Shared Mobility Optimization in Large Scale Transportation Networks:

Methodology and Applications

by

Monirehalsadat Mahmoudi

A Dissertation Presented in Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy

Approved February 2018 by the
Graduate Supervisory Committee:
Xuesong Zhou, Chair
Pitu B. Mirchandani
Harvey J. Miller
Ram M. Pendyala

ARIZONA STATE UNIVERSITY

May 2018

ABSTRACT

Optimization of on-demand transportation systems and ride-sharing services involves solving a class of complex vehicle routing problems with pickup and delivery with time windows (VRPPDTW). Previous research has made a number of important contributions to the challenging pickup and delivery problem along different formulation or solution approaches. However, there are a number of modeling and algorithmic challenges for a large-scale deployment of a vehicle routing and scheduling algorithm, especially for regional networks with various road capacity and traffic delay constraints on freeway bottlenecks and signal timing on urban streets. The main thrust of this research is constructing hyper-networks to implicitly impose complicated constraints of a vehicle routing problem (VRP) into the model within the network construction. This research introduces a new methodology based on hyper-networks to solve the very important vehicle routing problem for the case of generic ride-sharing problem. Then, the idea of hyper-networks is applied for (1) solving the pickup and delivery problem with synchronized transfers, (2) computing resource hyper-prisms for sustainable transportation planning in the field of time-geography, and (3) providing an integrated framework that fully captures the interactions between supply and demand dimensions of travel to model the implications of advanced technologies and mobility services on traveler behavior.

To my parents

ACKNOWLEDGMENTS

I would like to thank my wonderful parents for always being supportive through every stage of my life, no matter how far away they are. I am thankful to my husband, Alireza, for helping me out to handle the burden of a doctoral student life. I am thankful for all of the times he listened to my concerns, no matter what, to make me feel better.

I would like to express my deepest gratitude to my adviser, Dr. Xuesong Zhou, for providing me with a great research topic for my dissertation, and for giving constant help and guidance during the program. I would also like to thank the rest of my committee: Dr. Harvey J. Miller, Dr. Pitu B. Mirchandani, and Dr. Ram Pendyala for helping me out with my dissertation.

I would like to thank my friends and research mates who have made research always engaging and exciting at Arizona State University, including but not limited to Jiangtao Liu and Peiheng Li from the School of Sustainable Engineering and the Built Environment at Arizona state University, Dr. Junhua Chen from the School of Traffic and Transportation at Beijing Jiaotong University, Dr. Tie Shi and Dr. Yongxiang Zhang from the School of Transportation and Logistics at Southwest Jiaotong University, Jeffrey D. Taylor from the Department of Civil and Environmental Engineering at the University of Utah, and Dr. Ying Song from the Department of Geography, Environment and Society at the University of Minnesota.

This work has been financially supported by National Science Foundation (NSF) grant # CMMI–1538105, NSF grant # BCS–1224102, NSF grant # CMMI–1663657, and a U.S. University Transportation Center project titled “scheduling and managing self-driving cars for enhanced transportation system mobility and safety”.

TABLE OF CONTENTS

APPENDIX Page

LIST OF FIGURES

Figure Page

Figure Page

Figure Page

Figure Page

Figure Page

LIST OF TABLES

Table Page

GLOSSARY

ABM	Activity-based Models
AC	Ant Colony
CG	Column Generation
CO	Carbon Monoxide
CO2	Carbon Dioxide
CPU	Central Processing Unit
DP	Dynamic Programming
DTA	Dynamic Traffic Assignment
DTALite	Dynamic Traffic Assignment-Light weight
EV	Electric Vehicle
GA	Genetic Algorithm
GPS	Global Positioning System
HC	Hydrocarbon
HOV	High-occupancy Vehicles
HOT	High-occupancy Toll
LR	Lagrangian Relaxation
MOVES	Motor Vehicle Emission Simulator
NOx	Nitrogen Oxides
NP-hard	Non-deterministic Polynomial-time hard
NTP	Network-time Prism
OD	Origin-destination

GLOSSARY

OpenAMOS	Open Source Activity Mobility Simulator
OpenMP	Open Multi-Processing
O2O	Online-to-offline
PDPT	Pickup and Delivery Problem with Transfers
PDPTW	Pickup and Delivery Problem with Time Window
POI	Point Of Interest
PSO	Particle Swarm Optimization
RAM	Random Access Memory
RHP	Resource Hyper-prism
SA	Simulated Annealing
SST	State-space-time
ST	Space-time
STP	Space-time Prism
STR	Space-time-resource
TNC	Transportation Network Companies
TS	Tabu Search
TSP	Traveling Salesman Problem
USEPA	US Environmental Protection Agency
VRP	Vehicle Routing Problem
VRPPD	Vehicle Routing Problem with Pickup and Delivery
VRPPDTW	Vehicle Routing Problem with Pickup and Delivery and Time Windows

GLOSSARY

VRPTW	Vehicle Routing Problem with Time Windows
VSP	Vehicle Specific Power

CHAPTER 1

INTRODUCTION

1.1. Motivations

As population and personal travel activities continue to increase, traffic congestion has remained as one of the major concerns for transportation system agencies with tight resource constraints. The next generation of transportation system initiatives aims to integrate various demand management strategies and traffic control measures to actively achieve mobility, environment, and sustainability goals. Various approaches hold promises of reducing the undesirable effects of traffic congestion due to driving-alone trips. In this research, we mainly focus on one of these approaches which is coordinated transportation and demand-responsive transit services.

In general, coordinated transportation services consist of three different levels of service: ride-hailing, ride-sharing without transfer, and ride-sharing with one or more than one synchronized transfer. Ride-hailing is a level of coordinated service in which a passenger hires a driver to get a transportation service for a fee, and the driver is supposed to deliver the passenger to exactly where he needs to go. Traditional taxi companies offer this form of transport. The way by which a passenger hails a car can be listed as follows: a passenger can hail a taxi from the street, call up a transport service on the phone, or hail a car from an app by his cellphone.

Ride-sharing without transfer is another level of coordinated transportation service which is slightly different from ride-hailing. In this mode of transportation, similar to the ride-hailing, a passenger hires a driver to take him exactly where he needs to go, but the passenger may share his ride with one or more than one passenger. A broad range of transportation network companies like Uber, Lyft, and Sidecar offers this type of transport service by the aid of three recent technological advances: (1) Global Positioning System (GPS) navigation devices, (2) smartphones, and (3) social networks.

The third level of coordinated transportation service is ride-sharing with synchronized transfers. In general, transfers are used to provide more efficient transportation networks by reducing the operational costs, as well as making more flexible routes available for passengers. A large number of daily trips is classified in this category. For instance, in multi-modal transit, a passenger may use two or more transport modes for his trip (e.g. train and bus). Multi-modal transit provides convenient and economical connection of various modes to make a complete journey from origin to destination. Another example of this type of transportation service can be the first mile/last mile transport of the commuters who need to go from an origin to a transit station and then from the station at the other end of the trip to a final destination. Ride-sharing between households or fellow workers is another example of ride-sharing with synchronized transfers. In this case, members of a family or any other social group arrange their trips informally and share their travel information such as departure time, stops, and transfer points among themselves.

In Fig. 1.1, different levels of coordinated transportation service have been shown by an example in which six passengers with different origins, destinations, and departure time windows have called for service. As shown in Fig. 1.1, in the case of ride-sharing with synchronized transfers, the vehicles' capacity can be utilized more in comparison to other types of coordinated transportation service.

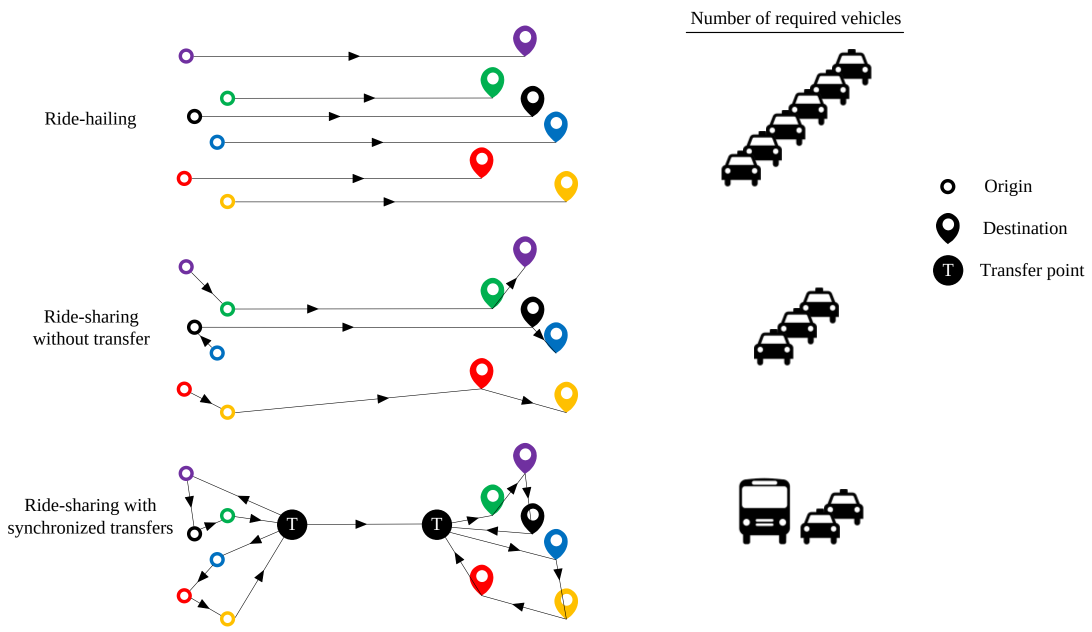

Number of required vehicles
Ride-hailing
Ride-sharing without transfer
Ride-sharing with synchronized transfers
T
Origin
Destination
Transfer point

Fig. 1.1. Different levels of coordinated transportation service.

The ride-sharing problem can be mathematically modeled by one of the well-known optimization problems which is the vehicle routing problem with pickup and delivery (VRPPD). Previous research has made a number of important contributions to this challenging problem along different formulation or solution approaches. However, there are a number of modeling and algorithmic challenges for a large-scale deployment of a vehicle routing and scheduling algorithm, especially for regional networks with various road capacity and traffic delay constraints on freeway bottlenecks and signal timing on urban streets.

In the field of operations research, a few previous research directly considers the underlying transportation network with time of day traffic congestion; however, other studies defines the pickup and delivery problem with time-windows (PDPTW) on a directed graph containing customers' origin and destination locations only which are connected by some links that are representative of the shortest distance or least travel time routes between origin–destination pairs. That is, with each link, there are associated fixed routing cost and travel time between the two service nodes. Moreover, despite its great practicality, due to the existence of various linear and non-linear constraints in this problem, this problem is hard to solve such that the largest instances that can be solved by the most effective exact algorithms proposed so far contains about 500 customers.

This problem has been also studied in the field of transportation engineering. The existing research in this field aims to study this problem on physical transportation-based networks in which travel time on each transportation link may vary over the time of a day (due to the traffic congestion in different times of day) or over the occupied seats of a vehicle (e.g. high-occupancy vehicles (HOV) or high-occupancy toll (HOT) lanes). However, most solution methods for solving this problem are heuristic-based and lack a sound optimization-based foundation.

In the first phase of this research, in order to improve the solution quality and computational efficiency of on-demand transportation systems and dynamic ride-sharing services, especially for large-scale real-world transportation networks, we propose a new mathematical programming model for the vehicle routing problem with pickup and delivery with time windows (VRPPDTW) that can fully recognize time-dependent link travel time caused by traffic congestion at different times of day.

The main idea of our solution approach is that we dynamically generate multi-dimensional networks in which the corresponding constraints are implicitly considered into the model. In the first phase of this research, we introduce a new methodology based on hyper-networks to solve the very important vehicle routing problem (VRP) for the case of generic ride-sharing problem. We apply the idea of hyper-networks for solving the ride-sharing problem with synchronized transfers in the second phase, computing hyper-prisms for sustainable transportation planning in the third phase, and presenting an activity-travel based vehicle scheduling framework in the fourth phase of this dissertation. Fig. 1.2 illustrates a summary of what we intend to present in this research.

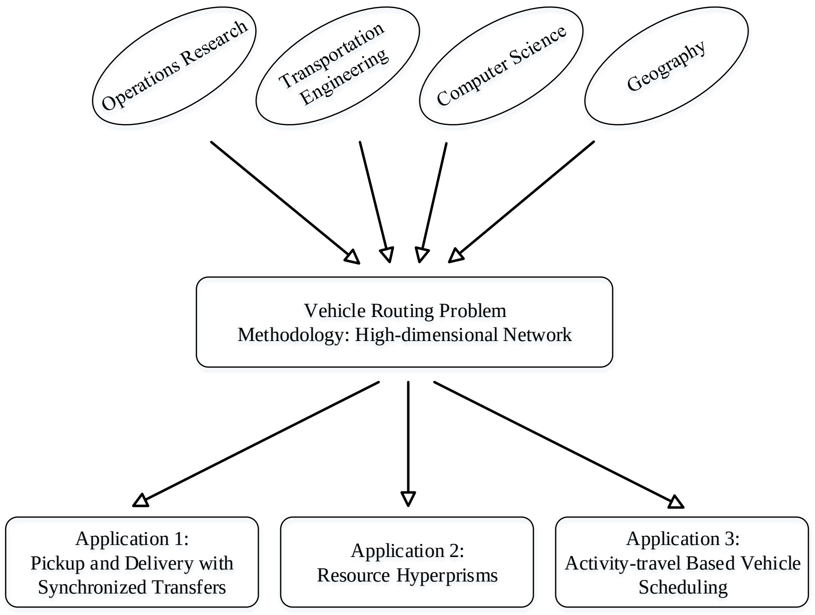

Operations Research
Transportation Engineering
Computer Science
Geography
Vehicle Routing Problem
Methodology: High-dimensional Network
Application 1:
Pickup and Delivery with Synchronized Transfers
Application 2:
Resource Hyperprisms
Application 3:
Activity-travel Based Vehicle Scheduling

Fig. 1.2. Our proposed high-dimensional network representation as the new methodology for solving the VRP and corresponding three applications in the field of operations research, time-geography, and travelers' behavior analysis.

The second phase of this research is dedicated to the more complicated version of pickup and delivery problem which is pickup and delivery with synchronized transfers. What motivated us to study the pickup and delivery problem with transfers was the fast-growing ride-sharing mode of transportation. In the last decade, ride-sharing companies have introduced a new mode of transportation which is much more convenient than public transit and less expensive than taxi. By introducing connected and autonomous vehicles to this mode of transportation and eliminating the cost of hiring drivers, it is expected that in the near future, ride-sharing becomes a good substitution of public transit for daily trips of middle-class families. Knowing passengers' trip itinerary in advance and offering incentives to those passengers who are flexible about their departure/arrival time and number of stops during their trip will help service providers to schedule more efficient trips for serving their customers.

The third phase of this study focuses on resource constrained space-time prims and their functionality on multi-modal accessibility analysis. Accessibility is the ease of obtaining desired destinations, activities, or services in an environment. A common accessibility measure in basic and applied transportation science is the space-time prism (STP) and the network-time prisms (NTPs): these are the envelopes of all possible paths between two locations and times in planar space and transportation networks, respectively. STPs and NTPs focus on time as the scarce resource limiting accessibility. However, other resource constraints can constrain space-time accessibility, such as limits or "budgets" for energy, emissions, or monetary expenses. This phase of this research extends NTPs to include other resource constraints in addition to time. Network-based *resource hyper-prisms* (RHPs) incorporate other resource constraints into NTP, capturing the trade-offs between time and other resources in determining space-time accessibility.

We also apply our proposed high-dimensional network representation for the fourth and last phase of this research to propose an activity-travel based vehicle scheduling framework. The rapidly growing popularity of transportation network companies coupled with autonomous vehicle technologies, could potentially redefine the way in which individuals schedule and execute their activities and also the way in which travel demand is managed by network operators. For the traveler, the freedom from having to drive could lead to more flexible activity schedules and increased productivity while travelling. On the other hand, network operators could handle demand by incentivizing/dis-incentivizing travel during a certain portion of the day (similar to surge pricing by Uber), or along a specific route. There is growing interest in the field to study incentive-based demand management strategies. It is therefore of critical importance to understand and accurately depict these transformative technologies and their implications for activity-travel patterns in travel demand model systems.

While the integrated models developed so far address modeling needs for the current array of travel options (modes, demand management strategies, etc.), they do not adequately handle emerging transportation technologies (ride-sharing services, autonomous vehicle technologies) that are increasingly penetrating the marketplace. Moreover, activity-based models (ABMs) still operate based on zonal level information (such as skims, by time-of-day) provided by dynamic traffic assignment models. The activity-based models are oblivious to network logistics such as availability of ride-sourcing options and incentives/disincentives customized to specific trips/travelers. On the other hand, vehicle routing problems, used to depict ride-sharing services in dynamic traffic assignment (DTA) models, view travel as disjoint trips that are independent of each other. The solutions to VRP are typically optimization-based and lack a sound behavioral foundation. Solutions to VRP in the standard dynamic traffic assignment models are often aimed at serving the maximum number of trip requests without taking into consideration the precedence constraints (or linkages) between the trips. This vital behavioral constraint is ignored in the VRP optimization techniques incorporated in dynamic traffic assignment models. The last phase of this research suggests an integrated framework that fully captures the interactions between supply and demand dimensions of travel to model the implications of advanced technologies and mobility services on traveler behavior.

1.2. Objectives

The main thrust of this research is constructing hyper-networks to implicitly impose complicated constraints of a VRP into the model within the network construction. The main objectives of this dissertation are listed as follows.

1. We introduce a new methodology based on hyper-networks to solve the very important vehicle routing problem for the case of generic ride-sharing problem.

2. We apply the idea of hyper-networks for solving the pickup and delivery problem with synchronized transfers on real-world data sets.
3. We apply the idea of hyper-networks for computing hyper-prisms for sustainable transportation planning in the field of time-geography.
4. We apply the idea of hyper-networks to provide an integrated framework that fully captures the interactions between supply and demand dimensions of travel to model the implications of advanced technologies and mobility services on traveler behavior.

1.3. Overview of Our Proposed Methods

We construct hyper-networks by adding ‘time’ and ‘state’ as new dimensions to the physical transportation networks. In the ride-sharing problem, presented in Chapter 3, the state is the carrying status of passengers in the vehicles, enabling us to track the service status of passengers at any time of day. In the ride-sharing problem with synchronized transfers, presented in Chapter 4, the state is the cumulative service status of passengers in the vehicles by which the service status of passengers are tracked cumulatively at any location and time of day.

In Chapter 5, we apply the concept of hyper-networks in the field of time-geography by adding the resource as a new dimension to the space-time networks. In this case, the level of resource is interpreted as the state which can be monitored for any agent (i.e. a gasoline passenger car or an electric vehicle) at any time of day. Fuel, money, carbon emission, and transportation mode are a few out of many types of resources in the real–world transportation systems.

We also introduce a new state dimension, called the 'under-service trip request' state, to the vehicle scheduling model in order to track the execution status of the trip requests at any time and transportation node. We also construct activity-travel graphs for passengers to detect the execution of the passenger's activities. Having this, not only we can guarantee that each activity request is systematically evaluated within its time window (depending on whether it is mandatory or optional), but also ensure that the road as well as vehicle capacity constraints are not violated.

Bu introducing hyper-networks to the aforementioned problems, the problems are converted to time-dependent state-dependent least cost problems which can be solved by computationally efficient algorithms proposed in the literature (e.g. label correcting, label setting, etc.).

1.4. Organization of the Dissertation

Chapter 2 provides a comprehensive literature review for the pickup and delivery problem. In this chapter, we focus on the applications of this problem, as well as solution methods used to solve this optimization problem.

Chapter 3 contains a precise mathematical description of the PDPTW in the state-space-time (SST) networks. We present our new integer programming model for the PDPTW. Then, we will show how the main problem is decomposed to an easy-to-solve problem by the Lagrangian relaxation (LR) algorithm. Finally we provide computational results of the six-node transportation network, followed by the Chicago sketch and Phoenix regional networks to demonstrate the computational efficiency and solution optimality of our developed algorithm coded by C++. After large-scale network experiments, we conclude this chapter with discussions on possible extensions.

Chapter 4 contains the problem statement and assumptions for the pickup and delivery problem with transfers (PDPT). In this chapter, we initially explain the clustering phase and present our proposed multi-commodity network flow programming model for the PDPT. We will further explain how to improve vehicles' performance by finding optimal chains of work pieces. We also explain our motives for applying the hyper-network structure in this section. Computational results over the instances applied by Ropke and Pisinger (2006) and the real-world data set proposed by Cainiao Network (logistics service provider to Alibaba Group) are provided to demonstrate the computational efficiency of our developed algorithm coded in C++. We summarize this chapter with discussions on possible extensions.

Chapter 5 provides a brief literature review on existing time geography frameworks. This chapter describes in detail, the construction of a space-time-resource (STR) hyper-network as the foundation of our method and presents a mathematical formulation to specify the borders of a RHP, followed by a dynamic programming solution approach. We also provide results from the application of the proposed algorithm to the large–scale Chicago sketch transportation network. Discussion, concluding remarks, and directions for future research form the final section of this chapter.

Chapter 6 describes in detail, the construction of the activity-travel graphs for passengers and SST networks for vehicles, and presents the mathematical formulation of the time-discretized multi-commodity network flow model, as well as the solution approach. We also provide results from the application of the proposed algorithm to the Phoenix subarea transportation network. Discussion, concluding remarks, and directions for future research form the last section of this chapter.

CHAPTER 2

LITERATURE REVIEW

2.1. Background

The Vehicle Routing Problem is one of the most well-studied combinatorial optimization problems in which we are looking for the optimal set of routes to be traveled by a fleet of vehicles to serve a given set of passengers. This problem was introduced by Dantzig and Ramser in 1959 for the first time. A few years later, in 1964, a heuristic was proposed by Clarke and Wright to improve the Dantzig-Ramser approach. Following these two seminal papers, thousands of mathematical models and algorithms were proposed for the optimal and approximate solution of the different versions of the VRP. We also know that hundreds of software packages for solving the various real-world VRPs are now available on the market. To name a few out of many, WorkWave Route Manager, Drivewyze PreClear, DIRECTOR Fleet Software, Route4Me, MyRouteOnline, Routific, Omnitracs Roadnet Routing, Speedy Route, GetSwift, Abivin vRoute, TourSolver, Route Optimizer, RouteSavvy, Road Warrior, RouteXL, Mapotempo Web, and mobi.Route. Despite its great practicality, the VRP is hard to solve such that the largest VRP instances that can be solved by the most effective exact algorithms proposed so far contains about 500 customers (Baldacci et al., 2011), while larger instances may be solved to optimality only in particular cases.

