-2017-392-ORG, https://repub.eur.nl/pub/95768

Albuquerque de Sousa, J.A., International stock markets: Essays on the determinants and consequences of financial market development, Promotors: Prof. M.A. van Dijk & Prof. P.A.G. van Bergeijk, EPS-2019-465-F&A, https://repub.eur.nl/pub/115988

Alexiou, A., Management of Emerging Technologies and the Learning Organization: Lessons from the Cloud and Serious Games Technology, Promotors: Prof.

S.J. Magala, Prof. M.C. Schippers & Dr. I. Oshri, EPS-2016-404-ORG, https://repub.eur.nl/pub/93818

Alserda, G.A.G., Choices in Pension Management, Promotors: Prof. S.G. van der Lecq & Dr. O.W. Steenbeek, EPS-2017-432-F&A, https://repub.eur.nl/pub/103496

Arampatzi, E., Subjective Well-Being in Times of Crises: Evidence on the Wider Impact of Economic Crises and Turmoil on Subjective Well-Being, Promotors: Prof. H.R. Commandeur, Prof. F. van Oort & Dr. M.J. Burger, EPS-2018-459-S&E, https://repub.eur.nl/pub/111830

Avci, E., Surveillance of Complex Auction Markets: a Market Policy Analytics Approach, Promotors: Prof. W. Ketter, Prof. H.W.G.M. van Heck & Prof. D.W. Bunn, EPS-2018-426-LIS, https://repub.eur.nl/pub/106286

Balen, T.H. van, Challenges of Early Stage Entrepreneurs : the Roles of Vision Communication and Team Membership Change, Promotors: Prof. J.C.M van den Ende & Dr. M. Tarakci, EPS-2018-468-LIS, https://repub.eur.nl/pub/115654

Benschop, N, Biases in Project Escalation: Names, frames & construal levels, Promotors: Prof. K.I.M. Rhode, Prof. H.R. Commandeur, Prof. M. Keil & Dr. A.L.P. Nuijten, EPS-2015-375-S&E, https://repub.eur.nl/pub/79408

Bernoster, I., Essays at the Intersection of Psychology, Biology, and Entrepreneurship, Promotors: Prof. A.R. Thurik, Prof. I.H.A. Franken & Prof. P.J.F Groenen, EPS-2018-463-S&E, https://repub.eur.nl/pub/113907

Beusichem, H.C. van, Firms and Financial Markets: Empirical Studies on the Informational Value of Dividends, Governance and Financial Reporting, Promotors: Prof. A. de Jong & Dr. G. Westerhuis, EPS-2016-378-F&A, https://repub.eur.nl/pub/93079

Bouman, P., Passengers, Crowding and Complexity: Models for Passenger Oriented Public Transport, Promotors: Prof. L.G. Kroon, Prof. A. SchÃ¶bel & Prof. P.H.M. Vervest, EPS-2017-420-LIS, https://repub.eur.nl/pub/100767

Bunderen, L. van, Tug-of-War: Why and when teams get embroiled in power struggles, Promotors: Prof. D.L. van Knippenberg & Dr. L. Greer, EPS-2018-446-ORG, https://repub.eur.nl/pub/105346

Burg, G.J.J. van den, Algorithms for Multiclass Classification and Regularized Regression, Promotors: Prof. P.J.F. Groenen & Dr. A. Alfons, EPS-2018-442-MKT, https://repub.eur.nl/pub/103929

Chammas, G., Portfolio concentration, Promotor: Prof. J. Spronk, EPS-2017-410-F&E, https://repub.eur.nl/pub/94975

Consiglio, I., Others: Essays on Interpersonal and Consumer Behavior, Promotor: Prof. S.M.J. van Osselaer, EPS-2016-366-MKT, https://repub.eur.nl/pub/79820