In this research, we mostly focus on the VRPPDTW or simply say, the PDPTW, which is a generalized version of the VRP with time windows (VRPTW). In the PDTW, each transportation request is a combination of pickup at the origin node and drop-off at the destination node. The PDPTW contains all constraints in the VRPTW plus added constraints in which either pickup or delivery has given time windows, and the service for each request must be performed by a single vehicle. These impose visiting each pickup and drop-off location exactly once during their given departure and arrival time windows, coupling the pickup and corresponding delivery stops on the same vehicle routes, and visit precedence among each pickup stop and its associated drop-off stop. Other constraints which are common between the PDPTW and VRPTW are as follows: not exceeding the capacity of vehicles, depot constraints that ensure vehicles start their route from their starting depot and end it to their ending depot, and resource constraints on the number of drivers and vehicle types.

2.2. Applications of the PDPTW

Several applications of the PDPTW have been reported in road, maritime, and air transportation environments, to name a few out of many, Fisher et al. (1982), Bell et al. (1983), Savelsbergh and Sol (1998), Wang and Regan (2002), and Zachariadis et al. (2015), Shiri and Huynh (2016), and Shiri and Huynh (2017) in road cargo routing and scheduling; Psaraftis et al. (1985), Fisher and Rosenwein (1989), Christiansen (1999), Christiansen et al. (2004), Christiansen et al. (2007), and Rodrigues et al. (2016) in sea cargo routing and scheduling; and Solanki and Southworth (1991), Solomon et al. (1992), Rappoport et al. (1992), Rappoport et al. (1994), Azadian et al. (2012) in air cargo routing and scheduling. Further applications of the VRPPDTW can be found in door-to-door transportation services for elderly or handicapped people (Jaw et al., 1986; Alfa, 1986; Madsen et al. 1995; Ioachim et al., 1995; and Toth and Vigo, 1997; Borndörfer et al. 1997; Colorni and Righini 2001; Diana and Dessouky 2004; Rekiek et al. 2006; Melachrinoudis et al. 2007), school bus routing and scheduling (Swersey and Ballard, 1983; and Bramel and Simchi-Levi, 1995), and ride-sharing (Hosni et al., 2014; and Wang et al., 2015, Mahmoudi and Zhou, 2016). Recently, Furuhata et al. (2013) offers an excellent review and provides a systematic classification of emerging ride-sharing systems.

What makes the application of the PDPTW in transporting people different from other routing problems is the human perspective. When passengers are transported, improving user convenience and quality of service must be balanced against minimizing operating costs. Quality of service can be measured by the time duration passengers wait until their service start (passengers waiting time), the time duration passengers spend in vehicles during their ride (passengers ride time), or difference between actual and desired delivery times. Moreover, vehicle capacity becomes an important constraint when transporting passengers, while it is often ignored in other applications of the PDPTW, particularly those related to the collection and delivery of letters and small parcels. In this research, we mainly focus on the PDPTW for people transportation and more specifically, the application of the PDPTW in ride-sharing mode of transportation.

The PDPTW can operate based on a static or a dynamic mode. In the static mode (off-line requests), all transportation requests are known in advance, whereas in the dynamic mode (on-line requests), transportation requests gradually call for service during the day and vehicle routes must be adjusted in the real-time manner to satisfy the demand. In practice, we may have a combination of on-line and off-line transportation requests.

2.3. Assumptions

In order to simplify the PDPTW, several assumptions on the attributes of vehicles serving passengers have been considered which can be found in the literature. For example, some primary research assume there is only one vehicle available to serve customers (e.g. Psaraftis, 1980; Psaraftis, 1983; Sexton and Bodin, 1985a and 1985b; Desrosiers et al. 1986 and Van Der Bruggen et al. 1993). Some studies assume the main consideration of the problem is minimizing the total number of vehicles subject to satisfying all demands; thus, total number of vehicles available for serving is unknown (e.g. Dumas et al. 1989; Desrosiers et al. 1991; Ioachim et al. 1995; Rekiek et al. 2006; Lou and Schonfeld, 2007 and 2011). Some assume vehicles serving passengers are homogeneous (e.g. Nanry and Barnes, 2000; Angelelli and Mansini, 2002; Männel and Bortfeldt, 2016; Veenstra et al. 2017). Homogeneous vehicles are those with the same capacity, type, origin and destination depots, and work shift. Speaking of vehicles type, vehicles may be designed to carry wheelchairs only, serve off-line transportation requests only, or serve both off-line and on-line requests. Back to the assumptions on the characteristics of vehicles, one can assume there are multiple vehicles (either homogeneous or non-homogeneous) available for service, but based at a single depot (e.g. Bodin and Sexton, 1986; Desrosiers et al. 1988; Ioachim et al. 1995; Cordeau and Laporte 2003; Diana and Dessouky, 2004; Parrah et al. 2009).

We can also consider several assumptions on passengers' preferences. For example, one can assume passengers have choice to determine either their departure or arrival time window not both. Another can assume that there is no restriction on passengers' ride and waiting times as long as passengers' departure/arrival time window constraints are not violated. The other assumption can be the one in which passengers have no preference about vehicles type and/or drivers serving them. Finally, one can assume passengers do not put any restriction on the number of passengers sharing their ride with them.

The PDPTW can be approached as soft or hard time windows. In the PDPTW with soft time windows, vehicles are allowed to arrive early/late at passengers' pickup/drop-off locations, but time window violations (earliness and tardiness) are penalized in the objective function. In contrast, in the PDPTW with hard time windows, earliness and tardiness are generally forbidden. From the practical point of view, for situations in which vehicles may not have any available place at customer locations to wait, the soft variant prevails.

The objective function of the PDPTW may also vary from one study to another based on what main consideration of the problem is. For example, some studies aim to satisfy all demands with less number of vehicles, while others target to maximize the number of requests that can be served by a fixed number of vehicles. Therefore, the former problem is minimizing costs subject to full demand satisfaction, while the latter one is maximizing satisfied demand subject to vehicles availability. In fact, the latter case is more practical.

One of the ways by which lack of vehicles can be handled is surge pricing. Surge pricing is how rideshare companies aim to control supply and demand and happens when there is a high demand for vehicles (i.e. lots of passengers are looking for a ride in the same area), while there are not enough vehicles to satisfy all the passengers. The goal of surge pricing is to incentive vehicles to perform trips during the busiest hours of the day. Surge pricing fixes this excess demand by applying a multiplier on every fare, therefore raising prices by certain percentages. As a result, some passenger will opt to not pay the higher fare, making more vehicles available for those passengers who are willing to pay the surge. Surge pricing can happen at any time of the day, but it is most common during rush hour, bad weather, holidays, and weekends - all times when there is a sense of urgency to get a ride. Another way to handle deficit in supply is serving some of the demand with the available vehicles and then using extra vehicles (e.g. virtual vehicles (taxis) with higher operating cost) if necessary. Note that operating costs may include fuel, maintenance, depreciation, insurance costs, and more importantly, cost of hiring full-time or part-time drivers. Other objective functions observed in the literature include but not limited to: minimization of difference between actual and desired delivery times (e.g. Bodin and Sexton, 1986), minimization of differences between actual and shortest possible ride times (e.g. Bodin and Sexton, 1986), minimization of total route duration (e.g. e.g. Dumas et al. 1989; Desrosiers et al. 1991; Ioachim et al. 1995), minimization of total route length (e.g. Cordeau and Laporte, 2003), minimization of vehicles idle time (e.g. Diana and Dessouky, 2004), minimization of user inconvenience/dissatisfaction (e.g.

Coslovich et al. 2006; Melachrinoudis et al. 2007), or a weighted combination of those mentioned above.

2.4. Solution Methods

Several solution methods for the PDPTW have been conducted before. There are two categories of solution approach: heuristics and optimization-based solutions. In this section, we start with the heuristic methods and then we will go through the details of optimization-based methods which provides exact solutions for the PDPTW.

2.4.1. Heuristics

2.4.1.1. Insertion Heuristic

Insertion heuristic starts with a route on small subsets of requests, and then extends this route by inserting the remaining requests one after the other until all requests have been inserted. Wilson et al. (1971), Wilson and Weissberg (1976), and Wilson and Colvin (1977) proposed an insertion heuristic in which the concept of building tours is introduced through sequential insertion of passengers. Roy et al. (1984a, 1984b) applied similar insertion heuristic for the problem of transporting disabled persons. In their project, majority of passengers call for service in advance. Knowing the information related to passengers' origin and destination locations as well as their desired pickup and delivery time beforehand, their algorithm generates initial routes for vehicles starting at the beginning of the day, then the algorithm inserts new requests in the existing routes or generates new routes if necessary. Jaw et al. (1986) developed an insertion heuristic, where transportation requests are sorted based on their earliest pickup time, and then requests are inserted in vehicles timetable considering a weighted combination of least operational costs and users' inconvenience. Feasible insertion is obtained by the aid of schedule blocks (continuous periods of time for active vehicles) between two successive periods of idle vehicle slack time. Madsen et al. (1995) developed a generalized version of the insertion heuristic proposed by Jaw et al. (1986) in which the algorithm can be applied on-line in a dynamic environment. Dethloff (2002), Diana and Dessouky (2004), Lu and Dessouky (2006), Quadrifoglio et al. (2007), Sungur et al. (2008), Caris and Janssens (2009), Masson et al. 2013, Coltin and Veloso (2014), and Wang et al. (2016) are a few out of many research applied insertion heuristic for the PDPTW.

2.4.1.2. Local Search Heuristic

Local search is an iterative algorithm that moves from one solution to another in the search space by applying local changes, until a near optimal solution is found or a time bound is elapsed. Local search heuristic initially introduced by Lin (1965) and Lin and Kernighan (1973) for the Traveling Salesman Problem. A few years later, Psaraftis (1983) applied a local search heuristic for solving the PDPTW. Van der Bruggen et al. (1993) developed a local search heuristic based on a variable-depth search. The algorithm consists of two phases: construction phase in which the algorithm searches for feasible routes and improvement phase in which the algorithm considers feasible solutions only to improve the final solution. Gendreau et al. (2006), Pisinger and Ropke (2010), Subramanian et al. (2010), Ribeiro and Laporte (2012), Masson et al. (2013), and

Grangier et al. (2016) are a few out of many research applied local search heuristic for the PDPTW.

2.4.1.3. Clustering Algorithm

Clustering passengers based on their geographical proximity is broadly used either a priori or in parallel with the routing process. Cullen et al. (1981) proposed a cluster-first, route-second heuristic in which the clustering and routing sub-problems are solved by column generation (CG) technique. A few years later, Bodin and Sexton (1986) also developed another cluster-first, route-second heuristic for the PDPTW. Passengers are first partitioned into clusters, and then, by using the algorithm proposed by Sexton and Bodin (1985a, 1985b), the algorithm constructs a tour on each cluster. Finally, the algorithm swaps passengers between routes and performs route re-optimizations.

In fact, finding high-quality clusters without having some levels of routing information is a difficult task. That is why Dumas et al. (1989) introduced the concept of mini-clusters, where passengers with the spatio-temporal closeness are clustered together. In their algorithm, a heuristic provides a set of mini-clusters. Then, a CG algorithm is used to optimally combine mini-clusters into vehicle routes. Finally, the algorithm performs route re-optimizations in order to obtain optimal routing and scheduling for each vehicle. Desrosiers et al. (1991) also proposed another way of constructing mini-clusters by a parallel insertion method based on spatio-temporal proximity of the requests. Ioachim et al. (1995) later applied an optimization-based technique instead of a heuristic for constructing mini-clusters. Pankratz (2005), Bard and Jarrah (2009), Qu and

Bard (2012), and Masson et al. (2013) are a few out of many examples of studies applied clustering algorithm for the PDPTW.

2.4.2. Metaheuristics

2.4.2.1. Tabu Search

One of the metaheuristics used for solving the PDPTW is tabu search (TS). Tabu search is a local search technique in which a potential solution to a problem is taken, and its immediate neighbors are scanned in the hope of finding an improved solution. If a potential solution has been previously visited within a certain short-term period, or if it has violated a rule, it is marked as “tabu” (forbidden) so that the algorithm does not consider that possibility repeatedly. Gendreau et al. (1998) proposed an adaptive memory-based tabu search for the dynamic PDPTW in which a neighborhood structure based on the concept of ejection chains is used. The concept of ejection chains is defined as follows: a transportation request is chosen, ejected from its route, and inserted to another route, where another request is pushed to move to another route. Toth and Vigo (1997) obtained further improvements by applying a tabu thresholding post-optimization phase after the parallel insertion step. Cordeau and Laporte (2003) proposed a tabu search heuristic for the static PDPTW. They developed a procedure for neighborhood evaluation to minimize route duration and ride time by adjusting the visit time of the nodes on the routes. Attanasio et al. (2004) suggested a number of parallel implementations of a tabu search heuristic previously developed by Cordeau and Laporte (2003) for the static PDPTW in order to solve this problem for the dynamic mode. Chen and Wu (2005), Crispim and Brandão (2005), Montané and Galvão (2006), Bianchessi and Righini (2007), Melachrinoudis et al. (2007), and Kirchler and Calvo (2013) are a few out of many examples of studies applied TS for solving the PDPTW.

2.4.2.2. Genetic Algorithm

The other metaheuristic used for solving the PDPTW is genetic algorithm (GA). Genetic algorithm is a technique based on a natural selection process that mimics biological evolution. The algorithm repeatedly modifies a population of individual solutions. At each step, the genetic algorithm randomly selects individuals from the current population and uses them as parents to produce the children for the next generation. Over successive generations, the population “evolves” toward an optimal solution. Rekiek et al. (2006) proposed a genetic algorithm for the clustering phase and an insertion algorithm for the routing phase. Jorgensen et al. (2007) developed a genetic algorithm heuristic to construct clusters and spatio-temporal nearest neighbor procedure to generate routes. Jung and Haghani (2000), Zhao et al. (2009), Tasan and Gen (2012), Wang and Chen (2012), Lin et al. (2014), and Cherkesly et al. (2015) are a few out of many examples of studies applied GA for the PDPTW.

2.4.2.3. Simulated Annealing

Simulated annealing (SA) is another metaheuristic used for solving the PDPTW. The simulated annealing algorithm was originally inspired from the process of annealing in metal work. Annealing is a procedure involves heating and cooling a material to alter its physical properties due to the changes in its internal structure. As the metal cools down, its new structure becomes fixed, consequently causing the metal to keep its newly obtained properties. In simulated annealing, we consider a temperature variable to simulate this heating process. We initially set the temperature variable high and then allow it to slowly ‘cool’ as the algorithm runs. While the temperature variable is high, the algorithm will be allowed, with more frequency, to accept solutions that are worse than our current solution. This gives the algorithm the ability to jump out of any local optimums it finds itself in early on in execution. As the temperature is reduced, so is the chance of accepting worse solutions, therefore allowing the algorithm to gradually focus in on an area of the search space in which hopefully, a close to optimum solution can be found. This gradual ‘cooling’ process is what makes the simulated annealing algorithm remarkably effective at finding a close to optimum solution when dealing with large problems which contain numerous local optimums. Van der Bruggen et al. (1993) developed a local search heuristic based on a variable-depth search, in which a simulated annealing algorithm is applied to avoid local optimums. Baugh et al. (1998) proposed cluster-first, route-second heuristic, where the clustering phase is solved by a simulated annealing heuristic and routing phase by a modified space-time nearest neighbor heuristic. Li and Lim (2001), Bent and Hentenryck (2006), Deng et al. (2009), Yu and Lin (2014), and Wang et al. (2015) are examples of research used SA for the PDPTW.

2.4.2.4. Particle Swarm Optimization

Particle swarm optimization (PSO) algorithm is one of the other metaheuristics employed for solving the PDPTW. The algorithm inspired by social behavior of bird flocking or fish schooling around food sources. At first, a flock of birds circling over an area, where they can smell a hidden source of food. The one who is closest to the food chirps the loudest and the other birds swing around in his direction. If any of the other circling birds comes closer to the target than the first, it chirps louder and the others veer over toward him. This tightening pattern continues until one of the birds happens upon the food. In this algorithm, over a number of iterations, a group of variables have their own values adjusted closer to the member whose value is closest to the target at any given moment. Ai and Kachitvichyanukul (2009), Sombuntham and Kachitvichayanukul (2010), Goksal et al. (2013), and Chen et al. (2016) are examples of research used PSO for the PDPTW.

2.4.2.5. Ant Colony

Ant colony (AC) algorithm is another metaheuristic used for solving the PDPTW. The ant colony algorithm finds optimal paths that is based on the behavior of ants searching for food. At first, ants wander randomly. When an ant finds food, it walks back to the colony leaving “markers”, called pheromones, showing that the path has food. When other ants come across the pheromones, they are likely to follow the path with a certain probability, and return and reinforce it if they eventually find food. As more ants find the path, it gets stronger until there are a couple streams of ants traveling to various food sources near the colony. Since the ants drop pheromones every time they bring food, shorter paths gets marched over more frequently, and thus the pheromone density becomes higher on shorter paths than longer ones. Over time, however, the pheromone trail starts to evaporate, thus reducing its attractive strength. Pheromone evaporation has the advantage of avoiding the convergence to a locally optimal solution. If there were no evaporation at all, the paths chosen by the first ants would tend to be excessively attractive to the following ones. In that case, the exploration of the solution space would be constrained. Gajpal and Abad (2009), Huang and Ting (2010), and Lu et al. (2016) are examples of research applied AC for solving the PDPTW.

Several other metaheuristic techniques can also be observed in the literature for solving the PDPTW. For instance, Shen et al. (1995) developed an expert consulting system using a neural network as a learning module, and Kikuchi and Donnelly (1992) and Teodorovic and Radivojevic (2000) applied fuzzy logic approach for solving the PDPTW. In the next section, we will discuss about the exact solution approaches used for solving the PDPTW to optimality.

2.4.3. Exact Solution Methods

2.4.3.1. Benders' Decomposition

The Benders' decomposition technique is named after Jacques F. Benders. The strategy behind this technique can be summarized as follows: the variables of the original problem are divided into two subsets. A master problem is solved over the first set of variables, and then for a given master problem solution, the values of the second set of variables are determined in a sub-problem. If the sub-problem determines that the value of decision variables for master problem are infeasible, then Benders cuts are generated and added to the master problem, which is then re-solved until no cuts can be generated. Since Benders decomposition adds new constraints as it progresses towards a solution, the approach is called “row generation”. Sexton and Bodin (1985a, 1985b) and Sexton and Choi (1986) decomposed the single vehicle PDPTW to a routing problem and a scheduling sub-problem, and then they applied Benders’ decomposition for both master problem and sub-problem, independently. Cortés et al. (2010) studied a problem where passengers can be transferred from one vehicle to another at specific transfer locations. Benders decomposition was their solution approach for solving the PDPTW with transfers. Contardo et al. (2012) introduced a dynamic public bike-sharing balancing problem in which the service provider dispatches vehicles to move bikes from those stations with surplus to stations with shortages in order to balance the demand and supply levels. They applied Dantzig-Wolfe decomposition and Benders’ decomposition to obtain lower bounds and feasible solutions in short computation times. Erdogan et al. (2014) studied the one-commodity pickup and delivery traveling salesman problem, and developed a branch-and-cut algorithm as well as a Benders’ decomposition scheme for solving this problem. Salazar-González and Santos-Hernández (2015) presented a flow-commodity formulation and also a Benders’ decomposition approach to solve the split-demand one-commodity pickup-and-delivery travelling salesman problem. Gendreau et al. 2015 addressed the one-commodity full truckload pickup and delivery Problem and presented three model formulations (one traveling salesman problem (TSP) formulation and two integrated formulation) for this problem. The integrated formulations were fitted for decomposition algorithms; that is why they applied the classical and the generalized Benders decomposition to these models to capture the routing and assignment composition of this problem.

2.4.3.2. Dantizg-Wolfe Decomposition

In opposition to Benders' decomposition, in Dantizg-Wolfe decomposition by using CG approach, we begin by solving the integer programming problem with a subset of columns and continue to introduce additional columns until a termination criterion proves optimality of the entire problem. Dumas et al. (1991) presented an exact algorithm for solving the PDPTW with multiple depots and different types of vehicles. Their algorithm uses a CG scheme with a constrained shortest path as a sub-problem. Sol (1994) dedicated his doctoral dissertation to the CG techniques for pickup and delivery problems. Christiansen and Nygreen (1998) proposed a mathematical formulation for the real ship planning problem which is a combination of a PDPTW and a multi-inventory model and applied a CG approach including sub-problems for ships and harbor inventories. Xu et al. (2003) formulated the PDPTW as a set partitioning problem and developed hybrid approaches that integrate CG methodology with fast heuristics for the sub-problems resulted in the CG procedure. Venkateshan and Mathur (2011) developed a specialized column-generation subroutine to reduce the combinatorial explosion in the number of routes generated by the standard column-generation subroutine and the number of nodes scanned in the branch-and-bound tree. Recently, Parragh and Schmid

(2013) proposed a hybrid CG and large neighborhood search algorithm for the PDPTW. Gschwind et al. (2017) also drafted a column-generation algorithm for the PDPTW consisting a set-partitioning master problem and an elementary shortest-path problem with resource constraints sub-problem.

2.4.3.3. Dynamic Programming

Dynamic programming (DP) breaks down a complex problem into a set of sub-problems, solving each of those just once, and saving their solutions. The next time if the same sub-problem occurs, instead of re-computing the problem, the algorithm looks up the previously computed solution to save computation time and storage space. The first published DP method for solving the single vehicle-PDPTW was developed by Psaraftis (1980). Psaraftis (1983) later modified the previous backward DP algorithm to a forward DP. Desrosiers et al. (1986) presented a DP for the single vehicle PDPTW as well. Dumas et al. (1991) also proposed a forward DP to solve a constrained shortest path problem with pickup and delivery and time windows. Bianco et al. (1994) developed an improved backward DP for the TSP in general and PDPTW in particular using a lower bound to reduce the search space. Recently, Ritzinger et al. (2016), in order to restrict the search space, presented a hybrid solution framework containing a DP and a large neighborhood search (LNS) heuristic for solving the PDPTW to deal with the curse of dimensionality. Mahmoudi and Zhou (2016) also constructed a multi-dimensional network to impose the complex constraints of the PDPTW corresponding to the validity of the time and load variables implicitly during the network construction. Then, the problem converted to a time-dependent state-dependent least cost path problem solved by a DP.

2.4.3.4. Branch and Bound

A branch-and-bound algorithm systematically enumerates candidate solutions. The strategy behind this technique can be summarized as follows: we assume an initial set of candidate solutions as the root of a tree. The algorithm explores branches of this tree representing subsets of the solution set. Before enumerating the candidate solutions of a branch, the branch is compared to upper and lower bounds on the optimal solution, and is discarded if it cannot generate a better solution than the best one found so far by the algorithm. Ruland and Rodin, 1997, Irnich (2000), Wang and Regan (2002), Hernández-Pérez and Salazar-González (2004), Cordeau (2006), Dell'Amico et al. (2006), Ropke et al. (2007), Cordeau et al. (2007), Carrabs et al. (2007), Ropke and Cordeau (2009), Cordeau et al. (2010), Cortés et al. (2010), Gutiérrez-Jarpa et al. (2010), Dumitrescu et al. (2010), Baldacci et al. (2011), Côté et al. (2012), Subramanian et al. (2013), Chemla et al. (2013), Masson et al. (2014), and Cherkesly et al. (2016) are a few out of many examples of studies using branch and bound algorithm for solving the PDPTW.

2.5. Summary

This chapter provides a comprehensive literature review for the pickup and delivery problem. In this chapter, we focus on the applications of this problem, as well as solution methods used to solve this optimization problem. In this chapter, we aim to emphasize that previous research has made a number of important contributions to the challenging pickup and delivery problem along different formulation or solution approaches. However, there are a number of modeling and algorithmic challenges for a large-scale deployment of a vehicle routing and scheduling algorithm, especially for regional networks with various road capacity and traffic delay constraints on freeway bottlenecks and signal timing on urban streets.

CHAPTER 3

FINDING OPTIMAL SOLUTIONS FOR VEHICLE ROUTING PROBLEM WITH PICKUP AND DELIVERY SERVICES WITH TIME WINDOWS

3.1. Introduction

As population and personal travel activities continue to increase, traffic congestion has remained as one of the major concerns for transportation system agencies with tight resource constraints. The next generation of transportation system initiatives aims to integrate various demand management strategies and traffic control measures to actively achieve mobility, environment, and sustainability goals. A number of approaches hold promises of reducing the undesirable effects of traffic congestion due to driving-alone trips, to name a few, demand-responsive transit services, dynamic ride-sharing, and intermodal traffic corridor management.

The optimized and coordinated ride-sharing services provided by transportation network companies (TNC) can efficiently utilize limited vehicle and driver resources while satisfying time-sensitive origin-to-destination transportation service requests. In a city with numerous travelers with different purposes, each traveler has his own traveling schedule. Instead of using his own car, the traveler can (by the aid of ride-sharing) bid and call a car just a few minutes before leaving his origin, or preschedule a car a day prior to his departure. The on-demand transportation system provides a traveler with a short waiting time even if he resides in a high-demand area. Currently, several real-time ride-sharing or, more precisely, app-based transportation network and taxi companies, such as Uber and Lyft are serving passengers in many metropolitan areas. In the long run, a fully automated and optimized ride-sharing approach is expected to handle very complex transportation supply-to-demand assignment tasks and offer a long list of benefits for transportation road users and TNC operators. These benefits might include reducing driver stress and driving cost, improving mobility for non-drivers, increasing safety and fuel efficiency, and decreasing road congestion as well as reducing overall societal energy use and pollution.

The ride-sharing problem can be mathematically modeled by one of the well-known optimization problems which is the vehicle routing problem with pickup and delivery. In this chapter, in order to improve the solution quality and computational efficiency of on-demand transportation systems and dynamic ride-sharing services, especially for large-scale real-world transportation networks, we propose a new mathematical programming model for the VRPPDTW that can fully recognize time-dependent link travel time caused by traffic congestion at different times of day. Based on the LR solution framework, we further present a holistic optimization approach for matching passengers' requests to transportation service providers, synchronizing transportation vehicle routing, and determining request pricing (e.g. through Lagrangian multipliers) for balancing transportation demand satisfaction and resource needs on urban networks.

3.2. Literature Review and Research Motivations

The VRPPDTW or simply, PDPTW, is a generalized version of the vehicle routing problem with time windows, in which each transportation request is a combination of pickup at the origin node and drop-off at the destination node (Desaulniers et al., 2002). The PDPTW under consideration in this chapter contains all constraints in the VRPTW plus added constraints in which either pickup or delivery has given time windows, and each request must be served by a single vehicle. The PDPTW may be observed as the dial-a-ride problem in the literature as well. Since the VRPTW is a non-deterministic polynomial-time hard (NP-hard) problem, the PDPTW is also NP-hard (Baldacci et al., 2011).

Several applications of the VRPPDTW have been reported in road, maritime, and air transportation environments, to name a few, Fisher et al. (1982), Bell et al. (1983), Savelsbergh and Sol (1998), Wang and Regan (2002), and Zachariadis et al. (2015) in road cargo routing and scheduling; Psaraftis et al. (1985), Fisher and Rosenwein (1989), and Christiansen (1999) in sea cargo routing and scheduling; and Solanki and Southworth (1991), Solomon et al. (1992), Rappoport et al. (1992), and Rappoport et al. (1994) in air cargo routing and scheduling. Further applications of the VRPPDTW can be found in transportation of elderly or handicapped people (Jaw et al., 1986; Alfa, 1986; Ioachim et al., 1995; and Toth and Vigo, 1997), school bus routing and scheduling (Swersey and Ballard, 1983; and Bramel and Simchi-Levi, 1995), and ride-sharing (Hosni et al., 2014; and Wang et al., 2015). Recently, Furuhata et al. (2013) offers an excellent review and provides a systematic classification of emerging ride-sharing systems.

Although clustering algorithms (Cullen et al., 1981; Bodin and Sexton, 1986; Dumas et al., 1989; Desrosiers et al., 1991; and Ioachim et al., 1995), meta-heuristics (Gendreau et al., 1998; Toth and Vigo, 1997; and Paquette et al., 2013), neural networks (Shen et al., 1995), and some heuristics such as double-horizon based heuristics (Mitrovic-Minic et al., 2004) and regret insertion heuristics (Diana and Dessouky, 2004) have been shown to be efficient in solving a particular size of PDPTW, in general, finding the exact solution via optimization approaches has still remained theoretically and computationally challenging. Focusing on the PDPTW for a single vehicle, Psaraftis (1980) presented an exact backward DP solution algorithm to minimize a weighted combination of the total service time and the total waiting time for all customers with $O(n^2 3^n)$ complexity. Psaraftis (1983) further modified the algorithm to a forward DP approach. Sexton and Bodin (1985a, b) decomposed the single vehicle PDPTW to a routing problem and a scheduling sub-problem, and then they applied Benders' decomposition for both master problem and sub-problem, independently. Based on a static network flow formulation, Desrosiers et al. (1986) proposed a forward DP algorithm for the single-vehicle PDPTW with the objective function of minimizing the total traveled distance to serve all customers. After presenting our proposed model in the later section, we will conduct a more systematical comparison between our proposed SST DP framework and the classical work by Psaraftis (1983) and Desrosiers et al. (1986).

There are a number of studies focusing on the multi-vehicle pickup and delivery problem with time windows. Dumas et al. (1991) proposed an exact algorithm to the multiple vehicle PDPTW with multiple depots, where the objective is to minimize the total travel cost with capacity, time window, precedence and coupling constraints. They applied a CG scheme with a shortest path sub-problem to solve the PDPTW, with tight vehicle capacity constraints, and a small size of requests per route. Ruland (1995) and

Ruland and Rodin (1997) proposed a polyhedral approach for the vehicle routing problem with pickup and delivery. Savelsbergh and Sol (1998) proposed an algorithm for the multiple vehicle PDPTW with multiple depots to minimize the number of vehicles needed to serve all transportation requests as the primary objective function, and minimizing the total distance traveled as the secondary objective function. Their algorithm moves toward the optimal solution after solving the pricing sub-problem using heuristics. They applied their algorithm for a set of randomly generated instances. In a two-index formulation proposed by Lu and Dessouky (2004), a branch-and-cut algorithm was able to solve problem instances. Cordeau (2006) proposed a branch-and-cut algorithm based on a three-index formulation. Ropke et al. (2007) presented a branch-and-cut algorithm to minimize the total routing cost, based on a two-index formulation. Ropke and Cordeau (2009) presented a new branch-and-cut-and-price algorithm in which the lower bounds are computed by the CG algorithm and improved by introducing different valid inequalities to the problem. Based on a set-partitioning formulation improved by additional cuts, Baldacci et al. (2011) proposed a new exact algorithm for the PDPTW with two different objective functions: the primary is minimizing the route costs, whereas the secondary is to minimize the total vehicle fixed costs first, and then minimize the total route costs.

Previous research has made a number of important contributions to this challenging problem along different formulation or solution approaches. On the other hand, there are a number of modeling and algorithmic challenges for a large-scale deployment of a vehicle routing and scheduling algorithm, especially for regional networks with various road capacity and traffic delay constraints on freeway bottlenecks and signal timing on urban streets. A few previous research directly considers the underlying transportation network with time of day traffic congestion (Kok et al. 2012, Gromicho et al. 2012) and has defined the PDPTW on a directed graph containing customers' origin and destination locations connected by some links which are representative of the shortest distance or least travel time routes between origin-destination (OD) pairs. That is, with each link, there are associated routing cost and travel time between the two service nodes. Unlike the existing offline network for the PDPTW in which each link has a fixed routing cost (travel time), our research particularly examines the PDPTW on real-world transportation networks containing a transportation node-link structure in which routing cost (travel time) along each link may vary over the time.

In order to consider many relevant practical aspects, such as waiting costs at different locations, we utilize space-time scheme (Hägerstrand, 1970; Miller, 1991, Ziliaskopoulos and Mahmassani, 1993) to formulate the PDPTW on SST transportation networks. The constructed networks are able to conveniently represent the complex pickup and delivery time windows without adding the extra constraints typically needed for the classical PDPTW formulation (e.g. Cordeau, 2006). The introduced SST networks also enable us to embed computationally efficient DP algorithms for solving the PDPTW without relying on off-the-shelf optimization solvers. Even though the solution space created by our formulation has multiple dimensions and accordingly large in its sizes, the readily available large amount of computer memory in modern workstations can easily accommodate the multi-dimensional solution vectors utilized in our application. Our fully customized solution algorithms, implemented in an advanced programming language such as C++, hold the promise of tackling large-sized regional transportation network instances. To address the multi-vehicle assignment requirement, we relax the transportation request satisfaction constraints into the objective function and utilize the related LR solution framework to decompose the primal problem to a sequence of time-dependent least-cost-path sub-problems.

In our proposed solution approach, we aim to incorporate several lines of pioneering efforts in different directions. Specifically, we (1) reformulate the VRPPDTW as a time-discretized, multi-dimensional, multi-commodity flow model with linear objective function and constraints, (2) extend the static DP formulation to a fully time-dependent DP framework for single-vehicle VRPPDTW problems, and (3) develop a LR solution procedure to decompose the multi-vehicle scheduling problem to a sequence of single-vehicle problems and further nicely integrate the demand satisfaction multipliers within the proposed SST network.

Based on the LR solution framework, we further present a holistic optimization approach for matching passengers' requests to transportation service providers, synchronizing transportation vehicle routing, and determining request pricing (e.g. through Lagrangian multipliers) for balancing transportation demand satisfaction and resource needs on urban networks.

3.3. Problem Statement Based on SST Network Representation

In this section, we first introduce our new mathematical model for the PDPTW. This is followed by a comprehensive comparison between our proposed model and the three-index formulation of Cordeau (2006) for the PDPTW, presented in Appendix A, for the demand node-oriented network.

3.3.1. Description of the PDPTW in SST Networks

We formulate the PDPTW on a transportation network, represented by a directed graph and denoted as $G(N, A)$, where N is the set of nodes (e.g. intersections or freeway merge points) and A is the set of links with different link types such as freeway segments, arterial streets, and ramps. As shown in Table 3.1, each directed link (i, j) has time-dependent travel time $TT(i, j, t)$ from node *i* to node *j* starting at time *t*. Every passenger p has a preferred time window for departure from his origin, $[a_p, b_p]$, and a desired time window for arrival at his destination, $[a'_p, b'_p]$, where a_p, b_p, a'_p, and b'_p are passenger p's earliest preferred departure time from his origin, latest preferred departure time from his origin, earliest preferred arrival time at his destination, and latest preferred arrival time at his destination, respectively. Each vehicle v also has the earliest departure time from its starting depot, e_v, and the latest arrival time at its ending depot, l_v. In the PDPTW, passengers may share their trip with each other; in other words, every vehicle *v*, considering its capacity Cap_v and the total routing cost, may serve as many passengers as possible provided that passenger *p* is picked up and dropped-off in his preferred time windows, $[a_p, b_p]$ and $[a'_p, b'_p]$, respectively.

Each transportation node has the potential to be the spot for picking up or dropping off a passenger. Likewise, a vehicle's depot might be located at any node in the transportation network. To distinguish regular transportation nodes from passengers' and vehicles' origin and destination, we add a single dummy node o_v' for vehicle v's origin depot and a single dummy node d_v' for vehicle v's destination depot. Similarly, we can also add dummy nodes o_p and d_p for passenger p. Each added dummy node is only connected to its corresponding physical transportation node by a link. The travel time on this link can be interpreted as the service time if the added dummy node is related to a passenger's origin or destination, and as preparation time if it is related to a vehicle's starting or ending depot. Table 3.1 lists the notations for the key sets, indices and parameters in the PDPTW.

Table 3.1
Sets, indices and parameters in the PDPTW.

Symbol	Definition
$\boldsymbol{V}$	Set of physical vehicles
$\boldsymbol{V}^*$	Set of virtual vehicles
$\boldsymbol{P}$	Set of passengers
$\boldsymbol{N}$	Set of physical transportation nodes in the physical traffic network based on geographical location
$\boldsymbol{W}$	Set of possible passenger carrying states
$\boldsymbol{v}$	Vehicle index
$\boldsymbol{v}_{\boldsymbol{p}}^*$	Index of virtual vehicle exclusively dedicated for passenger p
$\boldsymbol{p}$	Passenger index
$\boldsymbol{w}$	Passenger carrying state index
$(\boldsymbol{i}, \boldsymbol{j})$	Index of physical link between adjacent nodes i and j
$\boldsymbol{TT}(\boldsymbol{i}, \boldsymbol{j}, \boldsymbol{t})$	Link travel time from node i to node j starting at time t
$\boldsymbol{Cap}_v$	Maximum capacity of vehicle v
$\boldsymbol{a}_{\boldsymbol{p}}$	Earliest departure time from passenger p's origin
$\boldsymbol{b}_{\boldsymbol{p}}$	Latest departure time from passenger p's origin
$\boldsymbol{a}_{\boldsymbol{p}}'$	Earliest arrival time at passenger p's destination
$\boldsymbol{b}_{\boldsymbol{p}}'$	Latest arrival time at passenger p's destination
$[\boldsymbol{a}_{\boldsymbol{p}}, \boldsymbol{b}_{\boldsymbol{p}}]$	Departure time window for passenger p's origin
$[\boldsymbol{a}_{\boldsymbol{p}}', \boldsymbol{b}_{\boldsymbol{p}}']$	Arrival time window for passenger p's destination
$\boldsymbol{o}_v'$	Dummy node for vehicle v's origin
$\boldsymbol{d}_v'$	Dummy node for vehicle v's destination

$\boldsymbol{e_v}$	Vehicle v's earliest departure time from the origin depot
$\boldsymbol{l_v}$	Vehicle v's latest arrival time to the destination depot
$\boldsymbol{o_p}$	Dummy node for passenger p's origin (pickup node for passenger p)
$\boldsymbol{d_p}$	Dummy node for passenger p's destination (delivery node for passenger p)

We now use an illustrative example to demonstrate key modeling features of constructed networks. Consider a physical transportation network consisting of six nodes presented in Fig. 3.1. Each link in this network is associated with time-dependent travel time $TT(i,j,t)$. Without loss of generality, the number written on each link denotes the time-invariant travel time $TT(i,j)$ in terms of minutes. Suppose two requests with two OD pairs should be served. For simplicity, it is assumed that both passengers have the same origin (node 2) and the same drop-off node (node 3). There is only one vehicle available for serving. Moreover, it is assumed that the vehicle starts its route from node 4 and ends it at node 1. Passenger 1 should be picked up from dummy node o_1 in time window [4,7] and dropped off at dummy node d_1 in time window [11,14], while Passenger 2 should be picked up from dummy node o_2 in time window [8,10] and dropped off at dummy node d_2 in time window [13,16]. Vehicle 1 also has the earliest departure time from its starting depot, $t = 1$, and the latest arrival time at its ending depot, $t = 20$.

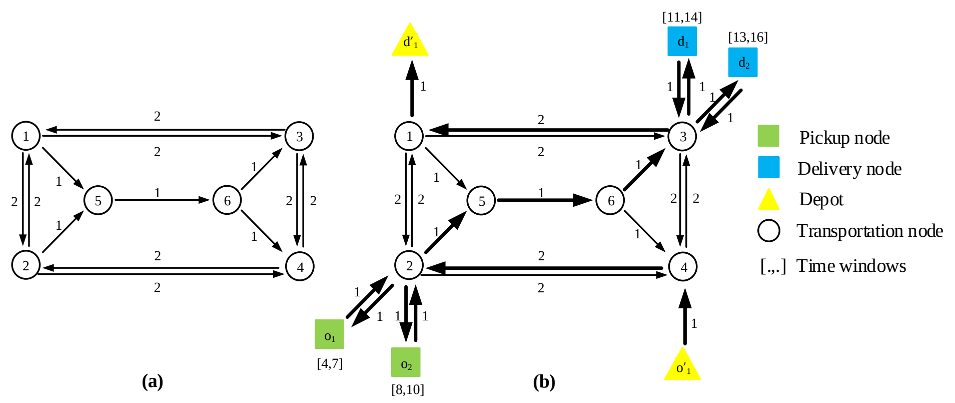
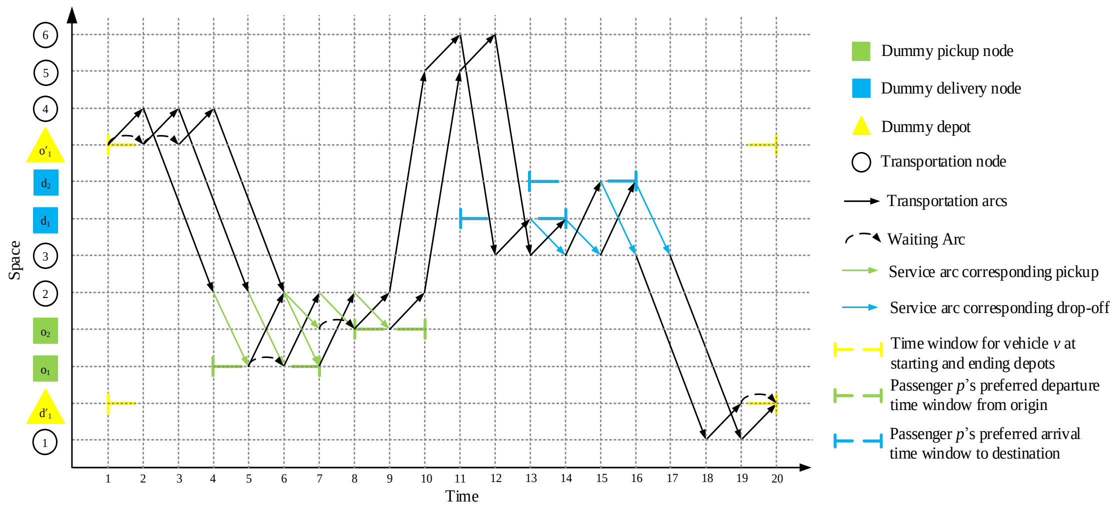


Fig. 3.1. (a) Six-node transportation network; (b) transportation network with the corresponding dummy nodes.

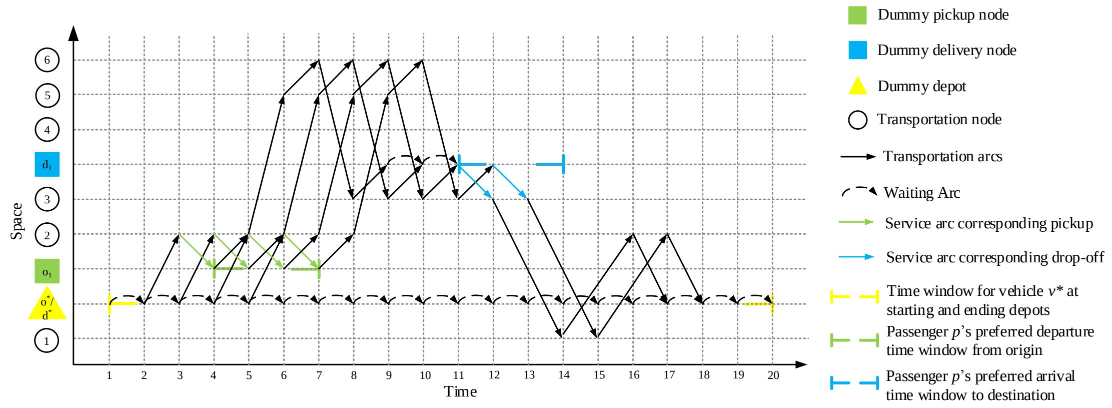
Note that the shortest path with node sequence $(o_1', 4, 2, o_1, 2, o_2, 2, 5, 6, 3, d_1, 3, d_2, 3, 1, d_1')$ from vehicle 1's origin to its ending depot is shown by bold arrows when it serves both passenger 1 and 2. To construct a SST network, the time horizon is discretized into a series of time intervals with the same time length. Without loss of generality, we assume that a unit of time has 1 min length. Interested readers are referred to Yang and Zhou (2014) on details about how to construct a space-time network. To avoid more complexity in the vehicle's space-time network illustrated in Fig. 3.2, only those arcs constituting the shortest paths from vehicle 1's origin to its destination are demonstrated. Our formulation has a set of precise rules to allow or restrict the vehicle waiting behavior in the constructed space-time network, depending on the type of nodes and the associated time window. First, vehicle v may wait at its own origin or destination depot or at any other physical transportation nodes. If a vehicle arrives at passenger p's origin node before time a_p, it must wait at the related physical node until the service is allowed to begin. Moreover, we assume that a vehicle is not allowed to stop at passenger p's dummy origin node after time b_p. Similarly, if a vehicle arrives at passenger p's destination node before time a_p', it must wait until it is allowed to drop-off passenger p, and vehicle v is not allowed to stop at passenger p's dummy destination node after time b_p'.