Cranenburgh, K.C. van, Money or Ethics: Multinational corporations and religious organisations operating in an era of corporate responsibility, Promotors: Prof. L.C.P.M. Meijs, Prof. R.J.M. van Tulder & Dr. D. Arenas, EPS-2016-385-ORG, https://repub.eur.nl/pub/93104

Darnihamedani, P., Individual Characteristics, Contextual Factors and Entrepreneurial Behavior, Promotors: Prof. A.R. Thurik & S.J.A. Hessels, EPS-2016-360-S&E, https://repub.eur.nl/pub/93280

Dennerlein, T., Empowering Leadership and Employees Achievement Motivations: the Role of Self-Efficacy and Goal Orientations in the Empowering Leadership Process, Promotors: Prof. D.L. van Knippenberg & Dr. J. Dietz, EPS-2017-414-ORG, https://repub.eur.nl/pub/98438

Depecik, B.E., Revitalizing brands and brand: Essays on Brand and Brand Portfolio Management Strategies, Promotors: Prof. G.H. van Bruggen, Dr. Y.M. van Everdingen and Dr. M.B. Ataman, EPS-2016-406-MKT, https://repub.eur.nl/pub/93507

Duijzer, L.E., Mathematical Optimization in Vaccine Allocation, Promotors: Prof. R. Dekker & Dr. W.L. van Jaarsveld, EPS-2017-430-LIS, https://repub.eur.nl/pub/101487

Duyvesteyn, J.G., Empirical Studies on Sovereign Fixed Income Markets, Promotors: Prof. P. Verwijmeren & Prof. M.P.E. Martens, EPS-2015-361-F&A, https://repub.eur.nl/pub/79033

El Nayal, O.S.A.N., Firms and the State: An Examination of Corporate Political Activity and the Business-Government Interface, Promotor: Prof. J. van Oosterhout & Dr. M. van Essen, EPS-2018-469-S&E, https://repub.eur.nl/pub/114683

Elemes, A., Studies on Determinants and Consequences of Financial Reporting Quality, Promotor: Prof. E. Peek, EPS-2015-354-F&A, https://repub.eur.nl/pub/79037

Erlemann, C., Gender and Leadership Aspiration: The Impact of the Organizational Environment, Promotor: Prof. D.L. van Knippenberg, EPS-2016-376-ORG, https://repub.eur.nl/pub/79409

Faber, N., Structuring Warehouse Management, Promotors: Prof. M.B.M. de Koster & Prof. A. Smidts, EPS-2015-336-LIS, https://repub.eur.nl/pub/78603

Feng, Y., The Effectiveness of Corporate Governance Mechanisms and Leadership Structure: Impacts on strategic change and firm performance, Promotors: Prof. F.A.J. van den Bosch, Prof. H.W. Volberda & Dr. J.S. Sidhu, EPS-2017-389-S&E, https://repub.eur.nl/pub/98470

Fernald, K., The Waves of Biotechnological Innovation in Medicine: Interfirm Cooperation Effects and a Venture Capital Perspective, Promotors: Prof. E. Claassen, Prof. H.P.G. Pennings & Prof. H.R. Commandeur, EPS-2015-371-S&E, https://repub.eur.nl/pub/79120

Fisch, C.O., Patents and trademarks: Motivations, antecedents, and value in industrialized and emerging markets, Promotors: Prof. J.H. Block, Prof. H.P.G. Pennings & Prof. A.R. Thurik, EPS-2016-397-S&E, https://repub.eur.nl/pub/94036

Fliers, P.T., Essays on Financing and Performance: The role of firms, banks and board, Promotors: Prof. A. de Jong & Prof. P.G.J. Roosenboom, EPS-2016-388-F&A, https://repub.eur.nl/pub/93019

Frick, T.W., The Implications of Advertising Personalization for Firms, Consumer, and Ad Platfroms, Promotors: Prof. T. Li & Prof. H.W.G.M. van Heck, EPS-2018-452-LIS, https://repub.eur.nl/pub/110314