Space
Time
6
5
4
o'1
d2
d1
3
2
o2
o1
d'1
1
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20
Dummy pickup node
Dummy delivery node
Dummy depot
Transportation node
Transportation arcs
Waiting Arc
Service arc corresponding pickup
Service arc corresponding drop-off
Time window for vehicle v at starting and ending depots
Passenger p's preferred departure time window from origin
Passenger p's preferred arrival time window to destination

Fig. 3.2. Shortest paths with node sequence $(o_1', 4, 2, o_1, 2, o_2, 2, 5, 6, 3, d_1, 3, d_2, 3, 1, d_1')$ in vehicle 1's space-time network.

In the problem under consideration, we assume all passengers' desired departure and arrival time windows are feasible. However, it is quite possible that some passenger transporting requests could not be satisfied at all since the total number of physically available vehicles in the ride-sharing company or organization is not enough to satisfy all the demands. To avoid infeasibility for the constructed optimization problem, we define a virtual vehicle for each passenger exclusively. We assume that both starting and ending depots of virtual vehicle v_p^* are located exactly where passenger p is going to be picked up. By doing so, there is no cost incurred if the virtual vehicle is not needed to carry the related passenger, and in this case the virtual vehicle simply waits at its own depot. On the other hand, if the virtual vehicle is needed to perform the service to ensure there is a feasible solution, then virtual vehicle v_p^* starts its route from its starting depot, picks up passenger p, delivers him to his destination, and then comes back to its ending depot. Fig. 3.3 shows the shortest paths with node sequence $(o'_{1^*}, 2, o_1, 2, 5, 6, 3, d_1, 3, 1, 2, d'_{1^*})$ in vehicle v_1^*'s space-time network.



Space
Time
6
5
4
d1
3
2
o1
o*/d*
1
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20
Dummy pickup node
Dummy delivery node
Dummy depot
Transportation node
Transportation arcs
Waiting Arc
Service arc corresponding pickup
Service arc corresponding drop-off
Time window for vehicle v* at starting and ending depots
Passenger p's preferred departure time window from origin
Passenger p's preferred arrival time window to destination

Fig. 3.3. Shortest paths with node sequence $(o'_{1^*}, 2, o_1, 2, 5, 6, 3, d_1, 3, 1, 2, d'_{1^*})$ in vehicle v_1^*'s space-time network.

3.3.2. Representing the State of System and Calculating the Number of States

In the context of DP, we need to decompose the complex VRP structure into a sequence of overlapping stage-by-stage sub-problems in a recursive manner. For each stage of the optimization problem, we need to define the state of the process so that the state of the system with n stages to go can fully summarize all relevant information of the system for future decision-making purposes no matter how the process has reached the current stage n. In our pickup and delivery problem, in each vehicle's network, the given time index t acts as the stage, and the state of the system is jointly defined by two indexes: node index i and the passenger carrying state index w. The latter passenger carrying state w can be also represented as a vector with $|P|$ number of elements

$[\pi_1, \pi_2, \dots, \pi_p, \dots, \pi_P]$, where π_p equals 1 or 0 and denotes the status of passenger p whether he is riding the vehicle or not. To facilitate the descriptions of the state transition, we introduce the following equivalent notation system for passenger carrying states: if a vehicle carries passenger p, the pth element of the state w is filled with passenger p's id; otherwise, it is filled with a dash sign, as illustrated in Table 3.2.

Table 3.2
Binary representation and equivalent character-based representation for passenger carrying states.

Binary representation	Equivalent character-based representation
$[\mathbf{0}, \mathbf{0}, \mathbf{0}]$	$[\_\ \_\ \_]$
$[\mathbf{1}, \mathbf{0}, \mathbf{0}]$	$[p_1\ \_\ \_]$
$[\mathbf{0}, \mathbf{1}, \mathbf{1}]$	$[\_\ p_2\ p_3]$

Without loss of generality, for a typical off-line vehicle routing problem, the initial and ending states of the vehicles are assumed to be empty, corresponding to the state $[\_\ \_\ \_]$. For an on-line dynamic vehicle dispatching application, one can define the starting passenger carrying state to indicate the existing passengers riding the vehicle, for example, $[p_1\ \_\ \_]$ if passenger 1 is being served currently. We use an illustrative example to demonstrate the concept of a passenger's carrying state clearly. Suppose three requests with three different OD pairs should be served. There is only one vehicle available for serving and let's assume that the vehicle can carry up to two passengers at the same time. We can enumerate all different carrying states for the vehicle. The first state is the state in which the vehicle does not carry any passenger$[_\ _\ _]$. There are C_1^3 number of possible carrying states in which the vehicle only carries one passenger at time t:$[p_1\ \_\ \_]$, $[\_\ p_2\ \_]$, and $[\_\ \_\ p_3]$. Similarly, there are C_2^3 number of possible carrying states in which the vehicle carries two passengers at time t which are $[p_1\ p_2\ \_]$, $[p_1\ \_\ p_3]$, and $[\_\ p_2\ p_3]$. Since the vehicle can carry up to two passengers at the same time, the state of $[p_1\ p_2\ p_3]$ is infeasible. Fig. 3.4(a) and Fig. 3.4(b) show shared ride state $[p_1\ p_2\ \_]$ and single-passenger-serving state $[\_\ p_2\ \_]$.

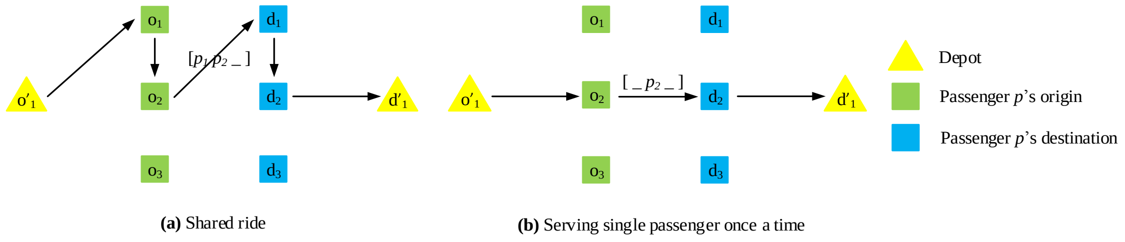


Fig. 3.4. State transition path (a) Passenger carrying state $[p_1\ p_2\ \_]$; (b) Passenger carrying state $[\_\ p_2\ \_]$.

We are further interested in the number of feasible states, which critically determines the computational efforts of the DP-based solution algorithm. First, there is a unique state in which vehicle v does not carry any passenger, which is a combinatory of C_0^P for selecting 0 passengers from the collection of P passengers. Similarly, there are C_1^P number of possible carrying states in which vehicle v only carries one passenger at a time. Likewise, there are C_k^P number of possible carrying states in which vehicle v carries k passengers at a time. Note that $k \leq Cap_v$. Therefore, the total number of possible passenger carrying states is equal to $\sum_{k=0}^{Cap_v} C_k^P$. It should be remarked that, according to the earliest departure time from the origin and the latest arrival time to the destination of different passengers, some of the possible carrying states, say $[\_\ p_2\ p_3]$, might be infeasible as there is insufficient transportation time to pick up those two passengers together while satisfying their time window constraints.

Consider the following example, where passenger 1 should be picked up in time window [4,7] and delivered in time window [9,12], whereas passenger 3's preferred time windows for being picked up and delivered are [20,24] and [25,29], respectively. So, it is obvious that passenger 1 and 3 cannot share their ride with each other and be transported at the same time by the same vehicle. Therefore, state $[p_1 \_ p_3]$ is definitely infeasible in this example. We will further explain how to reduce the search region by defining some rational rules and simple heuristics in section 3.5.3.

3.3.3. State Transition Associated with Pickup and Delivery Links

Each vehicle starts its trip from the empty state in which the vehicle does not carry any passengers. We call this state as the initial state (w_0). Each vertex in the constructed SST network is recognized by a triplet of three different indexes: node index i, time interval index t, and passenger carrying state index w. In the space-time transportation network construct, we can identify a traveling arc (i, j, t, s) starting from node *i* at time *t* arriving at node *j* at time *s*. Accordingly, in the SST network, each vertex (i, t, w) is connected to vertex (j, s, w') through arc (i, j, t, s, w, w'). To find all feasible combinations of passenger carrying state transition (w, w') on an arc, it is sufficient to follow these rules:

Rule 1. On a pickup link (with the passenger origin dummy node as the downstream node), vehicle v picks up passenger p, so π_p is changed from 0 to 1, or equivalently, the pth element of the corresponding states should be changed from a dash sign to passenger p id.

Rule 2. On a drop-off link (with the passenger destination dummy node as the upstream node), vehicle v drops off passenger p, so π_p is changed from 1 to 0, and the

p^{th} element of the corresponding states should be changed from passenger p id to a dash sign.

Rule 3. On a transportation link or links connected to vehicle dummy nodes, vehicle v neither picks up nor drops off any passenger, and all elements of w and w' should be the same.

To find all feasible passengers state transition (w, w'), we need to examine all possible combinations of w and w'. Consider a three-passenger case, in which Table 3.3 identifies all possible combinations of these state transitions. Note that the vehicle can carry up to two passengers at the same time in this example. The empty cells indicate impossible state transitions in the constructed space-time network with dedicated dummy nodes. The corresponding possible passenger carrying state transitions (pickup or drop-off) are illustrated in one graph in Fig. 3.5. Fig. 3.6 represents the projection on state-space network for the example presented in section 3.3.1.

Table 3.3
All possible combinations of passenger carrying states.

$\boldsymbol{w}$ \ $\boldsymbol{w'}$	$[\_\,\_\,\_]$	$[\boldsymbol{p_1}\,\_\,\_]$	$[\_\,\boldsymbol{p_2}\,\_]$	$[\_\,\_\,\boldsymbol{p_3}]$	$[\boldsymbol{p_1}\,\boldsymbol{p_2}\,\_]$	$[\boldsymbol{p_1}\,\_\,\boldsymbol{p_3}]$	$[\_\,\boldsymbol{p_2}\,\boldsymbol{p_3}]$
$[\_\,\_\,\_]$	no change	pickup	Pickup	pickup			
$[\boldsymbol{p_1}\,\_\,\_]$	drop-off	no change			Pickup	pickup	
$[\_\,\boldsymbol{p_2}\,\_]$	drop-off		no change		Pickup		pickup
$[\_\,\_\,\boldsymbol{p_3}]$	drop-off			no change		pickup	pickup
$[\boldsymbol{p_1}\,\boldsymbol{p_2}\,\_]$		drop-off	drop-off		no change		
$[\boldsymbol{p_1}\,\_\,\boldsymbol{p_3}]$		drop-off		drop-off		no change	
$[\_\,\boldsymbol{p_2}\,\boldsymbol{p_3}]$			drop-off	drop-off			no change

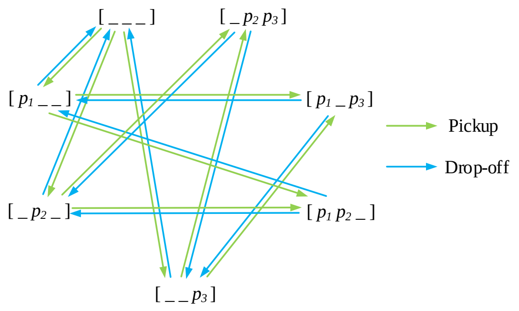


Fig. 3.5. Finite states graph showing all possible passenger carrying state transition (pickup or drop-off).

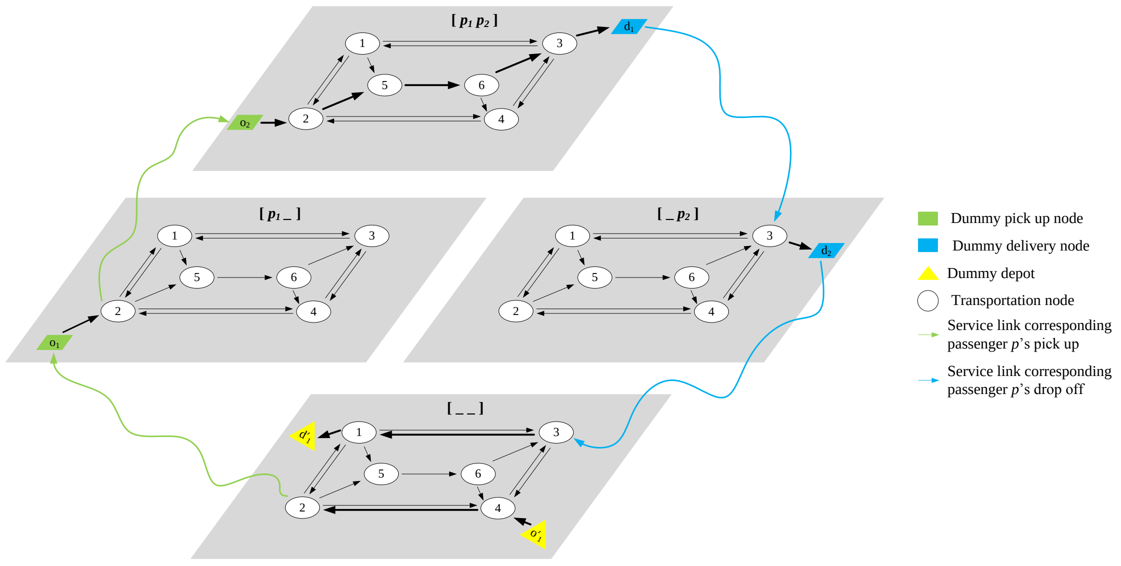


Fig. 3.6. Projection on state-space network representation for ride-sharing path (pick up passenger p_1 and then p_2).

3.4. Time-discretized Multi-commodity Network Flow Programming Model

Based on the constructed SST networks that can capture essential pickup and delivery time window constraints, we now start constructing a multi-commodity network flow programing model for the VRPPDTW. In this multi-dimensional network, the challenge is to systematically describe the related flow balance constraints for vehicles and request satisfaction constraints for passengers. As shown in Table 3.4, we use (i, t, w) to represent the indices of SST vertexes, and the corresponding arc index which is

(i, j, t, s, w, w'). Let B_v denote the set of SST arcs in vehicle v's network, which has three different types of arcs, namely, service arcs, transportation arcs and waiting arcs.

i. All passenger carrying state transitions (i.e., pickup or drop-off) occurs only on service arcs. In other words, all incoming arcs to passengers' origin nodes (pickup arcs shown by green lines in Fig. 3.5 and Fig. 3.6) and all outgoing arcs from their destination nodes (drop-off arcs shown by blue lines in Fig. 3.5 and Fig. 3.6) are considered service arcs.

ii. A link with both ends as physical nodes or vehicle dummy nodes are considered transportation arcs.

iii. Vehicles (both physical and virtual) may wait at their own origin or destination depot or at any other physical transportation nodes through waiting arcs $(i, i, t, t+1, w, w)$ from time *t* to time $t+1$ with the same passenger carrying state w.

Table 3.4
Indexes and variables used in the time-discretized network flow model.

Symbol	Definition
$(\boldsymbol{i}, \boldsymbol{t}, \boldsymbol{w})$, $(\boldsymbol{j}, \boldsymbol{s}, \boldsymbol{w}')$	Indexes of SST vertexes
$(\boldsymbol{i}, \boldsymbol{j}, \boldsymbol{t}, \boldsymbol{s}, \boldsymbol{w}, \boldsymbol{w}')$	Index of a space-time-state arc indicating that one can travel from node i at time t with passenger carrying state w to the node j at time s with passenger carrying state w'
$\boldsymbol{B_v}$	Set of SST arcs in vehicle v's network
$\boldsymbol{c}(\boldsymbol{v}, \boldsymbol{i}, \boldsymbol{j}, \boldsymbol{t}, \boldsymbol{s}, \boldsymbol{w}, \boldsymbol{w}')$	Routing cost of arc (i, j, t, s, w, w') traveled by vehicle v
$\boldsymbol{TT}(\boldsymbol{v}, \boldsymbol{i}, \boldsymbol{j}, \boldsymbol{t}, \boldsymbol{s}, \boldsymbol{w}, \boldsymbol{w}')$	Travel time of arc (i, j, t, s, w, w') traveled by vehicle v
$\boldsymbol{\Psi_{p,v}}$	Set of pickup service arcs of passenger p in vehicle v's networks
$\boldsymbol{y}(\boldsymbol{v}, \boldsymbol{i}, \boldsymbol{j}, \boldsymbol{t}, \boldsymbol{s}, \boldsymbol{w}, \boldsymbol{w}')$	$= 1$ if arc (i, j, t, s, w, w') is used by vehicle v; $= 0$ otherwise

In general, the travel time $TT(v, i, j, t, s, w, w')$ is the travel time of traversing from node i at time t with passenger carrying state w to node j at time s with passenger carrying state w' by vehicle v. Travel time for service arcs can be interpreted as the service time needed to pick up or drop-off a passenger, and as the preparation time if the arc is related to a vehicle's starting or ending depot. In addition, the travel time of the waiting arcs is assumed to be a unit of time.

The routing cost $c(v, i, j, t, s, w, w')$ for an arc can be defined as follows. The routing cost of a transportation arc is defined as a ratio of its travel time. For the physical vehicle, this ratio is basically the total transportation cost per hour when the vehicle is traveling, which may include the fuel, maintenance, depreciation, insurance costs, and more importantly, the cost of hiring a full-time or part-time driver. Let's assume that, in total, the transportation by a physical vehicle costs x dollars per hour. Since passengers should be served by physical vehicles by default and virtual vehicles serve passengers only if there is no available physical vehicle to satisfy their demand, we impose a quite expensive transportation cost per hour for virtual vehicles, let's say $2x$ dollars per hour. The routing cost of the service arcs are defined similarly to the routing cost of the transportation arcs. The routing cost of a waiting arc is also defined as a ratio of its waiting time. However, this ratio is basically the total cost of the physical vehicle v per hour when the driver has turned off the vehicle and is waiting at a node, which may only include the cost of hiring a full-time or part-time driver. Let's assume that, in total, waiting at a node by a physical vehicle costs y dollars per hour, with a typical relationship of waiting cost $<$ transportation cost per hour, i.e., $y < x$. We assume that waiting at the origin and destination depot for a physical vehicle has no charge for the service provider in order to encourage a vehicle to reduce the total transportation time, if possible. Moreover, for virtual vehicles, the waiting cost is always equal to zero to allow a virtual vehicle be totally idle at its own depot.

The model uses binary variables $y(v,i,j,t,s,w,w')$ equal to 1 if and only if SST arc (i,j,t,s,w,w') is used by vehicle v. Without loss of generality, we assume that a vehicle does not carry any passenger when it departs from its origin depot or arrives to its destination depot, which correspond to the passenger carrying state at node $(i=o_v',t=e_v)$ and $(j=d_v',s=l_v)$ as an empty state denoted by w_0. Note that, since passenger carrying state transitions only occur through service arcs, $w=w'=w_0$ for $y(v,o_v',j,e_v,s,w,w')$ and $y(v,i,d_v',t,l_v,w,w')$.

Note that each vehicle must end its route at the destination depot with the empty passenger carrying state. Therefore, if vehicle v picks up passenger p from his origin, to maintain the flow balance constraints, the vehicle must drop-off the passenger at his destination node so that the vehicle comes back to its ending depot with the empty passenger carrying state. As a result, constraints corresponding to the passengers' drop-off request is redundant and it does not need to enter into the following formulation. After constructing the SST transportation network for each vehicle, the PDPTW can be formulated as follows:

$$Min\ Z=\sum_{v\in(V\cup V^*)}\sum_{(i,j,t,s,w,w')\in B_v}c(v,i,j,t,s,w,w')y(v,i,j,t,s,w,w') \qquad (3.1)$$

s.t.

Flow balance constraints at vehicle v's origin vertex

$$\sum_{(o_v',j,e_v,s,w_0,w_0)\in B_v}y(v,o_v',j,e_v,s,w_0,w_0)=1 \qquad \forall v\in(V\cup V^*) \qquad (3.2)$$

Flow balance constraint at vehicle v's destination vertex

$$\sum_{(i,d_v',t,l_v,w_0,w_0)\in B_v}y(v,i,d_v',t,l_v,w_0,w_0)=1 \qquad \forall v\in(V\cup V^*) \qquad (3.3)$$

Flow balance constraint at intermediate vertex

$\sum_{(j,s,w'')} y(v,i,j,t,s,w,w'') - \sum_{(j',s',w')} y(v,j',i,s',t,w',w) = 0 \qquad (i,t,w) \notin \{(o'_v, e_v, w_0), (d'_v, l_v, w_0)\}, \forall v \in (V \cup V^*)$ (3.4)

Passenger p's pickup request constraint

$\sum_{v \in (V \cup V^*)} \sum_{(i,j,t,s,w,w') \in \Psi_{p,v}} y(v,i,j,t,s,w,w') = 1 \qquad \forall p \in P$ (3.5)

Binary definitional constraint

$y(v,i,j,t,s,w,w') \in \{0,1\} \qquad \forall (i,j,t,s,w,w') \in B_v, \forall v \in (V \cup V^*)$ (3.6)

The objective function (3.1) minimizes the total routing cost. Constraints (3.2) to (3.4) ensure flow balance on every vertex in vehicle v's SST transportation network. Constraints (3.5) express that each passenger is picked up exactly once by a vehicle (either physical or virtual). Constraint (3.6) defines that the decision variables are binary.

The three-index formulation of Cordeau (2006) for the PDPTW in the OD network is presented in Appendix A. Table 3.5 shows that our proposed model encompasses all constraints used in Cordeau's model.

Table 3.5
An analogy between Cordeau's model and our model for the PDPTW.

Cordeau (2006)	Our model
Three-index variables x_{ij}^{v} for vehicle v on link (i,j)	Seven-index variable $y(v,i,j,t,s,w,w')$ for vehicle v on arc (i,j,t,s,w,w').
(A.1) minimizes the total routing cost.	(3.1) minimizes the total routing cost.
(A.2) guarantees that each passenger is picked up.	(3.5) guarantees that each passenger is picked up.
(A.2) and (A.3) ensure that each passenger's origin and destination are visited exactly once by the same vehicle.	(3.2) to (3.5) ensure that the same vehicle v transports passenger p from his origin to his destination.
(A.4) expresses that each vehicle starts its route from the origin depot.	(3.2) expresses that each vehicle starts its route from the origin depot.
(A.5) ensures the flow balance on each node.	(3.2) to (3.4) ensure flow balance on every vertex in vehicle v's network.
(A.6) expresses that each vehicle ends its route to the destination depot.	(3.3) expresses that each vehicle ends its route to the destination depot.

(A.7) ensures the validity of the time variables.	The essence of SST networks ensures the time variables are calculated correctly through arc (i, j, t, s, w, w'), where arrival time $s = t + TT(i, j, t)$.
(A.8) ensures the validity of the load variables.	The structure of SST networks ensures that each vehicle transports a number of passengers up to its capacity at a time, in terms of feasible states (w, w').
(A.9) defines each passenger's ride time.	Employing SST networks defines each passenger's ride time.
(A.10) imposes the maximal duration of each route.	Vehicle v's network is constructed subject to time window $[e_v, l_v]$.
(A.11) imposes time windows constraints.	Passenger p's network is constructed subject to time window $[a_p, b'_p]$.
(A.12) imposes ride time of each passenger constraints.	Passenger p's network is constructed subject to time window $[a_p, b'_p]$.
(A.13) imposes capacity constraints.	The structure of SST networks ensures that each vehicle transports a number of passengers up to its capacity at a time.
(A.14) defines that the decision variables are binary.	(3.6) defines binary decision variables.

3.5. LR-based Solution Approach

Defining multi-dimensional decision variables $y(v, i, j, t, s, w, w')$ leads to computational challenges for the large-scale real-world data sets, which should be addressed properly by specialized programs and an innovative solution framework. We reformulate the problem by relaxing the complicating constraints (3.5) into the objective function and introducing Lagrangian multipliers, $\lambda(p)$, to construct the dualized Lagrangian function (3.7).

$$L = \sum_{v\in(V\cup V^*)} \sum_{(i,j,t,s,w,w')\in B_v} c(v, i, j, t, s, w, w') y(v, i, j, t, s, w, w') +$$

$$\sum_{p\in P} \lambda(p) \left[\sum_{v\in(V\cup V^*)} \sum_{(i,j,t,s,w,w')\in \Psi_{p,v}} y(v, i, j, t, s, w, w') - 1\right] \qquad (3.7)$$

Therefore, the new relaxed problem can be written as follows:

$$Min\ L \qquad (3.8)$$

s.t.

$\sum_{(o'_v,j,e_v,s,w_0,w_0)\in B_v} y(v,o'_v,j,e_v,s,w_0,w_0)=1 \qquad \forall v\in(V\cup V^*)$ (3.9)

$\sum_{(i,d'_v,t,l_v,w_0,w_0)\in B_v} y(v,i,d'_v,t,l_v,w_0,w_0)=1 \qquad \forall v\in(V\cup V^*)$ (3.10)

$\sum_{(j,s,w'')} y(v,i,j,t,s,w,w'') - \sum_{(j',s',w')} y(v,j',i,s',t,w',w)=0 \ \ (i,t,w)\notin$

$\{(o'_v,e_v,w_0),(d'_v,l_v,w_0)\},\forall v\in(V\cup V^*)$ (3.11)

$y(v,i,j,t,s,w,w')\in\{0,1\} \qquad \forall(i,j,t,s,w,w')\in B_v,\forall v\in(V\cup V^*)$ (3.12)

If we further simplify function L, the problem will become a time-dependent least-cost path problem in the constructed SST network. The simplified Lagrangian function L can be written in the following form:

$L=\ \sum_{v\in(V\cup V^*)}\sum_{(i,j,t,s,w,w')\in B_v}\xi(v,i,j,t,s,w,w')y(v,i,j,t,s,w,w')-\sum_{p\in P}\lambda(p)$ (3.13)

Where the generalized arc cost $\xi(v,i,j,t,s,w,w')$ equals $c(v,i,j,t,s,w,w')+\lambda(p)$ for each arc $(i,j,t,s,w,w')\in\Psi_{p,v}$, and equals $c(v,i,j,t,s,w,w')$, otherwise.

3.5.1. Time-dependent Forward DP and Computational Complexity

Several efficient algorithms have been proposed to compute time-dependent shortest paths in a network with time-dependent arc costs (Ziliaskopoulos and Mahmassani, 1993; Chabini, 1998). In this section, we use a time-dependent DP algorithm to solve the least-cost path problem obtained in section 3.4. The structure of the SST network ensures that time always advances on the arcs of the networks. In this chapter, let us consider the unit of time as 1 min. Let $\mathcal{N}$ denote the set of nodes including both physical transportation and dummy nodes, $\mathcal{A}$ denote the set of links, $\mathcal{T}$ denote the set of time stamps covering all vehicles' time horizons, $\mathcal{W}$ denote the set of all feasible passenger carrying states, and

$L(i, t, w)$ denote the label of vertex (i, t, w) and term “pred” stands for the predecessor.

Algorithm 3.1 described below uses forward DP:

```
// Algorithm 3.1: Time-dependent forward DP algorithm
for each vehicle $v \in (V \cup V^*)$ do
begin
// initialization
    $L(.,.,.) := +\infty$;
    node pred of vertex $(.,.,.) := -1$;
    time pred of vertex $(.,.,.) := -1$;
    state pred of vertex $(.,.,.) := -1$;
    // vehicle $v$ starts its route from the empty state at its origin at the earliest departure time $L(o'_v, e_v, w_0) := 0$;
    for each time $t \in [e_v, l_v]$ do
    begin
                    for each link $(i, j)$ do
                    begin
                    for each state $w$ do
                    begin
                derive downstream state $w'$ based on the possible state transition on link $(i, j)$;
                derive arrival time $s = t + TT(i, j, t)$;
                if $(L(i, t, w) + \xi(v, i, j, t, s, w, w') < L(j, s, w'))$ then
                begin
                    $L(j, s, w') := L(i, t, w) + \xi(v, i, j, t, s, w, w')$ ; //label update
                    node pred of vertex $(j, s, w') := i$;
                    time pred of vertex $(j, s, w') := t$;
                    state pred of vertex $(j, s, w') := w$;
                end;
                end;
    end;
    end;
end;
```

Let’s define $|\mathcal{T}|$, $|\mathcal{A}|$, $|\mathcal{W}|$ as the number of time stamps, links, and passenger carrying states, respectively. Therefore, the worst-case complexity of the DP algorithm is $|\mathcal{V}||\mathcal{T}||\mathcal{A}||\mathcal{W}|$, which can be interpreted as the maximum number of steps to be performed in this algorithm in this four-loop structure, corresponding to the sequential loops over vehicle, time, link, and starting carrying state dimensions. It should be remarked that the ending state w' is uniquely determined by the starting state w and the related link (i, j) depending on its service type: pickup, delivery, or pure transportation.

In a transportation network, the size of links is much smaller than the counterpart in a complete graph, that is, $|\mathcal{A}| \ll |\mathcal{N}||\mathcal{N}|$; in fact, the typical out-degree of a node in transportation networks is about 2-4.

Table 3.6 shows detailed comparisons between the existing DP-based approach (Psaraftis, 1983 and Desrosiers et al. 1986) and our proposed method. We guarantee the completeness of state representation. The state representation of Psaraftis (1983), $(L, k_1, k_2, \dots, k_n)$, consists of L, the location currently being visited, and k_i, the status of passenger i. In this representation, $L = 0$, $L = i$, and $L = i + n$ denote starting location, passenger i's origin, and passenger i's destination, respectively. In addition, the status of passenger i is chosen from the set $\{1,2,3\}$, where 3 means passenger i is still waiting to be picked up, 2 means passenger i has been picked up but the service has not been completed, and 1 means passenger i has been successfully delivered. This cumulative passenger service state representation (in terms of $k_1, k_2, \dots, k_P$) requires a space complexity of $O(3^p)$, while our proposed (prevailing) passenger carrying state representation has a much smaller space requirement of $\sum_{k=0}^{Cap_v} C_k^P$ when the vehicle capacity is low (e.g. 2 or 3 for taxi). Desrosiers et al. (1986) use state representation (S, i), where S is the set of passengers' origin, $\{1, \dots, n\}$, and destination, $\{n + 1, \dots 2n\}$. State (S, i) is defined if and only if there exists a feasible path that passes through all nodes in S and ends at node i. In fact, our time-dependent state (w, i, t), which is jointly defined by three indexes: (i) the status of customers, (ii) the current node being visited, and (iii) the current time, is more focused on the time-dependent current state at exact time stamp t, while $(L, k_1, k_2, \dots, k_n)$ and (S, i) representations use a time-lagged time-period-based state representation to cover complete or mutually exclusive states from time 0 to time t.

Table 3.6
Comparison between existing DP based approach and the method proposed in this chapter.

Features	Existing DP based approach		DP proposed in this chapter
	Psaraftis (1983)	Desrosiers et al. (1986)	
Type of problem	Single vehicle, Many-to-many, Single depot	Single vehicle, Many-to-many, Single depot	Multiple vehicle, Many-to-many, Multiple depot
Network	Consists of passengers' origin and destination nodes and the vehicle depot	Consists of passengers' origin and destination nodes and the vehicle depot	Consists of transportation nodes, passengers' origin and destination, and vehicles' depots
Time-dependent link travel time	No	No	Yes
Objective function	Minimize route duration	Minimize total distance traveled	Minimize total routing cost consisting of transportation and waiting costs
State	State-space $(L, k_1, k_2, \dots, k_n)$	State-space (S, i)	State-space-time (w, i, t)
Stage	Node index	Node index	Time index
States reduction due to the vehicle capacity and time windows	Yes	Yes	Yes

We come back to the illustrative example presented in section 3.3.1. Let's assume the routing cost of a transportation or service arc traversed by a physical vehicle is \$22/h, while the routing cost of a transportation or service arc traversed by a virtual vehicle is \$50/h. Moreover, assume that the waiting cost of a physical vehicle is \$15/h, while the waiting cost of a virtual vehicle is assumed to be \$0/h. Table 3.7 shows how the label of each vertex is calculated by the DP solution algorithm presented above. Note that w_0, w_1, w_2, and w_3 are passenger carrying states $[\_\,\_]$, $[p_1\,\_]$, $[p_1\,p_2]$, and $[\_\,p_2]$, respectively. For instance, according to Fig. 3.1, traveling from node 4 to node 2 takes 2 min. Since the number written on each link denotes the time-invariant travel time $TT(i,j)$, we can conclude that travel time for link $(4,2)$ starting at time stamp $t = 2$ is also 2 min. To update the label corresponding to node 2, it is sufficient to calculate the routing cost of the stated arc in terms of dollars which can be obtained by $\frac{2}{60} \times 22(\frac{\$}{\text{h}}) = 0.73(\$)$ and add it to the current label of node 4 which is 0.37. Therefore, the updated label for node 2 will be 1.1. Similarly, we can calculate the routing cost of a waiting link (o_2, o_2) starting at time stamp $t = 7$ by $\frac{1}{60} \times 15\left(\frac{\$}{\text{h}}\right) = 0.25\ (\$)$.

Table 3.7
SST trajectory for ride-sharing service trip with node sequence $(o_1', 4, 2, o_1, 2, o_2, 2, 5, 6, 3, d_1, 3, d_2, 3, 1, d_1')$.

Time index	1	2	4	5	6	7	8	9	10	11	12	13	14	15	16	18	19	20
Node index	o_1'	4	2	o_1	2	o_2	o_2	2	5	6	3	d_1	3	d_2	3	1	d_1'	d_1'
State index	w_0	w_0	w_0	w_1	w_1	w_2	w_2	w_2	w_2	w_2	w_2	w_2	w_3	w_3	w_0	w_0	w_0	w_0
Cost	0.0	.37	.73	.37	.37	.37	.25	.37	.37	.37	.37	.37	.37	.37	.37	.73	.37	0.0
Cumulative cost	0.0	.37	1.1	1.47	1.84	2.21	2.46	2.83	3.2	3.57	3.94	4.31	4.68	5.05	5.42	6.15	6.52	6.52

3.5.2. LR-based Solution Procedure

In this section, we describe the LR solution approach implemented to solve the time-dependent least cost path problem presented in section 3.5. According to Eq. (3.13), $\xi(v,i,j,t,s,w,w')$ is only updated for $\forall(v,i,j,t,s,w,w') \in \Psi_{p,v}$. Table 3.8 lists the notations for the sets, indices and parameters required for the LR algorithm.

Table 3.8
Notations used in LR algorithm.

Symbol	Definition
$\boldsymbol{\lambda^k(p)}$	LR multiplier corresponding to the passenger p's pickup request constraint at iteration k
$\boldsymbol{\xi(v,i,j,t,s,w,w')}$	Modified routing cost of arc (i,j,t,s,w,w') after introducing Lagrangian multipliers
$\boldsymbol{k}$	Iteration number
$\boldsymbol{Y}$	Set of solution vectors $y(v,i,j,t,s,w,w')$
$\boldsymbol{LB^k}$	Global lower bound for the primal problem at iteration k
$\boldsymbol{UB^k}$	Global upper bound for the primal problem at iteration k
$\boldsymbol{Y_{LB}^k}$	Set of lower bound solution vectors Y at LR iteration k
$\boldsymbol{Y_{UB}^k}$	Set of upper bound solution vectors Y at LR iteration k
$\boldsymbol{\theta^k}$	Step size at iteration k

$\boldsymbol{LB^*}$	Best global lower bound for the primal problem
$\boldsymbol{UB^*}$	Best global upper bound for the primal problem
$\boldsymbol{Y^*}$	Best solution vectors derived from the best lower bound
$\boldsymbol{base\_profit}$	The amount of money (in terms of dollars) passenger p initially offers to be served

The optimal value of the Lagrangian dual problem provides a lower bound for the primal problem. To find the optimal solution for the Lagrangian dual problem, it is sufficient to compute time-dependent least cost SST path for each vehicle v based on updated arc cost ξ's by calling time-dependent forward DP algorithm mentioned before.

The optimal solution of the Lagrangian dual problem may or may not be feasible for the primal problem. If the optimal solution of the Lagrangian dual problem is feasible for the primal problem, we have definitely obtained the optimal solution of the primal problem. If not, we apply a heuristic to find an upper bound for the primal solution. In this heuristic, the physical vehicles initially leave their depots to serve unserved customers provided that the money obtained in return for services overweigh the cost of transportation. Finally, if there is any unserved customer remained in the system, in order to avoid infeasibility, the virtual vehicle corresponding to the unserved customer departs from its depot to serve the passenger. The LR algorithm can be described as follows:

```
// Algorithm 3.2: LR algorithm
// step 0. initialization
        –  set iteration $k = 0$;
        –  initialize $Y_{LB}^0$, $Y_{UB}^0$, $Y^*$, and $\lambda^0(p)$ to zero;
        –  initialize $\theta^0(p)$ to $base\_profit$;
        –  initialize $LB^*$ to $-\infty$; and $UB^*$ to $+\infty$;
        –  define a termination condition such as if $k$ becomes greater than a predetermined
           maximum iteration number, or if the relative gap percentage between $LB^*$ and $UB^*$ becomes
           less than a predefined gap (i.e. 5%);
while termination condition is false, for each LR iteration $k$ do
begin
        –  reset the visit count for each arc $(v,i,j,t,s,w,w') \in \Psi_{p,v}$ to zero; // $v \in (V \cup V^*)$
// step 1. generating $LB^k$
// step 1.1. least cost path calculation for each vehicle sub-problem
```

```
                    – initialize $LB^k$ to 0;
for each vehicle $v \in (V \cup V^*)$ do
begin
                // input: $\xi(v,i,j,t,s,w,w')$
                    – compute time-dependent least cost SST path for vehicle $v$ based on updated arc
                    cost $\xi$'s by calling Algorithm 1;
                    – update the visit count for each arc $(v,i,j,t,s,w,w') \in \Psi_{p,v}$;
                // output: $Y_{LB}^k$
    end;
// step 1.2. update $LB^*$
                    – update $LB^k$ by substituting solution vector $Y_{LB}^k$ in the objective function of the
                    dual problem (Eq. (13));
                    – update $LB^*$ by $max(LB^k, current\ LB^*)$ and $Y^*$ by its corresponding solution;
// step 1.3. sub-gradient calculation
                    – calculate the total number of visits of passenger $p$'s origin by expression (3.14);
                    $\sum_{v\in(V\cup V^*)}\sum_{(i,j,t,s,w,w')\in\Psi_{p,v}} y(v,i,j,t,s,w,w')$    (3.14)
                    – compute sub-gradients by Eq. (3.15);
                    $\nabla L_{\lambda^k(p)} = \sum_{v\in(V\cup V^*)}\sum_{(v,i,j,t,s,w,w')\in\Psi_{p,v}} y(v,i,j,t,s,w,w') - 1$ for$\forall p$    (3.15)
                    – update arc multipliers by Eq. (3.16);
                    $\lambda^{k+1}(p) = \lambda^k(p) + \theta^k(p)\nabla L_{\lambda^k(p)}$ for $\forall p$    (3.16)
                    – update arc cost $\xi(v,i,j,t,s,w,w')$ for each arc $(v,i,j,t,s,w,w') \in \Psi_{p,v}$ by Eq.
                    (3.17);
                    $\xi(v,i,j,t,s,w,w') = c(v,i,j,t,s,w,w') + \lambda^{k+1}(p)$    (3.17)
                    – update step size by Eq. (3.18);
                    $\theta^{k+1}(p) = \frac{\theta^0(p)}{k+1}$    (3.18)
    // Step 2. generating $UB^k$
        // step 2.1. finding a feasible solution for the primal problem
                    – set $UB^k = 0$;
                    – adopt the passenger-to-vehicle assignment matrix from the lower bound solution
                    in step 1.2;
                for each vehicle $v \in (V \cup V^*)$ do
                begin
                    // if passenger $p$ is served by multiple vehicles, then designate one of the vehicles
                    (e.g. first in the set) to serve this passenger, which means that the other vehicles
                    should not serve this passenger in the upper bound generation stage.
                    if (passenger $p$ is assigned to physical vehicle $v$) do
                    begin
                        if passenger $p$ has not been already served by any other vehicle
                            set arc cost on the pickup arc for passenger $p$ temporarily to $-M$;
                            // $M$ is chosen a big positive number in order to attract vehicle $v$ for serving
                            passenger $p$
                               else
                            set arc cost on the pickup arc for passenger $p$ temporarily to $+M$;
                            // $M$ is chosen a big positive number in order to guarantee vehicle $v$ does
                            not serve passenger $p$
                               end;
                    // if passenger $p$ is not served by any physical vehicle, then designate the
                    corresponding virtual vehicle to serve this passenger.
                    if (passenger $p$ is not served by any physical vehicle & vehicle $v$ is the
                    corresponding virtual vehicle for passenger $p$)
                        set arc cost on the pickup arc for passenger $p$ temporarily to $-M$;
                        // $M$ is chosen a big positive number in order to attract vehicle $v$ for serving
                        passenger $p$
                        –      compute time-dependent least cost path for vehicle $v$ by calling
                        Algorithm 1;
                        –      compute the actual transportation costs (denoted as $TC_v$) along the path
                        solution for vehicle $v$
                        –      update upper bound objective function as $UB^k = UB^k + TC_v$.
                    end;
                // The result of this passenger-to-vehicle assignment updating is that each passenger is
                served by exactly one vehicle (either physical or virtual).
        // step 2.2. update $UB^k$
                    –  update $UB^k$ by substituting solution vector $Y_{UB}^k$ in the objective function of the
                    primal problem;
        // step 2.3. update $UB^*$
                    –  $UB^* = min(UB^k, current\ UB^*)$;
                    –  find the relative gap percentage between $LB^*$ and $UB^*$ by $\frac{UB^*-LB^*}{UB^*} \times 100$;
                    –  $k = k + 1$;
        end;
```

We would like to make remarks in following two cases:

(i) In the upper bound solution, all passengers are only served by the physical vehicles. In this case, we can be sure that the total number of physical vehicles has been sufficient to serve all requests. Accordingly, the service prices in the corresponding lower bound solution typically have been set such that the money obtained in return for services overweighs the cost of transportation so that physical vehicles are dispatched to serve customers.

(ii) In the final optimal solution, there might be some passengers who are served by virtual vehicles. Obviously, serving a passenger by a virtual vehicle is expensive due to its transportation cost. In addition, when the virtual vehicle drops off the passenger, it should perform a deadheading trip with significantly high cost from the passenger's destination to its depot (the passenger’s origin).

3.5.3. Search Region Reduction

In this section, we describe how to reduce the search region by the aid of some simple heuristics in which some rational rules are applied. Let EDT, LDT, EAT, and LAT denote the earliest departure time from origin, latest departure time from origin, earliest arrival time to destination, and latest arrival time to destination, respectively. In addition, let $TTSP_{x \to y}$ denote the travel time corresponding to the shortest path from node x to node y.

Rule 1. No overlapping time windows: The first rational rule is that if $LAT(p_1) < EDT(p_2)$, then passenger p_1 and p_2's ride-sharing is impossible. Fig. 3.7 illustrates an example of two passengers whose ride-sharing is impossible due to no overlapping time windows.

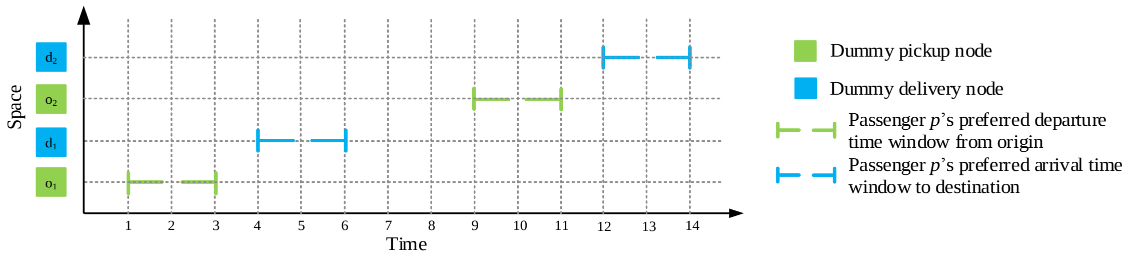

Space
d₂
o₂
d₁
o₁
Time
1
2
3
4
5
6
7
8
9
10
11
12
13
14
Dummy pickup node
Dummy delivery node
Passenger p's preferred departure time window from origin
Passenger p's preferred arrival time window to destination

Fig. 3.7. Illustration of the first rational rule for search region reduction.