Fytraki, A.T., Behavioral Effects in Consumer Evaluations of Recommendation Systems, Promotors: Prof. B.G.C. Dellaert & Prof. T. Li, EPS-2018-427-MKT, https://repub.eur.nl/pub/110457

Gaast, J.P. van der, Stochastic Models for Order Picking Systems, Promotors: Prof. M.B.M de Koster & Prof. I.J.B.F. Adan, EPS-2016-398-LIS, https://repub.eur.nl/pub/93222

Ghazizadeh, P., Empirical Studies on the Role of Financial Information in Asset and Capital Markets, Promotors: Prof. A. de Jong & Prof. E. Peek. EPS-2019-470-F&A https://repub.eur.nl/pub/114023

Giurge, L., A Test of Time; A temporal and dynamic approach to power and ethics, Promotors: Prof. M.H. van Dijke & Prof. D. De Cremer, EPS-2017-412-ORG, https://repub.eur.nl/pub/98451

Gobena, L., Towards Integrating Antecedents of Voluntary Tax Compliance, Promotors: Prof. M.H. van Dijke & Dr. P. Verboon, EPS-2017-436-ORG, https://repub.eur.nl/pub/103276

Groot, W.A., Assessing Asset Pricing Anomalies, Promotors: Prof. M.J.C.M. Verbeek & Prof. J.H. van Binsbergen, EPS-2017-437-F&A, https://repub.eur.nl/pub/103490

Hanselaar, R.M., Raising Capital: On pricing, liquidity and incentives, Promotors: Prof. M.A. van Dijk & Prof. P.G.J. Roosenboom, EPS-2018-429-F&A-9789058925404, https://repub.eur.nl/pub/113274

Harms, J. A., Essays on the Behavioral Economics of Social Preferences and Bounded Rationality, Promotors: Prof. H.R. Commandeur & Dr. K.E.H. Maas, EPS-2018-457-S&E, https://repub.eur.nl/pub/108831

Hekimoglu, M., Spare Parts Management of Aging Capital Products, Promotor: Prof. R. Dekker, EPS-2015-368-LIS, https://repub.eur.nl/pub/79092

Hendriks, G., Multinational Enterprises and Limits to International Growth: Links between Domestic and Foreign Activities in a Firms Portfolio, Promotors: Prof. P.P.M.A.R. Heugens & Dr. A.H.L Slangen, EPS-2019-464-S&E, https://repub.eur.nl/pub/114981

Hengelaar, G.A., The Proactive Incumbent: Holy grail or hidden gem? Investigating whether the Dutch electricity sector can overcome the incumbents curse and lead the sustainability transition, Promotors: Prof. R.J. M. van Tulder & Dr. K. Dittrich, EPS-2018-438-ORG, https://repub.eur.nl/pub/102953

Hogenboom, A.C., Sentiment Analysis of Text Guided by Semantics and Structure, Promotors: Prof. U. Kaymak & Prof. F.M.G. de Jong, EPS-2015-369-LIS, https://repub.eur.nl/pub/79034

Hollen, R.M.A., Exploratory Studies into Strategies to Enhance Innovation-Driven International Competitiveness in a Port Context: Toward Ambidextrous Ports, Promotors: Prof. F.A.J. Van Den Bosch & Prof. H.W. Volberda, EPS-2015-372-S&E, https://repub.eur.nl/pub/78881

Jacobs, B.J.D., Marketing Analytics for High-Dimensional Assortments, Promotors: Prof. A.C.D. Donkers & Prof. D. Fok, EPS-2017-445-MKT, https://repub.eur.nl/pub/103497

Jia, F., The Value of Happiness in Entrepreneurship, Promotors: Prof. D.L. van Knippenberg & Dr. Y. Zhang, EPS-2019-479-ORG, https://repub.eur.nl/pub/115990