Rule 2. Travel time is insufficient: The second rational rule can be stated as follows: if $\left\{LDT(p_2) - EDT(p_1) < TTSP_{o_{p_1} \to o_{p_2}} \ \& \ LDT(p_1) - EDT(p_2) < TTSP_{o_{p_2} \to o_{p_1}}\right\}$, then passenger p_1 and p_2 cannot share their ride with each other. It means that if the maximum time a vehicle can have to go from passenger p_1's origin to p_2's origin, $LDT(p_2) - EDT(p_1)$, is less than the total travel time corresponding to the shortest path from o_{p_1} to o_{p_2}, and also if the maximum time a vehicle can have to go from passenger p_2's origin to

p_1's origin, $LDT(p_1) - EDT(p_2)$, is less than the total travel time corresponding to the shortest path from o_{p_2} to o_{p_1}, then passenger p_1 and p_2's ride-sharing is impossible. Similarly, if $\{LAT(p_2) - EAT(p_1) < TTSP_{d_{p_1} \to d_{p_2}}$ & $LAT(p_1) - EAT(p_2) < TTSP_{d_{p_2} \to d_{p_1}}\}$, then passenger p_1 and p_2's ride-sharing is impossible. The total number of passenger carrying states is dramatically decreased via this rule. Fig. 3.8 illustrates the second rule by an example. Suppose two requests with two OD pairs should be served by a vehicle. Fig. 3.8(a) illustrates transportation network with the corresponding dummy nodes and time windows. According to the Fig. 3.8(a), $TTSP_{o_{p_1} \to o_{p_2}}$ and $TTSP_{o_{p_2} \to o_{p_1}}$ are 5 and 6, respectively. Since $\{(6-4) < 5$ & $(5-4) < 6\}$, then passenger p_1 and p_2's ride-sharing is impossible.

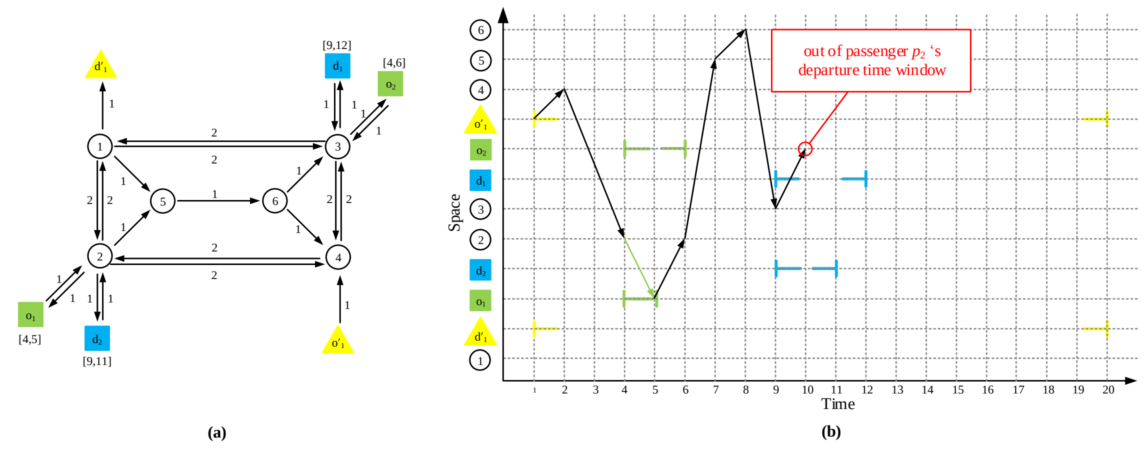


Fig. 3.8. Illustration of the second rational rule for search region reduction; (a) transportation network with the corresponding dummy nodes and time windows; (b) vehicle 1's space-time network.

Rule 3. A node is too far away from the vehicle starting or ending depot: The third rational rule is stated as follows: if $(TTSP_{o_v \to x} + TTSP_{x \to d_v}) > (LAT(v) - EDT(v))$, then vehicle v does not have enough time to visit node x in its time horizon; therefore, node x is not accessible for vehicle v and should not be considered in vehicle v's search region. Note that node x can be any physical or dummy node. Fig. 3.9 illustrates the third rule by an example. Suppose a passenger with an OD pair should be served by a vehicle. Fig. 3.9(a) illustrates transportation network with the corresponding dummy nodes and time windows. Fig. 3.9(b) shows that passenger p_1's origin, o_1, is not accessible for the vehicle. In addition to this rule, we can also say that a passenger is inaccessible for a vehicle if the time for a vehicle to pick up the passenger and visit his destination is longer than the vehicle's time window.

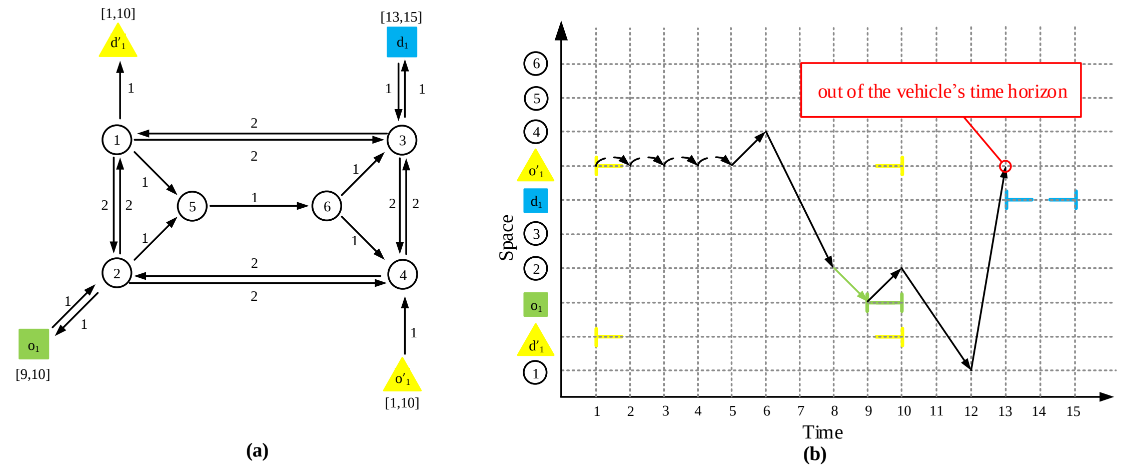


Fig. 3.9. Illustration of the third rational rule for search region reduction; (a) transportation network with the corresponding dummy nodes and time windows; (b) vehicle 1's space-time network.

The first three rules are hard rules at which we are able to eliminate some vertexes in the SST networks. The forth heuristic is the way of estimating the search region reduction ratio. Let path α be the longest possible path in vehicle v's SST networks with total travel time τ_α. Let m_p denote the middle point of passenger p's departure time window. Therefore, $m_p = \frac{EDT(p)+LDT(p)}{2}$. Let's assume that M, the middle point of a passenger's departure time window, is a random variable uniformly distributed in vehicle v's time horizon with $LAT(v) - EDT(v)$ length. It may be reasonable to assume that if $\left|m_{p_1} - m_{p_2}\right| > \tau_\alpha$, then passenger p_1 and p_2 cannot be in the same vehicle at a time. We use an example to show that this rule can reduce the search region considerably. Assume vehicle v's time window is [0, 240], and M is a random variable uniformly distributed in vehicle v's time horizon [0,240]. Let's assume $\tau_\alpha = 60$ min. The probability of having two passengers who share their ride with each other can be calculated by finding the $Prob\left(\left|m_{p_1} - m_{p_2}\right| \leq 60\ minutes\right)$, where m_{p_1} and m_{p_2} are randomly generated from [0, 240]. This probability equals to $\frac{7}{16} = 43.75\%$. This can be shown with the following derivation. The shaded area in in Fig. 3.10 shows $Prob\left(\left|m_{p_1} - m_{p_2}\right| \leq 60\right)$.

$$Prob\left(\left|m_{p_1} - m_{p_2}\right| \leq 60\right) = Prob\left(-60 \leq m_{p_1} - m_{p_2} \leq 60\right)$$

$$Prob\left(-60 \leq m_{p_1} - m_{p_2} \leq 60\right)$$

$$= 1 - \left[Prob\left(m_{p_1} - m_{p_2} < -60\right) + Prob\left(m_{p_1} - m_{p_2} > 60\right)\right]$$

$$= 1 - \left[\frac{\frac{180 \times 180}{2}}{240 \times 240} + \frac{\frac{180 \times 180}{2}}{240 \times 240}\right] = \frac{7}{16}$$

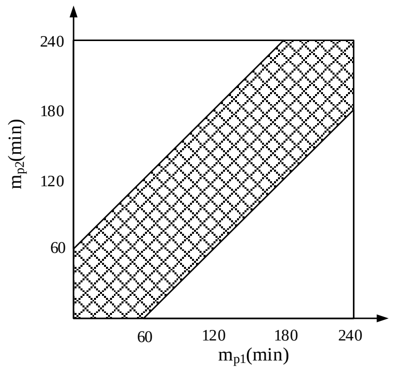


Fig. 3.10. The probability of having two passengers who share their ride with each other where m_{p_1} and m_{p_2} are uniformly distributed in [0, 240]. Note that $\tau_\alpha = 60$ min.

Therefore, by considering this practical rule in this example, we can reduce the total number of passenger carrying states in which two passengers share their ride with each other by more than half. By considering this rational rule, calculating the probability of having more than two passengers at the same time in vehicle v is more complicated, but at least we know that the probability of having k number of passengers ($k > 2$) who may share their ride with each other is certainly less than 43.75%.

3.6. Computational Results and Discussions

The algorithms described in this chapter were coded in C++ platforms. The experiments were performed on an Intel Workstation running two Xeon E5-2680 processors clocked at 2.80 GHz with 20 cores and 192 GB RAM running Windows Server 2008 x64 Edition. In addition, parallel computing and Open Multi-Processing (OpenMP) technique (Chandra et al., 2000) are implemented for generating lower bound and upper bound at each iteration in the LR algorithm. In this section, we initially examine our proposed model on a six-node transportation network followed by the medium-scale and large-scale transportation networks, Chicago and Phoenix, to demonstrate the computational efficiency and solution optimality of our developed algorithm. The scenarios and test cases are randomly generated in those transportation networks. Moreover, we test our algorithms on the modified version of instances proposed by Ropke and Cordeau (2009) which is publicly available at http://www.diku.dk/~sropke/.

3.6.1. Illustrative cases

According to Section 3.5.1, it is assumed that the routing cost of a transportation or service arc traversed by a physical vehicle is \$22/h, while the routing cost of a transportation or service arc traversed by a virtual vehicle is \$50/h. Moreover, the waiting cost of a physical vehicle either at a transportation or at a dummy node is \$15/h (waiting at dummy nodes corresponding starting and ending depots has \$0/h cost), while the waiting cost of a virtual vehicle at any node is assumed to be \$0/h. The value of $base\_profit$ is also assumed to be \$10 for all passengers. Initially, we test our algorithm on the six-node transportation network illustrated in Fig. 3.1(a) for six scenarios. Table 3.9 shows these scenarios with various number of passengers and vehicles, OD pairs, and passengers' departure and arrival time windows. Then, we will examine the results corresponding to each scenario individually. Terms "TW" and "TH" stands for time window and time horizon, respectively. Table 3.10 shows the results corresponding each scenario.

Scenario I. Two passengers are served by one vehicle, where passengers have different OD pairs with overlapping time windows. In this case, the vehicle serves both passengers in their preferred time windows through ride-sharing mode.

Scenario II. Two passengers with different OD pairs are served by one vehicle; however, unlike in scenario I, passengers could not share their ride with each other due to their time windows. In this case, the vehicle may wait at any node to finally serve both passengers.

Scenario III. Two passengers with different OD pairs and one vehicle are present in the system; however, due to the passengers' overlapping time windows, serving both passengers by one vehicle is impossible. Therefore, the driver would prefer to transport a passenger incurring the least cost. In this case, passenger p_1 is selected to be served.

Scenario IV. Two passengers with different OD pairs and two vehicles are present in the system and, due to the passengers' and vehicles' time windows, p_1 is assigned to v_1 and p_2 is assigned to v_2.

Scenario V. Three passengers are served by one vehicle, where passengers have different OD pairs with overlapping time windows. In this case, the vehicle serves all passengers in their preferred time windows through ride-sharing mode.

Scenario VI. One passenger and two vehicles are present in the system. In this case, two vehicles compete for serving the passenger. Ultimately, the vehicle whose routing is less costly wins the competition and serves the passenger.

Table 3.9
Six scenarios with various numbers of passengers and vehicles, OD pairs, and passengers' departure and arrival time windows.

Scenario	I	II	III	IV	V	VI
Number of passengers	2	2	2	2	3	1
Number of vehicles	1	1	1	2	1	2
$\boldsymbol{o_1}$	Node 2	Node 2	Node 2	Node 2	Node 2	Node 2
$\boldsymbol{d_1}$	Node 6	Node 6	Node 1	Node 1	Node 3	Node 6
$\boldsymbol{o_2}$	Node 5	Node 5	Node 3	Node 3	Node 5	-
$\boldsymbol{d_2}$	Node 3	Node 3	Node 6	Node 6	Node 3	-
$\boldsymbol{o_3}$	-	-	-	-	Node 6	-
$\boldsymbol{d_3}$	-	-	-	-	Node 1	-
$\boldsymbol{o'_1}$	Node 4	Node 4	Node 4	Node 2	Node 4	Node 4
$\boldsymbol{d'_1}$	Node 1	Node 1	Node 1	Node 1	Node 1	Node 1
$\boldsymbol{o'_2}$	-	-	-	Node 3	-	Node 6

d'_2	-	-	-	Node 6	-	Node 1
TW_{o_1}	[5, 7]	[5, 7]	[4, 5]	[4, 5]	[4, 7]	[4, 7]
TW_{d_1}	[9, 12]	[9, 12]	[8, 10]	[8, 10]	[13, 16]	[9, 12]
TW_{o_2}	[8, 10]	[16, 19]	[3, 5]	[4, 6]	[7, 10]	-
TW_{d_2}	[11, 14]	[21, 24]	[11, 14]	[11, 14]	[14, 18]	-
TW_{o_3}	-	-	-	-	[10, 13]	-
TW_{d_3}	-	-	-	-	[19, 23]	-
TH_{v_1}	[1, 30]	[1, 30]	[1, 30]	[1, 30]	[1, 30]	[1, 30]
TH_{v_2}	-	-	-	[1, 30]	-	[1, 30]

Table 3.10
Results obtained from testing our algorithm on the six-node transportation network for six scenarios.

iteration $\boldsymbol{k}$	$\boldsymbol{LB}^*$	$\boldsymbol{UB}^*$	gap%	vehicles assigned to $\boldsymbol{p_1}$, $\boldsymbol{p_2}$, and $\boldsymbol{p_3}$	$\boldsymbol{\lambda^k}(\boldsymbol{p_1})$	$\boldsymbol{\lambda^k}(\boldsymbol{p_2})$	$\boldsymbol{\lambda^k}(\boldsymbol{p_3})$
Scenario I. Two passengers are served by one vehicle through ride-sharing mode.							
1	1.47	5.75	74.5%	v_1, v_1, -	10	10	-
2	1.47	5.75	74.5%	v_1, v_1, -	5	5	-
3	5.75	5.75	0.0%	v_1, v_1, -	5	5	-
Scenario II. Two passengers are served by one vehicle (not through ride-sharing mode).							
1	1.47	7.22	79.68%	v_1, v_1, -	10	10	-
2	5.55	7.22	23.10%	v_1, v_1, -	5	5	-
3	7.22	7.22	0.0%	v_1, v_1, -	5	5	-
Scenario III. Two passengers and one vehicle; one passenger remains unserved.							
1	1.47	10.43	85.94%	v_1, v_2^*, -	10	10	-
2	7.1	10.43	31.95%	v_1, v_2^*, -	5	10	-
3	10.43	10.43	0.0%	v_1, v_2^*, -	5	10	-
Scenario IV. Two passengers and two vehicles; each vehicle is assigned to a passenger							
1	2.2	6.13	64.13%	v_1, v_2, -	10	10	-
2	2.2	6.13	64.13%	v_1, v_2, -	5	5	-
3	6.13	6.13	0.0%	v_1, v_2, -	5	5	-
Scenario V. Three passengers are served by one vehicle through ride-sharing mode							
1	1.47	6.97	78.95%	v_1, v_1, v_1	10	10	10
2	1.47	6.97	78.95%	v_1, v_1, v_1	5	5	5
3	6.97	6.97	0.0%	v_1, v_1, v_1	5	5	5
Scenario VI. Two vehicles compete for serving a passenger							
1	2.57	5.13	50.0%	v_1, -, -	10	-	-
2	2.63	5.13	48.70%	v_1, -, -	10	-	-
3	5.13	5.13	0.0%	v_1, -, -	10	-	-

Fig. 3.11 also presents the vehicle routing corresponding each scenario. We increase the number of passengers and vehicles to show the computational efficiency and solution optimality of our developed algorithm. Table 3.11 shows the results for the six-node

transportation network when the numbers of passengers and vehicles have been increased. The term "CPU" stands for central processing unit.

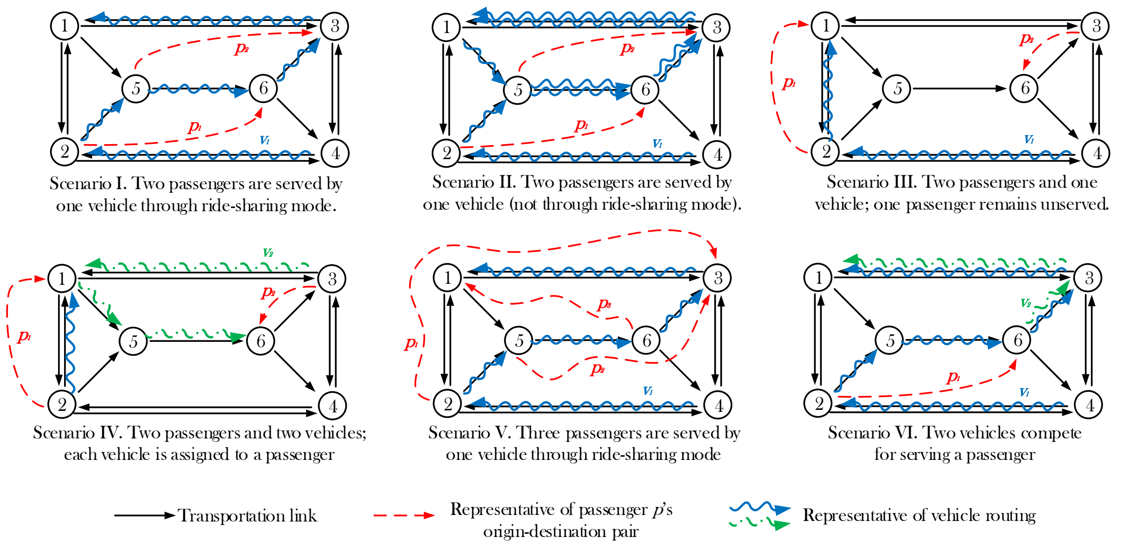

Scenario I. Two passengers are served by one vehicle through ride-sharing mode.
Scenario II. Two passengers are served by one vehicle (not through ride-sharing mode).
Scenario III. Two passengers and one vehicle; one passenger remains unserved.
Scenario IV. Two passengers and two vehicles; each vehicle is assigned to a passenger
Scenario V. Three passengers are served by one vehicle through ride-sharing mode
Scenario VI. Two vehicles compete for serving a passenger
Transportation link
Representative of passenger p's origin-destination pair
Representative of vehicle routing

Fig. 3.11. The vehicle routing corresponding each scenario.

Table 3.11
Results for the six-node transportation network.

Test case number	Number of iterations	Number of passengers	Number of vehicles	$\boldsymbol{LB}^*$	$\boldsymbol{UB}^*$	Gap (%)	Number of passengers not served	CPU running time (s)
1	30	6	1	15.83	15.83	0.00%	0	5.94
2	30	12	2	33.17	33.17	0.00%	0	12.02
3	30	24	4	61.67	65.33	5.61%	0	30.97

We explain the pricing mechanism in this algorithm via test case 1 with 6 passengers and 1 vehicle. Fig. 3.12 shows $\lambda^k(p_i)$, $i = 1,2,..,6$, along 30 iterations. It is clear that each passenger's Lagrangian multiplier ultimately converges to a specific value. This value can be literally interpreted as the passenger p's service price. Through the pricing mechanism of this algorithm, the provider would be able to offer a reasonable bid to its customers to be served.

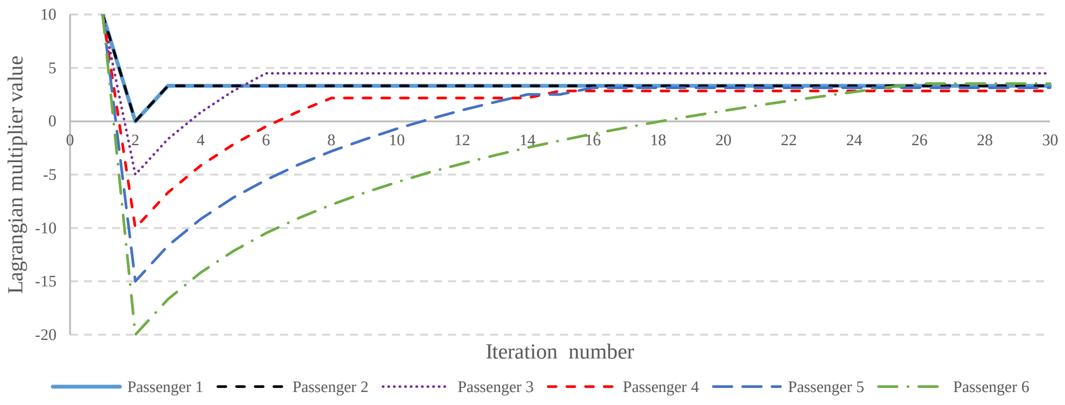


Fig. 3.12. Lagrangian multipliers along 30 iterations in test case 1 for the six-node transportation network.