Kahlen, M. T., Virtual Power Plants of Electric Vehicles in Sustainable Smart Electricity Markets, Promotors: Prof. W. Ketter & Prof. A. Gupta, EPS-2017-431-LIS, https://repub.eur.nl/pub/100844

Kampen, S. van, The Cross-sectional and Time-series Dynamics of Corporate Finance: Empirical evidence from financially constrained firms, Promotors: Prof. L. Norden & Prof. P.G.J. Roosenboom, EPS-2018-440-F&A, https://repub.eur.nl/pub/105245

Karali, E., Investigating Routines and Dynamic Capabilities for Change and Innovation, Promotors: Prof. H.W. Volberda, Prof. H.R. Commandeur & Dr. J.S. Sidhu, EPS-2018-454-S&E, https://repub.eur.nl/pub/106274

Keko, E., Essays on Innovation Generation in Incumbent Firms, Promotors: Prof. S. Stremersch & Dr. N.M.A. Camacho, EPS-2017-419-MKT, https://repub.eur.nl/pub/100841

Kerkkamp, R.B.O., Optimisation Models for Supply Chain Coordination under Information Asymmetry, Promotors: Prof. A.P.M. Wagelmans & Dr. W. van den Heuvel, EPS-2018-462-LIS, https://repub.eur.nl/pub/109770

Khattab, J., Make Minorities Great Again: a contribution to workplace equity by identifying and addressing constraints and privileges, Promotors: Prof. D.L. van Knippenberg & Dr. A. Nederveen Pieterse, EPS-2017-421-ORG, https://repub.eur.nl/pub/99311

Kim, T. Y., Data-driven Warehouse Management in Global Supply Chains, Promotors: Prof. R. Dekker & Dr. C. Heij, EPS-2018-449-LIS, https://repub.eur.nl/pub/109103

Klitsie, E.J., Strategic Renewal in Institutional Contexts: The paradox of embedded agency, Promotors: Prof. H.W. Volberda & Dr. S. Ansari, EPS-2018-444-S&E, https://repub.eur.nl/pub/106275

Kong, L., Essays on Financial Coordination, Promotors: Prof. M.J.C.M. Verbeek, Dr. D.G.J. Bongaerts & Dr. M.A. van Achter. EPS-2019-433-F&A, https://repub.eur.nl/pub/114516

Koolen, D., Market Risks and Strategies in Power Systems Integrating Renewable Energy, Promotors: Prof. W. Ketter & Dr. R. Huisman, EPS-2019-467-LIS, https://repub.eur.nl/pub/115655

Krämer, R., A license to mine? Community organizing against multinational corporations, Promotors: Prof. R.J.M. van Tulder & Prof. G.M. Whiteman, EPS-2016-383-ORG, https://repub.eur.nl/pub/94072

Kyosev, G.S., Essays on Factor Investing, Promotors: Prof. M.J.C.M. Verbeek & Dr. J.J. Huij, EPS-2019-474-F&A, https://repub.eur.nl/pub/116463

Lamballais, T., Optimizing the Performance of Robotic Mobile Fulfillment Systems, Promotors: Prof. M.B.M de Koster & Prof. R. Dekker & Dr. D. Roy, EPS-2019-411-LIS, https://repub.eur.nl/pub/116477

Lee, C.I.S.G., Big Data in Management Research: Exploring New Avenues, Promotors: Prof. S.J. Magala & Dr. W.A. Felps, EPS-2016-365-ORG, https://repub.eur.nl/pub/79818

Legault-Tremblay, P.O., Corporate Governance During Market Transition: Heterogeneous responses to Institution Tensions in China, Promotor: Prof. B. Krug, EPS-2015-362-ORG, https://repub.eur.nl/pub/78649

Lenoir, A.S., Are You Talking to Me? Addressing Consumers in a Globalised World, Promotors: Prof. S. Puntoni & Prof. S.M.J. van Osselaer, EPS-2015-363-MKT, https://repub.eur.nl/pub/79036

Leung, W.L., How Technology Shapes Consumption: Implications for Identity and Judgement, Promotors: Prof. S. Puntoni & Dr. G. Paolacci, EPS-2019-485-MKT, https://repub.eur.nl/pub/117432

Li, D., Supply Chain Contracting for After-sales Service and Product Support, Promotor: Prof. M.B.M. de Koster, EPS-2015-347-LIS, https://repub.eur.nl/pub/78526

Li, X., Dynamic Decision Making under Supply Chain Competition, Promotors: Prof. M.B.M de Koster, Prof. R. Dekker & Prof. R. Zuidwijk. EPS-2018-466-LIS, https://repub.eur.nl/pub/114028

Liu, N., Behavioral Biases in Interpersonal Contexts, Promotors: Prof. A. Baillon & Prof. H. Bleichrodt, EPS-2017-408-MKT, https://repub.eur.nl/pub/95487

Ma, Y., The Use of Advanced Transportation Monitoring Data for Official Statistics, Promotors: Prof. L.G. Kroon & Dr. J. van Dalen, EPS-2016-391-LIS, https://repub.eur.nl/pub/80174

Maas, A.J.J., Organizations and their external context: Impressions across time and space, Promotors: Prof. P.P.M.A.R. Heugens & Prof. T.H. Reus, EPS-2019-478-S&E, https://repub.eur.nl/pub/116480

Maira, E., Consumers and Producers, Promotors: Prof. S. Puntoni & Prof. C. Fuchs, EPS-2018-439-MKT, https://repub.eur.nl/pub/104387

Mell, J.N., Connecting Minds: On The Role of Metaknowledge in Knowledge Coordination, Promotor: Prof. D.L. van Knippenberg, EPS-2015-359-ORG, https://repub.eur.nl/pub/78951

Meulen, D. van der, The Distance Dilemma: the effect of flexible working practices on performance in the digital workplace, Promotors: Prof. H.W.G.M. van Heck & Prof. P.J. van Baalen, EPS-2016-403-LIS, https://repub.eur.nl/pub/94033

Moniz, A, Textual Analysis of Intangible Information, Promotors: Prof. C.B.M. van Riel, Prof. F.M.G de Jong & Dr. G.A.J.M. Berens, EPS-2016-393-ORG, https://repub.eur.nl/pub/93001

Mulder, J., Network design and robust scheduling in liner shipping, Promotors: Prof. R. Dekker & Dr. W.L. van Jaarsveld, EPS-2016-384-LIS, https://repub.eur.nl/pub/80258

Neerijnen, P., The Adaptive Organization: the socio-cognitive antecedents of ambidexterity and individual exploration, Promotors: Prof. J.J.P. Jansen, P.P.M.A.R. Heugens & Dr. T.J.M. Mom, EPS-2016-358-S&E, https://repub.eur.nl/pub/93274

Okbay, A., Essays on Genetics and the Social Sciences, Promotors: Prof. A.R. Thurik, Prof. Ph.D. Koellinger & Prof. P.J.F. Groenen, EPS-2017-413-S&E, https://repub.eur.nl/pub/95489

Oord, J.A. van, Essays on Momentum Strategies in Finance, Promotor: Prof. H.K. van Dijk, EPS-2016-380-F&A, https://repub.eur.nl/pub/80036

Peng, X., Innovation, Member Sorting, and Evaluation of Agricultural Cooperatives, Promotor: Prof. G.W.J. Hendriks, EPS-2017-409-ORG, https://repub.eur.nl/pub/94976

Pennings, C.L.P., Advancements in Demand Forecasting: Methods and Behavior, Promotors: Prof. L.G. Kroon, Prof. H.W.G.M. van Heck & Dr. J. van Dalen, EPS-2016-400-LIS, https://repub.eur.nl/pub/94039