3.6.2. Results from Medium-scale and Large-scale Transportation Networks

In our computational experiments for the medium-scale and large-scale networks, for simplicity, we assume that each passenger has a fixed departure time (the earliest and latest departure time are the same). In addition, we assume that no passenger has a preferred time window for arrival to his destination. Tables 3.12 and 3.13 show the results for the Chicago transportation network, shown as Fig. 3.13(a) with 933 nodes and 2,967 links, and the Phoenix transportation network, as shown in Fig. 3.13(b) with 13,777 nodes and 33,879 links, respectively.

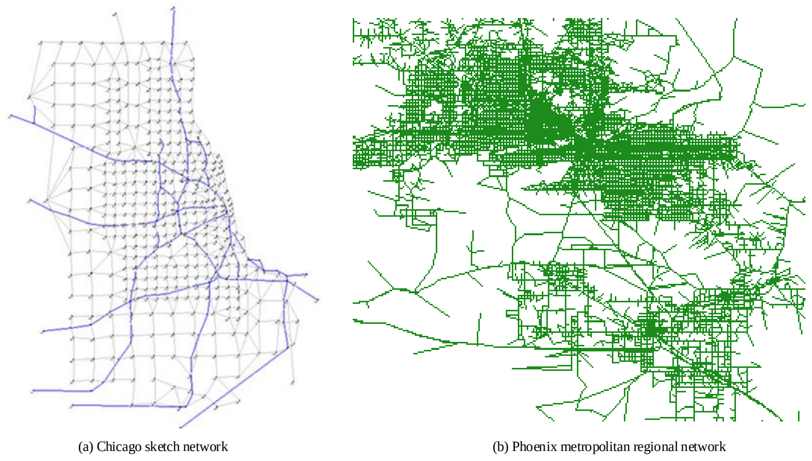

(a) Chicago sketch network
(b) Phoenix metropolitan regional network

Fig. 3.13. Medium and large-scale transportation networks for computational performance testing.

Note that we generally run the algorithm for a fixed number of iterations; however, the algorithm may converge in less number of iterations. Fig. 3.14 shows the gap percentage along 20 iterations corresponding each test case.

Table 3.12
Results for the Chicago network with 933 transportation nodes and 2,967 links.

Test case number	Number of iterations	Number of passengers	Number of vehicles	$\boldsymbol{LB^*}$	$\boldsymbol{UB^*}$	Gap (%)	Number of passengers not served	CPU running time (s)
1	20	2	2	108.43	108.43	0.00%	0	17.43
2	20	11	3	352.97	352.97	0.00%	0	91.87
3	20	20	5	616.66	626.18	1.52%	1	327.51
4	20	46	15	1586.81	1664.07	4.64%	2	4681.52
5	20	60	15	1849.98	1878.55	1.52%	3	7096.50

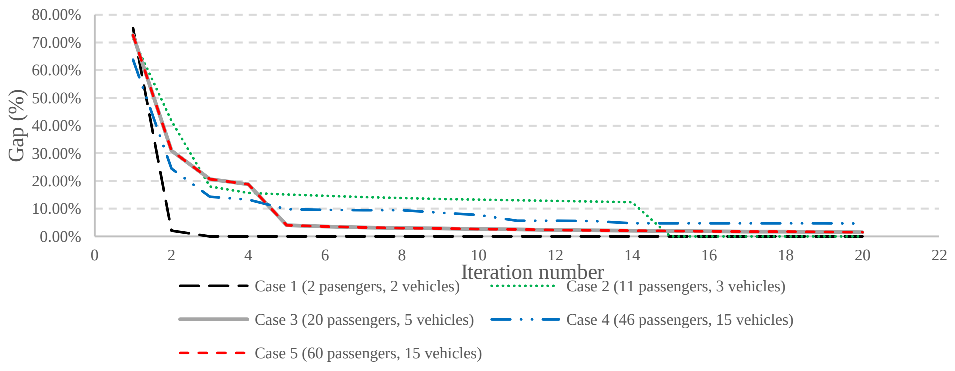


Fig. 3.14. Gap percentage along 20 iterations corresponding each test case in Chicago network.

As you can see in Fig. 3.14, after 10-15 iterations, the sub-gradient algorithm is typically able to converge to a small gap (about 5%) for the Chicago Network.

Table 3.13
Results for the Phoenix network with 13,777 transportation nodes and 33,879 links.

Test case number	Number of iterations	Number of passengers	Number of vehicles	$\boldsymbol{LB^*}$	$\boldsymbol{UB^*}$	Gap (%)	Number of passengers not served	CPU running time (s)
1	6	4	2	70.95	70.95	0.00%	0	110.39
2	6	10	5	191.55	207.05	7.49%	1	398.37
3	6	20	6	310.37	310.37	0.00%	0	1323.18
4	6	40	12	622.23	622.23	0.00%	0	3756.51
5	6	50	15	784.07	784.07	0.00%	0	6983.19

3.6.3. Handling Randomly Generated Test Instances

To further examine the computational efficiency and solution optimality of our proposed algorithm, we also test our algorithms on randomly generated instances proposed by Ropke and Cordeau (2009) which is publicly available at http://www.diku.dk/~sropke/. The data set introduced by Ropke and Cordeau (2009) is the modified version of instances employed by Ropke et al. (2007) initially proposed by Savelsbergh and Sol (1998). In this data set, the coordinates of passengers' pickup and drop-off locations are randomly selected and uniformly distributed over a $[0,50] \times [0,50]$ square. In addition, they considered a single depot located in the center of the square. The load q_i of passenger i is randomly selected from $[5, Q]$, where Q is the maximum capacity of the vehicle. A planning horizon $[0,600]$ is considered. Feasible departure and arrival time windows are also randomly generated for each passenger.

Ropke and Cordeau (2009) formulate the PDPTW on a network that is built based on demand request nodes, and the links are defined as direct connections between pickup and delivery nodes (without explicitly considering transportation links or paths), while we formulate the PDPTW on transportation networks. To test our algorithms on their data set, we need to convert their OD network to a transportation network. Specifically, we treat their demand node-oriented network as a transportation network, and each origin/destination node acts as a transportation node. As a result, in this converted transportation network, each transportation node is connected to all other transportation nodes, and similar to what we performed before, dummy nodes are added and connected to their corresponding transportation nodes. Obviously, the constructed transportation network is a complete digraph with a very large number of links. In this data set, the coordinates of passengers' pickup and drop-off locations are randomly chosen and uniformly distributed over a small square, so the densely distributed passengers could impose a difficult problem of assigning different vehicles to different passengers. In comparison, in our test data set, the Chicago and Phoenix transportation networks, the vehicles and passengers are naturally spatially and sparsely distributed such that fewer vehicles compete for serving a particular passenger. Thus, in that situation, the first stage of the vehicle assignment problem could be easily solved using our proposed LR framework with a good matching between the vehicles and passengers.

3.6.4. Challenges of Multi-vehicle Assignment Problems and Usefulness of Single-vehicle Routing Algorithm

In general, VRPPDTW even for the single vehicle cases is still categorized as one of the toughest tasks of combinatorial optimization (Azi et al., 2007; Hernández-Pérez and Salazar-González, 2009; and Häme, 2011). Several approaches have been recently suggested to resolve the issue mentioned above by converting multi-vehicle cases to the single-vehicle ones. For instance, Häme and Hakula (2015) have suggested a maximum cluster algorithm in which the multi-vehicle solution is based on a recursive single-vehicle algorithm.

To fully address the complexity of assigning different vehicles to multiple passengers, Fisher et al. (1997) proposed a new set of variables $x(p,v)$ and decomposed constraints (5) to two sets of constraints: constraints (3.19) and (3.20).

$\sum_{v\in(V\cup V^*)} x(v,p) = 1 \qquad \forall p \qquad (3.19)$

$\sum_{(j,s,w')} y(v,i,j,t,s,w,w') = x(v,p) \qquad (i,j,t,s,w,w') \in \Psi_{p,v}, \forall p, \forall v \qquad (3.20)$

Constraints (3.19) guarantee that each passenger is visited exactly once. Constraints (3.20) control vehicle v's route and show the relations between the variables $x(v,p)$ and $y(v,i,j,t,s,w,w')$. By using the Lagrangian decomposition method and relaxing these two sets of constraints into the objective function, the main problem can be decomposed to two sub-problems where sub-problem (1) becomes a semi-assignment problem and sub-problem (2) as a time-dependent least-cost path problem. The first sub-problem can be easily solved by inspection. The second sub-problem also can be solved by computationally efficient algorithms, e.g., proposed in our research. However, due to the integrality of x and y, there may be a gap between lower bounds and upper bounds of the primal problem. To further reduce the duality gap, Fisher et al. (1997) introduce a branch-and-bound method and use the variable splitting approach to control the lower bounds; but using a new branch-and-bound method decreases the computational efficiency of our algorithm dramatically.

In our research, in order to address the similar concerns, we also apply set partitioning approach to enumerate all possible passengers' service patterns. To define passengers' service patterns, we utilize the path representation for the TSP suggested by Bellman (1962) and Held and Karp (1962). Service pattern j is defined as a vector consists of $|P|$ number of elements (P is the set of passengers). Note that pth element of pattern j is representative of passenger p's service status. The service status of passenger p is chosen from the set {0,1,2}, where 0 means passenger p is still waiting to be picked up, 1 means passenger p has been picked up but the service has not been completed, and 2 means passenger p has been successfully delivered. Let c_{vj} denote the travel cost of pattern j traversed by vehicle v. Moreover, assume that α_{vpj} is a binary constant which equals 1 if pattern j traversed by vehicle v includes passenger p, and 0 otherwise. z_{vj} is also a binary variable equals 1 if pattern j is used by vehicle v, and 0 otherwise. Thus, we will have:

$$min \sum_{v \in V} \sum_{j \in J} c_{vj} z_{vj} \tag{3.21}$$

s.t.

$$\sum_{v \in V} \sum_{j \in J} \alpha_{vpj} z_{vj} = 1 \qquad \forall p \tag{3.22}$$

$$\sum_{j \in J} z_{vj} = 1 \qquad \forall v \tag{3.23}$$

$$z_{vj} \in \{0,1\} \qquad \forall v, j \tag{3.24}$$

In this formulation, objective function (3.21) minimizes the total travel cost. Constraints (3.22) guarantee that each passenger is served exactly once. Constraints (3.23) ensure that each vehicle selects only one pattern. Constraints (3.24) define that the decision variables are binary. In order to assess the solution optimality of our developed algorithm on instances proposed by Ropke and Cordeau (2009) and avoid the computational challenges, two scenarios have been examined. First, we test our algorithm for the single-vehicle cases to avoid the complexity of assigning different vehicles to multiple passengers. In this case, it is obvious that we do not need to apply the set partitioning method mentioned above. Second, we test our algorithm on the small subsets of their instances for the multiple-vehicle cases with a limited number of transportation requests, so that all possible combinations of passenger-to-vehicle assignment patterns α_{vpj} can be enumerated, and then solved in the set partitioning problem defined above. In both sets of scenarios, the exact solutions are obtained for all the restricted master problems in the test data sets.

3.7. Conclusions

A new generation of transportation network companies uses mobile-phone-based platforms to seamlessly connect drivers to passengers from different origins to different destinations with specific, preferred departure or arrival times. Many relevant practical aspects need to be carefully formulated for real-world planning/dispatching system deployment, such as time-dependent link travel times on large-scale regional transportation networks, and tight vehicle capacity and passenger service time window constraints.

By reformulating the PDPTW through space-time networks to consider time window requirements, our proposed approach can not only solve the vehicle routing and scheduling problem directly in large-scale transportation networks with time-dependent congestion, but also avoid the complex procedure to eliminate any sub-tour possibly existing in the optimal solution for many existing formulations. By further introducing virtual vehicle constructs, the proposed approach can fully incorporate the full set of interacting factors between passenger demand and limited vehicle capacity in this model to derive feasible solutions and practically important system-wide cost-benefit estimates for each request through a sub-gradient-based pricing method. This joint optimization and pricing procedure can assist transportation network service providers to quantify the operating costs of spatially and temporally distributed trip requests.

On a large-scale regional network, the capacity impact of optimized passenger-to-vehicle matching results can be further evaluated in mesoscopic dynamic traffic simulation packages such as an open-source Dynamic Traffic Assignment-Light weight (DTALite) (Zhou and Taylor, 2014). Future work will concentrate on the development of the model for the following cases: (i) Passengers may desire different ride-sharing capacities (i.e. a passenger may desire to share his ride with up to only one passenger, whereas the other passenger may have no restriction about the number of passengers which share their ride with him). (ii) A passenger may desire to be or not to be served by a particular vehicle. (iii) A transportation request could contain a group of passengers who have the same origin, while they may or may not have the same destination. Alternatively, a transportation request could contain a group of passengers who have the same destination, while they may or may not have the same origin. In this case, we are interested in adding dummy nodes corresponding to passengers' origins and destinations more wisely and efficiently. In addition, in our future research, a comprehensive branch-and-bound algorithm should be included in our solution framework to fully address the complexity of assigning different vehicles to multiple passengers.

CHAPTER 4

A CUMULATIVE SERVICE STATE REPRESENTATION FOR THE PICKUP AND DELIVERY PROBLEM WITH SYNCHRONIZED TRANSFERS

4.1. Introduction

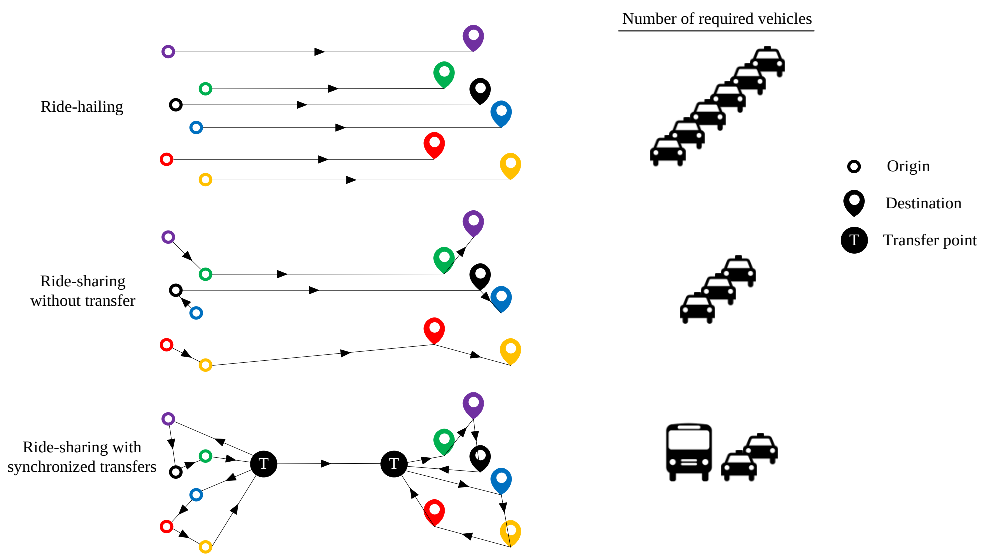
Coordinated transportation services consist of three different levels of service: ride-hailing, ride-sharing without transfer, and ride-sharing with one or more than one synchronized transfer. Ride-hailing is a level of coordinated service in which a passenger hires a driver to get a transportation service for a fee, and the driver is supposed to deliver the passenger to exactly where he needs to go. Traditional taxi companies offer this form of transport. The way by which a passenger hails a car can be listed as follows: a passenger can hail a taxi from the street, call up a transport service on the phone, or hail a car from an app by his cellphone.

Ride-sharing without transfer is another level of coordinated transportation service which is slightly different from ride-hailing. In this mode of transportation, similar to the ride-hailing, a passenger hires a driver to take him exactly where he needs to go, but the passenger may share his ride with one or more than one passenger. Recently, a broad range of transportation network companies like Uber, Lyft, and Sidecar offers this type of transport service by the aid of three recent technological advances: (1) GPS, (2) smartphones, and (3) social networks.

The third level of coordinated transportation service is ride-sharing with synchronized transfers. In general, transfers are used to provide more efficient transportation networks by reducing the operational costs, as well as making more flexible routes available for passengers. A large number of daily trips is classified in this category. For instance, in multi-modal transit, a passenger may use two or more transport modes for his trip (e.g. train and bus). Multi-modal transit provides convenient and economical connection of various modes to make a complete journey from origin to destination. Another example of this type of transportation service can be the first mile/last mile transport of the commuters who need to go from an origin to a transit station and then from the station at the other end of the trip to a final destination. Ride-sharing between households or fellow workers is another example of ride-sharing with synchronized transfers. In this case, members of a family or any other social group arrange their trips informally and share their travel information such as departure time, stops, and transfer points among themselves.

The concept of pickup and delivery with transfer(s) is not only used in passenger transit, but also applied frequently in freight transportation in what is called “freight consolidation”. Freight consolidation is when several small shipments, all being forwarded to the same location, are bundled and shipped together. Freight consolidation is a service offered by some shipping companies to lower the total shipping cost and to increase shipping security. It is also known as consolidation service, assembly service, and cargo consolidation. Different terms are used for transfer centers depending on the transportation mode and the type of good being transported, such as rail yards in the railway industry, hub airports in air cargo transportation, transshipment ports in sea cargo transportation, and terminals for transporting goods by trucks.

In Fig. 4.1, different levels of coordinated transportation service have been shown by an example in which six passengers with different origins, destinations, and departure time windows have called for service. As shown in Fig. 4.1, in the case of ride-sharing with synchronized transfers, the vehicles' capacity can be utilized more in comparison to other types of coordinated transportation service.


Number of required vehicles
Ride-hailing
Ride-sharing without transfer
Ride-sharing with synchronized transfers
Origin
Destination
Transfer point
T
Fig. 4.1. Different levels of coordinated transportation service.

What motivated us to study the pickup and delivery problem with transfers for passenger transit was the fast-growing ride-sharing mode of transportation. In the last decade, ride-sharing companies have introduced a new mode of transportation which is much more convenient than public transit and less expensive than taxi. By introducing connected and autonomous vehicles to this mode of transportation and eliminating the cost of hiring drivers, it is expected that in the near future, ride-sharing becomes a good substitution of public transit for daily trips of middle-class families. Knowing passengers' trip itinerary in advance and offering incentives to those passengers who are flexible about their departure/arrival time and number of stops during their trip will help service providers to schedule more efficient trips for serving their customers.

A number of studies have focused on showing the usefulness of transfer and schedule coordination in the pickup and delivery problem, to name a few, Mitrovic´-Minic´and Laporte (2006), Qu and Bard (2012), Masson et al. (2013), Kim and Schonfeld (2014), Sun and Schonfeld (2016), and Ghilas et al. (2016). In general, ride-hailing and ride-sharing without transfers can be mathematically modeled by the VRPPDTW. The VRPPDTW is a combinatorial optimization problem that searches for an optimal set of routes for a fleet of vehicles to serve a set of transportation requests. Each request is a combination of pickup at the origin and drop-off at the destination within particular time windows.

Several algorithms have been suggested for solving the VRPPDTW. For instance, Dumas et al. (1991) used a set-partitioning model to minimize the total travel cost considering tight vehicle capacity constraints, as well as time windows and precedence constraints. They proposed a CG scheme with a constrained shortest path as a sub-problem to construct admissible routes. Savelsbergh and Sol (1998) developed a branch-and-price algorithm to minimize the total number of vehicles needed to serve all transportation requests as the primary objective, and minimize the total distance traveled as the secondary objective. In addition, Lu and Dessouky (2004), Cordeau (2006), and Ropke et al. (2007) proposed branch-and-cut algorithms to minimize the total routing cost. Ropke and Cordeau (2009) also presented a branch-and-cut-and-price algorithm in which the lower bounds are controlled by a CG scheme and strengthened by introducing several valid inequalities to the problem. Baldacci et al. (2011) proposed a new exact algorithm based on a set-partitioning formulation improved by additional cuts to minimize total routing costs. In a recent clustering algorithm proposed by Häme and Hakula (2015), the multi-vehicle routing solution is obtained by calling a recursive single-vehicle algorithm, based on the passenger-to-vehicle assignment from the first clustering stage.

In terms of algorithmic development, a number of studies have focused on solving the VRPPDTW by the DP approach. For instance, the classical work by Psaraftis (1980) presented an exact backward DP solution algorithm for the single-vehicle routing problem with pickup and delivery with time windows to minimize a weighted combination of the total service and waiting time for passengers with $O(n^2 3^n)$ complexity, where n denotes the total number of passengers in the system. Psaraftis (1980) proposed a passengers' service state representation that was adapted from the path representation for the TSP proposed by Bellman (1962) and Held and Karp (1962). Psaraftis (1983) further modified the algorithm to a forward DP approach with the same space complexity. Desrosiers et al. (1986) proposed a forward DP algorithm for the single-vehicle routing problem with pickup and delivery with time windows to minimize the total distance traveled to serve all passengers. Recently, by the aid of LR solution framework, Mahmoudi and Zhou (2016) have proposed a forward DP solution-based algorithm to minimize the total routing costs of the single vehicle sub-problems on a three-dimensional SST network. Their time-dependent single-vehicle state is jointly defined by the passengers' carrying state, the current node being visited, and the time to embed time windows and vehicles capacity constraints in a well-structured hyper-network. Furthermore, their special three-dimensional network representation for the multi-vehicle routing problem with pickup and delivery with time windows reduces the space complexity of the DP solution algorithm from the exponential order, i.e., 3^n where n is the total number of passengers, to the much smaller space requirement of $\sum_{k=0}^{Cap_v} C_k^n$, where Cap_v is vehicle v's capacity and C_k^n is the number of k-combinations from n passengers. Note that Cap_v is not a large number in practice (e.g. 2 or 3 for taxi).