Petruchenya, A., Essays on Cooperatives: Emergence, Retained Earnings, and Market Shares, Promotors: Prof. G.W.J. Hendriks & Dr. Y. Zhang, EPS-2018-447-ORG, https://repub.eur.nl/pub/105243

Plessis, C. du, Influencers: The Role of Social Influence in Marketing, Promotors: Prof. S. Puntoni & Prof. S.T.L.R. Sweldens, EPS-2017-425-MKT, https://repub.eur.nl/pub/103265

Pocock, M., Status Inequalities in Business Exchange Relations in Luxury Markets, Promotors: Prof. C.B.M. van Riel & Dr. G.A.J.M. Berens, EPS-2017-346-ORG, https://repub.eur.nl/pub/98647

Pozharliev, R., Social Neuromarketing: The role of social context in measuring advertising effectiveness, Promotors: Prof. W.J.M.I. Verbeke & Prof. J.W. van Strien, EPS-2017-402-MKT, https://repub.eur.nl/pub/95528

Protzner, S., Mind the gap between demand and supply: A behavioral perspective on demand forecasting, Promotors: Prof. S.L. van de Velde & Dr. L. Rook, EPS-2015-364-LIS, https://repub.eur.nl/pub/79355

Reh, S.G., A Temporal Perspective on Social Comparisons in Organizations, Promotors: Prof. S.R. Giessner, Prof. N. van Quaquebeke & Dr. C. Troster, EPS-2018-471-ORG, https://repub.eur.nl/pub/114522

Riessen, B. van, Optimal Transportation Plans and Portfolios for Synchromodal Container Networks, Promotors: Prof. R. Dekker & Prof. R.R. Negenborn, EPS-2018-448-LIS, https://repub.eur.nl/pub/105248

Rietdijk, W.J.R., The Use of Cognitive Factors for Explaining Entrepreneurship, Promotors: Prof. A.R. Thurik & Prof. I.H.A. Franken, EPS-2015-356-S&E, https://repub.eur.nl/pub/79817

Roza, L., Employee Engagement in Corporate Social Responsibility: A collection of essays, Promotor: Prof. L.C.P.M. Meijs, EPS-2016-396-ORG, https://repub.eur.nl/pub/93254

RÃ¶sch, D., Market Efficiency and Liquidity, Promotor: Prof. M.A. van Dijk, EPS-2015-353-F&A, https://repub.eur.nl/pub/79121

Schie, R. J. G. van, Planning for Retirement: Save More or Retire Later?, Promotors: Prof. B. G. C. Dellaert & Prof. A.C.D. Donkers, EOS-2017-415-MKT, https://repub.eur.nl/pub/100846

Schoonees, P., Methods for Modelling Response Styles, Promotor: Prof. P.J.F. Groenen, EPS-2015-348-MKT, https://repub.eur.nl/pub/79327

Schouten, K.I.M., Semantics-driven Aspect-based Sentiment Analysis, Promotors: Prof. F.M.G. de Jong, Prof. R. Dekker & Dr. F. Frasincar, EPS-2018-453-LIS, https://repub.eur.nl/pub/112161

Schouten, M.E., The Ups and Downs of Hierarchy: the causes and consequences of hierarchy struggles and positional loss, Promotors; Prof. D.L. van Knippenberg & Dr. L.L. Greer, EPS-2016-386-ORG, https://repub.eur.nl/pub/80059

Sihag, V., The Effectiveness of Organizational Controls: A meta-analytic review and an investigation in NPD outsourcing, Promotors: Prof. J.C.M van den Ende & Dr. S.A. Rijsdijk, EPS-2019-476-LIS, https://repub.eur.nl/pub/115931

Smit, J., Unlocking Business Model Innovation: A look through the keyhole at the inner workings of Business Model Innovation, Promotor: Prof. H.G. Barkema, EPS-2016-399-S&E, https://repub.eur.nl/pub/93211

Straeter, L.M., Interpersonal Consumer Decision Making, Promotors: Prof. S.M.J. van Osselaer & Dr. I.E. de Hooge, EPS-2017-423-MKT, https://repub.eur.nl/pub/100819

Stuppy, A., Essays on Product Quality, Promotors: Prof. S.M.J. van Osselaer & Dr. N.L. Mead. EPS-2018-461-MKT, https://repub.eur.nl/pub/111375

Subasi, B., Demographic Dissimilarity, Information Access and Individual Performance, Promotors: Prof. D.L. van Knippenberg & Dr. W.P. van Ginkel, EPS-2017-422-ORG, https://repub.eur.nl/pub/103495

Suurmond, R., In Pursuit of Supplier Knowledge: Leveraging capabilities and dividing responsibilities in product and service contexts, Promotors: Prof. J.Y.F Wynstra & Prof. J. Dul. EPS-2018-475-LIS, https://repub.eur.nl/pub/115138

Szatmari, B., We are (all) the champions: The effect of status in the implementation of innovations, Promotors: Prof. J.C.M van den Ende & Dr. D. Deichmann, EPS-2016-401-LIS, https://repub.eur.nl/pub/94633

Toxopeus, H.S., Financing sustainable innovation: From a principal-agent to a collective action perspective, Promotors: Prof. H.R. Commandeur & Dr. K.E.H. Maas. EPS-2019-458-S&E, https://repub.eur.nl/pub/114018

Turturea, R., Overcoming Resource Constraints: The Role of Creative Resourcing and Equity Crowdfunding in Financing Entrepreneurial Ventures, Promotors: Prof. P.P.M.A.R Heugens, Prof. J.J.P. Jansen & Dr. I. Verheuil, EPS-2019-472-S&E, https://repub.eur.nl/pub/112859

Valogianni, K., Sustainable Electric Vehicle Management using Coordinated Machine Learning, Promotors: Prof. H.W.G.M. van Heck & Prof. W. Ketter, EPS-2016-387-LIS, https://repub.eur.nl/pub/93018

Vandic, D., Intelligent Information Systems for Web Product Search, Promotors: Prof. U. Kaymak & Dr. Frasincar, EPS-2017-405-LIS, https://repub.eur.nl/pub/95490

Verbeek, R.W.M., Essays on Empirical Asset Pricing, Promotors: Prof. M.A. van Dijk & Dr. M. Szymanowska, EPS-2017-441-F&A, https://repub.eur.nl/pub/102977

Vermeer, W., Propagation in Networks:The impact of information processing at the actor level on system-wide propagation dynamics, Promotor: Prof. P.H.M.Vervest, EPS-2015-373-LIS, https://repub.eur.nl/pub/79325

Versluis, I., Prevention of the Portion Size Effect, Promotors: Prof. Ph.H.B.F. Franses & Dr. E.K. Papies, EPS-2016-382-MKT, https://repub.eur.nl/pub/79880

Vishwanathan, P., Governing for Stakeholders: How Organizations May Create or Destroy Value for their Stakeholders, Promotors: Prof. J. van Oosterhout & Prof. L.C.P.M. Meijs, EPS-2016-377-ORG, https://repub.eur.nl/pub/93016

Vlaming, R. de, Linear Mixed Models in Statistical Genetics, Promotors: Prof. A.R. Thurik, Prof. P.J.F. Groenen & Prof. Ph.D. Koellinger, EPS-2017-416-S&E, https://repub.eur.nl/pub/100428

Vries, H. de, Evidence-Based Optimization in Humanitarian Logistics, Promotors: Prof. A.P.M. Wagelmans & Prof. J.J. van de Klundert, EPS-2017-435-LIS, https://repub.eur.nl/pub/102771

Vries, J. de, Behavioral Operations in Logistics, Promotors: Prof. M.B.M de Koster & Prof. D.A. Stam, EPS-2015-374-LIS, https://repub.eur.nl/pub/79705

Wagenaar, J.C., Practice Oriented Algorithmic Disruption Management in Passenger Railways, Promotors: Prof. L.G. Kroon & Prof. A.P.M. Wagelmans, EPS-2016-390-LIS, https://repub.eur.nl/pub/93177

Wang, P., Innovations, status, and networks, Promotors: Prof. J.J.P. Jansen & Dr. V.J.A. van de Vrande, EPS-2016-381-S&E, https://repub.eur.nl/pub/93176

Wang, R., Corporate Environmentalism in China, Promotors: Prof. P.P.M.A.R Heugens & Dr. F. Wijen, EPS-2017-417-S&E, https://repub.eur.nl/pub/99987

Wasesa, M., Agent-based inter-organizational systems in advanced logistics operations, Promotors: Prof. H.W.G.M van Heck, Prof. R.A. Zuidwijk & Dr. A. W. Stam, EPS-2017-LIS-424, https://repub.eur.nl/pub/100527

Wessels, C., Flexible Working Practices: How Employees Can Reap the Benefits for Engagement and Performance, Promotors: Prof. H.W.G.M. van Heck, Prof. P.J. van Baalen & Prof. M.C. Schippers, EPS-2017-418-LIS, https://repub.eur.nl/pub/99312

Wiegmann, P.M., Setting the Stage for Innovation: Balancing Diverse Interests through Standardisation, Promotors: Prof. H.J. de Vries & Dr. K. Blind, EPS-2019-473-LIS, https://repub.eur.nl/pub/114519

Wijaya, H.R., Praise the Lord! : Infusing Values and Emotions into Neo-Institutional Theory, Promotors: Prof. P.P.M.A.R. Heugens & Prof. J.P. Cornelissen, EPS-2019-450-S&E, https://repub.eur.nl/pub/115973

Williams, A.N., Make Our Planet Great Again: A Systems Perspective of Corporate Sustainability, Promotors: Prof. G.M. Whiteman & Dr. S. Kennedy, EPS-2018-456-ORG, https://repub.eur.nl/pub/111032

Witte, C.T., Bloody Business: Multinational investment in an increasingly conflict-afflicted world, Promotors: Prof. H.P.G. Pennings, Prof. H.R. Commandeur & Dr. M.J. Burger, EPS-2018-443-S&E, https://repub.eur.nl/pub/104027

Ye, Q.C., Multi-objective Optimization Methods for Allocation and Prediction, Promotors: Prof. R. Dekker & Dr. Y. Zhang, EPS-2019-460-LIS, https://repub.eur.nl/pub/116462

Ypsilantis, P., The Design, Planning and Execution of Sustainable Intermodal Port-hinterland Transport Networks, Promotors: Prof. R.A. Zuidwijk & Prof. L.G. Kroon, EPS-2016-395-LIS, https://repub.eur.nl/pub/94375

Yuan, Y., The Emergence of Team Creativity: a social network perspective, Promotors: Prof. D. L. van Knippenberg & Dr. D. A. Stam, EPS-2017-434-ORG, https://repub.eur.nl/pub/100847

Yuferova, D., Price Discovery, Liquidity Provision, and Low-Latency Trading, Promotors: Prof. M.A. van Dijk & Dr. D.G.J. Bongaerts, EPS-2016-379-F&A, https://repub.eur.nl/pub/93017

Zhang, Q., Financing and Regulatory Frictions in Mergers and Acquisitions, Promotors: Prof. P.G.J. Roosenboom & Prof. A. de Jong, EPS-2018-428-F&A, https://repub.eur.nl/pub/103871

Zuber, F.B., Looking at the Others: Studies on (un)ethical behavior and social relationships in organizations, Promotor: Prof. S.P. Kaptein, EPS-2016-394-ORG, https://repub.eur.nl/pub/94388