Despite the extensive research done before on ride-hailing and ride-sharing without transfers, few studies have focused on ride-sharing with transfers. Ride-sharing with transfers is observed as pickup and delivery problems with transshipments/transfers (PDPT) in the literature. A number of previous research articles have focused on solving PDPT by heuristic/meta-heuristic algorithms. For example, the practice in a large San Francisco-based courier company motivated Mitrovic´-Minic´and Laporte (2006) to present an empirical study on the effectiveness of transfer points in the pickup and delivery problem. Since the company was serving a large area covering several neighboring cities, they were allowing transshipment of loads between vehicles to keep drivers in their home area. In their study, they found circumstances under which transfers may be useful. They applied a two-phase heuristic (a construction phase followed by an improvement phase) to solve the problem. Another practice in a regional air carrier for finding daily route planning inspired Qu and Bard (2012) is to examine the usefulness of transshipment. In their study, they developed a greedy randomized adaptive search procedure (GRASP) to handle this complex problem. Masson et al. (2013) and Masson et al. (2014) also proposed an adaptive large neighborhood search for solving the PDPT. They tested their algorithm on real-life instances related to the transportation of mentally or physically disabled people. They showed that adding the concept of transfer to the pickup and delivery problem can provide significant improvements to the value of the objective function. Recently, Ghilas et al. (2016) have proposed an adaptive large neighborhood search heuristic algorithm for the PDPT. They also showed the merits of using transfers in the pickup and delivery problem by testing their algorithm on sets of generated instances.

A number of research articles also have focused on reaching exact solutions for this problem. For example, Mues and Pickl (2005) developed a path-based mixed integer programming model for the PDPT and applied a CG solution approach to solve the model. Cortés et al. (2010) proposed a mixed integer programming model for the PDPT in which passengers have different options for transfer from one vehicle to another at particular transfer nodes. They used a branch-and-cut algorithm based on Benders decomposition to solve the model. In three major papers on this topic by Drexl (2012a), (2012b), and (2013), different types of synchronization (i.e., task synchronization, operation synchronization, movement synchronization, load synchronization, and resource synchronization) have been extensively discussed. Recently, Rais et al. (2014) proposed a mixed integer programming formulation for the PDPT with and without time windows for services in which heterogeneous vehicles and flexible fleet size are allowed. They used the commercial solver GUROBI, which uses the simplex method, on linear-programming relaxations combined with branch-and-cut and branch-and-bound techniques to solve the mixed integer programming model. We present their mathematical model in Appendix B to better show our motives for building hyper-networks. Dealing with several constraints, especially those related to the validity of the time and load variables in their model, prompted us to look at this challenging problem from a different angle.

In this research, we add time dimension to the space graph to not only track the location of vehicles at any time, but also impose passengers' preferred pickup and delivery time windows, synchronization time points, and precedence constraints to the problem. Defining time as an explicit dimension and physical transportation networks as the base of our proposed hyper-networks help us to handle networks with links whose travel time may vary over the time of day or over the load of vehicles (e.g. HOV or HOT lanes). We also add another dimension, called the "passengers' cumulative service state," to the constructed space-time graph to track the service status of requests at any time and impose the coupling and precedence constraints to the model. In brief, we aim to have the following three aspects of contributions.

First, in terms of mathematical modeling contributions, we propose a new mathematical model for the pickup and delivery problem with different types of synchronization in which heterogeneous vehicles and flexible fleet size are allowed. Based on our SST network representation, we apply a DP solution algorithm to embed the vehicle-to-task assignment constraints and provide exact solutions for small scale problems (more precisely, a pseudo optimal solution due to the time discretization). Our SST network representation prevents sub-tours as well as infeasibility in the final solutions and mitigates the solution symmetry issue that could occur in a broader class of LR-based solution framework (e.g., Sherali and Smith, 2001; Mahmoudi and Zhou, 2016).

More importantly, to address the issues of the curse of dimensionality, we demonstrate a consistent transition from the microscopic cumulative service states to macroscopic cumulative flow count diagrams, which can be used to effectively estimate the overall dynamic system performance. It should be remarked that the concept of cumulative flow count diagram is widely applied as a representation of dynamic activities in traffic science literature. It generally concerns the cumulative flow count of vehicles passing through a transportation system, in which vehicle concentrations, queue sizes, travel times, and delays are the main measures of the system performance evaluation. Newell (1982) and Hall (1991) successively documented the corresponding methodologies in a systematic way. Edie and Foote (1960) first designed cumulative curves to describe the cumulative flow count of vehicles, and Gazis and Potts (1963) used a cumulative diagram as a predictive tool for the first time. Newell (1993) further presented a three-dimensional version of a cumulative diagram regarding space, time, and cumulative flow, to merge the concepts of cumulative diagrams with wave theory (Makigami et al. 1971). Daganzo (2001) presented a seminal study in his book for extending N curves to study the dynamics and stability of supply chain systems. Our cumulative flow count approach sheds more light on a recent development direction in the field of discrete-time integer programming (Boland et al., 2017a), which aims to iteratively refine and find an optimal continuous-time solution without explicitly modeling the microscopic state changes along the discrete time dimension.

Third, with the consistent microscopic and macroscopic system representation, our model and algorithm can effectively handle large-scale real-world instances and generate a good initial solution to be applied for our Lagrangian heuristic. Then, our Lagrangian heuristic evaluates the price of each synchronized transfer and guides a fast search for real-world test cases with about 10,000 delivery orders. We can also impose road capacity constraints to the model in order to encourage synchronized transfers and reduce the number of vehicles and corresponding traffic congestion impact.

The rest of the chapter is organized into the following sections. Section 4.2 contains the problem statement and assumptions for the PDPT. In Section 4.3, we initially explain the clustering phase and present our proposed multi-commodity network flow programming model for the PDPT. We will further explain how to improve vehicles' performance by finding optimal chains of work pieces. We also explain our motives for applying the hyper-network structure in this section. Section 4.4 provides computational results over the instances applied by Ropke and Pisinger (2006) and the real-world data set proposed by Cainiao Network (logistics service provider to Alibaba Group) to demonstrate the computational efficiency of our developed algorithm coded in C++. We conclude the chapter in Section 4.5 with discussions on possible extensions.

4.2. Problem Statement for the PDPT

The pickup and delivery problem with time windows searches for an optimal set of routes for a fleet of vehicles to serve a set of transportation requests. Each request is a combination of pickup and drop-off within particular time windows. In the case of ride-hailing and ride-sharing without transfer, passenger j is picked up and dropped off only once. In the case of ride-sharing with synchronized transfers, passenger j is picked up and dropped off more than once. In other words, passenger j's trip request contains more than one OD pair. In this case, the OD pair can be a journey from the passenger's origin to a transfer point, from a transfer point to another transfer point, or from a transfer point to its destination.

4.2.1. Assumptions

For any feasible solution in the PDPT, the following statements must hold:

- Every vehicle must start its route from its starting depot at the time when its work shift starts and end the route at its ending depot at the ending time of its work shift.
- Every request must be served exactly once.
- For every OD pair, the origin must be visited before the destination (precedence constraint).
- If a passenger reaches a transfer point by vehicle v, then he must leave the transfer point by vehicle v', such that vehicle v arrives at the transfer point before vehicle v' leaves it (transfer synchronization).

– For every OD pair, the pickup/drop-off locations must be visited within the passenger’s preferred departure/arrival time windows.

– For every OD pair, the journey from origin to destination must be done by a single vehicle (coupling constraint).

– Every vehicle must not exceed its capacity.

– Road capacity must not be violated.

We have presented the mixed integer programming model for the PDPT proposed by Rais et al. (2014) in Appendix B. The existing mixed integer-programming model for the PDPT contains several constraints related to the validity of the time and load which make the problem difficult to solve for a large number of passengers. In Section 4.3.7, we comprehensively discuss our motives for building hyper-networks for solving this problem. In the next section, we show how to construct a hyper-network for the PDPT.

4.3. Our Proposed Hyper-network for the PDPT

The main thrust of this chapter is how to construct a network such that the concept of assignment and routing in the PDPT are seamlessly integrated.

4.3.1. Space-time Network

We define the problem on a transportation network, which includes a set of nodes (e.g. intersections or freeway merge points) and a set of directed links with different types (e.g. freeway segments, arterial streets, or ramps). Each directed link has time-dependent travel time.

In order to distinguish physical transportation nodes from passengers' pickup and drop-off locations and vehicles' origin and destination depots, dummy nodes corresponding to these spots should be added to the transportation network (Mahmoudi and Zhou, 2016). It is necessary to distinguish pickup/drop-off locations and vehicles' depots from ordinary traverse points (regular transportation nodes), because pickup/drop-off spots and vehicles' depots have time window restrictions, while ordinary traverse nodes do not. Moreover, we will explain that based on our definition for OD pairs' cumulative service states, pickup and drop-off actions (state transitions) must occur only from/at particular nodes which should be differentiated from regular transportation ones.

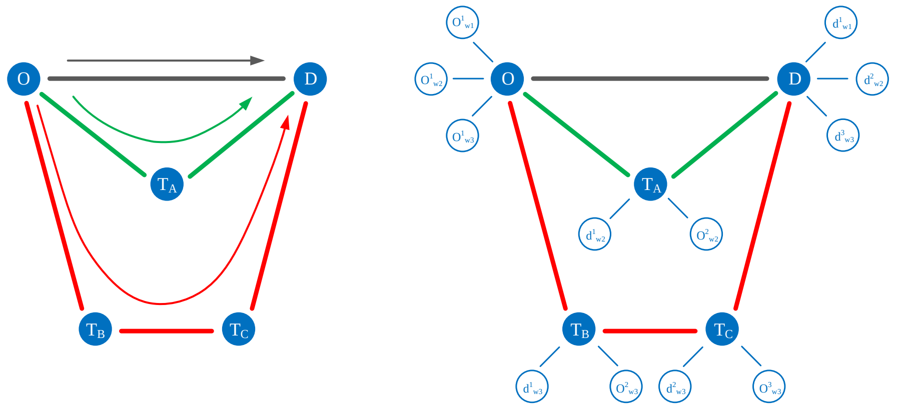
In the case of ride-hailing and ride-sharing without transfer, only two dummy nodes corresponding to passenger j's origin and destination should be added to the network. In the case of ride-sharing with synchronized transfers, for each OD pair, one dummy node for pickup and one for drop-off should be added to the network. These extra pickup/drop-off jobs occurring at transfer points must be agreed upon by the passenger and transportation service provider. Passengers may agree to trips with one or more than one transfer by receiving some incentives from the service provider such as lower priced service or a particular bonus for their future trips.

In this chapter, we assume that a set of ways for passenger j has already been agreed upon by the passenger and the transportation service provider. A "way" is defined as a sequence of landmarks (i.e., his origin), predefined transfer points (if existing), and his destination. These ways not only should be time-feasible routes (there should be at least one time-feasible route starting from his origin, passing through all corresponding predefined transfer points (if any), and ending at his destination) and have the potential to provide profit for the service provider, but also should be acceptable to the passenger. Suppose a passenger who does not want to share his ride with any other passenger has called for the service. We know that if his request is served by ride-sharing, it would be more profitable than when it is served by ride-hailing. However, since ride-sharing is not acceptable to the passenger, the set of ways by which this passenger can be served is limited to ride-hailing.

Fig. 4.2(a) illustrates three ways by which passenger j can be served. The first way, indicated by grey color, is a non-stop trip, the second way is a one-stop trip stopping at transfer point A, indicated by green color, and the last way is a two-stop trip stopping at transfer points B and C, indicated by red color. Each way contains passenger j's origin, transfer point(s) (if any), and his destination. The price of each way is different from each other. We assume that the passenger has no priority in selecting his way among all available ways. That is why the service provider decides and assigns a way to him to be served. The selected way must eventually provide more profit for the service provider in comparison to other ways.

In general, each passenger has at most 3 different ways to be served in our experiments. Note that there are several passengers in our experiments that only want to be served by non-stop trips. This is similar to what we observe in practice. Although any transportation node can be a transfer point for a passenger, we have a limited number of connection options for him to be transferred in reality.

Suppose $\Omega(j)$ is the set of ways by which passenger j makes a complete journey from origin to destination. Let $\omega_n(j)$ denote n^{th} way by which passenger j's request is fulfilled, and $OD_{\omega_n(j)}$ denote the set of OD pairs in $\omega_n(j)$. Then, $OD^m_{\omega_n(j)}$ defines the m^{th} OD pair in $\omega_n(j)$. Let $o^m_{\omega_n(j)}$ and $d^m_{\omega_n(j)}$ denote dummy nodes corresponding to passenger j's origin and destination in the m^{th} OD pair of $\omega_n(j)$, respectively. Each dummy node is only connected to its corresponding physical transportation node by a link. The travel time of this link is interpreted as the service time. We call these links pickup/drop-off links. Fig. 4.2(b) demonstrates a network in which dummy nodes corresponding to each OD pair have been added to the network.



(a) Three ways by which passenger j can be served.

(b) Dummy nodes corresponding each OD pair has been added.

Fig. 4.2. (a) Three ways, i.e., $O \rightarrow D$, $O \rightarrow T_A \rightarrow D$, $O \rightarrow T_B \rightarrow T_C \rightarrow D$, by which passenger j can be served; (b) dummy nodes corresponding to each OD pair have been added to the physical transportation network.

Dummy nodes $o^m_{\omega_n(j)}$ and $d^m_{\omega_n(j)}$ have exclusive time windows, denoted by $[a_{o^m_{\omega_n(j)}}, b_{o^m_{\omega_n(j)}}]$ and $[a_{d^m_{\omega_n(j)}}, b_{d^m_{\omega_n(j)}}]$, respectively, where $a_{o^m_{\omega_n(j)}}$ and $b_{o^m_{\omega_n(j)}}$ are the earliest and latest departure times from $o^m_{\omega_n(j)}$, and $a_{d^m_{\omega_n(j)}}$, and $a_{d^m_{\omega_n(j)}}$ are earliest and latest arrival times at $d^m_{\omega_n(j)}$, respectively.

In addition, let o_v and d_v denote dummy nodes corresponding to the origin and destination depots for vehicle v, respectively. Again, each dummy node is only connected to its corresponding physical node by a link. The travel time for this link is interpreted as the preparation time.

In order to illustrate space-time networks, we initially map the two-dimensional space graph (physical transportation network with added dummy nodes) to a one-dimensional space, at which all nodes are positioned in a row. Then we add the time dimension, where the time horizon has been discretized into a series of time intervals with the same time length. In general, time discretization may affect the solution optimality; however, in this chapter, we have assumed one minute as the unit of time, and one minute in comparison with OD pairs' departure/arrival time windows and ride time are quite small such that time discretization may damage optimality with a low possibility. However, in order to be precise about the optimality of solutions, we use the term "pseudo-optimal" for the solution obtained from solving the time-dependent least cost path problem.

Thus far, we have a discretized space-time graph. Note that if vehicle v arrives at a passenger's pickup/drop-off location early, it should wait until the passenger's departure/arrival time window starts, while arriving late to these nodes is not permitted (hard time windows). In addition, if it arrives at its ending depot earlier than the ending time of the work shift, it should wait until its planning horizon ends, and arriving later than that time is not allowed.

In the classical study by Psaraftis (1980), he presented an exact backward DP solution algorithm for the single-VRPPDTW to minimize a weighted combination of the total service and waiting time for passengers. Psaraftis (1983) further developed a forward recursion scheme in his DP solution algorithm to deal with passengers' time windows. However, his approach is not easily used for the VRPPDTW. In his chapter, the state representation consists of the location currently being visited and the service status of passengers. The service status of passengers is chosen from set $\{1,2,3\}$, where 3 means passenger j is still waiting to be picked up, 2 means passenger j has been picked up, but the service has not been completed, and 1 means passenger j has been successfully delivered. In the next section, we add another dimension, called the "passengers' cumulative service state" to the constructed space-time graph to track the service status of OD pairs at any time.

4.3.2. Cumulative Service State

In our study, we adapt the Bellman-Held-Karp path representation scheme (Bellman (1962) and Held and Karp (1962)) in the TSP to define passengers' service patterns. In their study, the passengers' service patterns consist of two terms: the node currently being visited and the cumulative service state (only visit). We extend the first term to the node currently being visited at time t, and the second term to the more complicated cumulative service state, i.e., "pickup" and "drop-off". Then, the status of $OD^{m}_{\omega_n(j)}$ is an element chosen from set $\{0,1,2\}$. In this set, 0 means passenger j is still waiting to be picked up, 1 means passenger j has been picked up but the trip has not been completed, and 2 means passenger j has been successfully delivered. It should be noted that our schema is different from set {1,2,3} used by Psaraftis (1980), as 0 in our case can better denote unserved status of an OD pair across different vehicle layers.

After defining the cumulative service states for OD pairs, the next step is defining all feasible state transitions. According to our definition for the cumulative service state, there are a limited number of feasible state transitions from state s to s', since several transitions from state s to s' violate the activity precedence constraints (e.g. if drop-off occurs before pickup action) or vehicles' capacity constraints.

Suppose two passengers have called for a taxi. Fig. 4.3 illustrates a number of feasible and infeasible state transitions for this example. According to our definition for cumulative service states, state transitions illustrated in Fig. 4.3(a) and Fig. 4.3(b) are feasible. Fig. 4.3(a) shows that both passengers are waiting to be picked up at state s, while at state s', passenger 1 has been picked up but his trip has not been completed yet. In Fig. 4.3(b), at state s, passenger 2 has already been served, and passenger 1 has already been picked up but his service has not been completed yet. At state s', both passengers have been served. The state transition shown in Fig. 4.3(c) is infeasible due to the violation of passenger 1's operations precedence constraint.

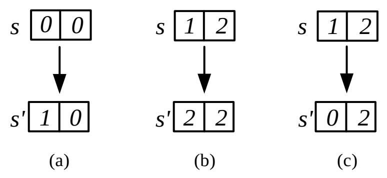


Fig. 4.3. (a) Feasible state transition in which passenger 1 is picked up, while passenger 2's service has not started yet; (b) feasible state transition in which passenger 1 is dropped off while passenger 2 has already been served; (c) infeasible state transition due to the violation of passenger 1's operations precedence constraint.

4.3.3. Microscopic Hyper-network Construction

A vertex in the hyper-network is an object of the form $(\mathrm{v}, \mathrm{i}, \mathrm{t}, \mathrm{s})$, where v is a vehicle index, i is a location index (it can be either a physical transportation node or a dummy node), $t \in \{t_{start}, \dots, t_{end}\}$ is a time, and s is the OD pairs' cumulative service state. Clearly, not all $(\mathrm{v}, \mathrm{i}, \mathrm{t}, \mathrm{s})$-tuples will form feasible hyper-nodes. In this section, we will define a set of rules determining when such a tuple is feasible.

Fig. 4.4 illustrates a number of important network constructs in our proposed multi-vehicle hyper-network by an example. Fig 4.4(a) presents an illustrative three-node transportation network with bi-directional links. Assume two OD pairs are supposed to be served in this example. Both OD pairs must be picked up from node 1 and delivered to node 3; however, their departure and arrival time windows are different from each other. Suppose two vehicles are available to serve these OD pairs. Both have the same origin depot (node 1) and destination depot (node 3). First of all, dummy nodes corresponding to OD pairs' pickup and drop-off locations, as well as vehicles' starting and ending depots should be added to the physical transportation network (Fig. 4.4(b)). Then, the two-dimensional space graph (XY plane) is mapped to a one-dimensional space network in which all nodes are positioned in a row (Fig. 4.4(c)). In the third step, the cumulative service state and vehicle-time are added as new dimensions to the one-dimensional space network (Fig. 4.4(c)). According to our explanation for OD pairs' cumulative service state, 3^2 states may occur in this example, i.e., [0,0], [1,0], [0,1], [1,1], [0,2], [2,0], [1,2], [2,1], and [2,2].

In this step, a unique block is generated for each vehicle. Several interior layers comprising OD pairs' cumulative service states are added along the vehicle time horizon within each block. Each block consists of two exterior layers, so-called opening and ending layers, whose task is transmitting the information related to the cumulative service state from the ending layer of the current block/vehicle to the opening layer of the next block/vehicle. In the two-passenger, two-vehicle example shown in Fig. 4.4, similar to a runner in a relay race, vehicle v_1 transmits the information related to the cumulative service state from node d_v on its ending layer at the end of its work shift, denoted by t_{end}, to the next vehicle's origin depot, o_v, on its opening layer at the beginning of vehicle v_2's time horizon, denoted by t_{start}. If a vehicle picks up an OD pair, it should make a complete journey from origin to destination. That is why we do not see states [1,0], [0,1], [1,1], [1,2], and [2,1] at the opening and ending layers of blocks. Here, we define a set of rules determining when $(\mathrm{v}, \mathrm{i}, \mathrm{t}, \mathrm{s})$-tuples are feasible.

Rule 1. The total number of vehicles is given. Index v should not exceed the total number of vehicles.

Rule 2. Node i is considered into vehicle v's hyper-network, if and only if it is accessible to nodes o_v *and* d_v during the time horizon. In this chapter, we have written a separate module, called a "space-time prism", to determine the set of accessible nodes for each vehicle. In this module, we use a label-correcting algorithm to solve a time-dependent minimum spanning tree problem.

Rule 3. Node i is only feasible within its particular time window. All nodes in the hyper-network have time window $[t_{start}, t_{end}]$, except dummy nodes corresponding to passengers' origin and destination locations, at which desired departure and arrival time windows should be imposed, respectively.

Rule 4. At the opening and ending layers of blocks, incomplete tasks should not be observed. In other words, there should not be any OD pair with cumulative service state "1" in state s on the exterior layer of a block.

Rule 5. State s should not violate vehicle v*'s* capacity constraint. This means that the total number of "1" at any time and node must not exceed vehicle v*'s capacity.*

An arc from vertex $(\mathrm{v}, \mathrm{i}, \mathrm{t}, \mathrm{s})$ to vertex $(\mathrm{v}', \mathrm{i}', \mathrm{t}', \mathrm{s}')$ in a hyper-network is also feasible if:

Rule 1. $v' = v$ if and only if $\mathrm{t}' \neq t_{start}$; $v' = v + 1$ if and only if $\mathrm{t}' = t_{start}$. Note that *Rule 1* for feasible vehicle index v and v' should not be violated as well.

Rule 2. Both vertexes $(\mathrm{i}, \mathrm{t}, \mathrm{s})$ and $(\mathrm{i}', \mathrm{t}', \mathrm{s}')$ must be feasible and reachable for vehicle v's hyper-network.

Rule 3. The physical link between adjacent nodes i and i' must exist.

Rule 4. $\mathrm{t}' = \mathrm{t} + TT(i, i', t)$, where $TT(i, i', t)$ is the link travel time from node i to node i' starting at time *t*.

Rule 5. State transition only occurs at a dummy node corresponding to pickup or drop-off locations within its time window (through a service arc). In other words, $\mathrm{s} = \mathrm{s}'$ if vehicle v is traversing a transportation arc.

Rule 6. State transition from state s to state s' should be feasible. We have explained this rule comprehensively in Section 4.3.2.

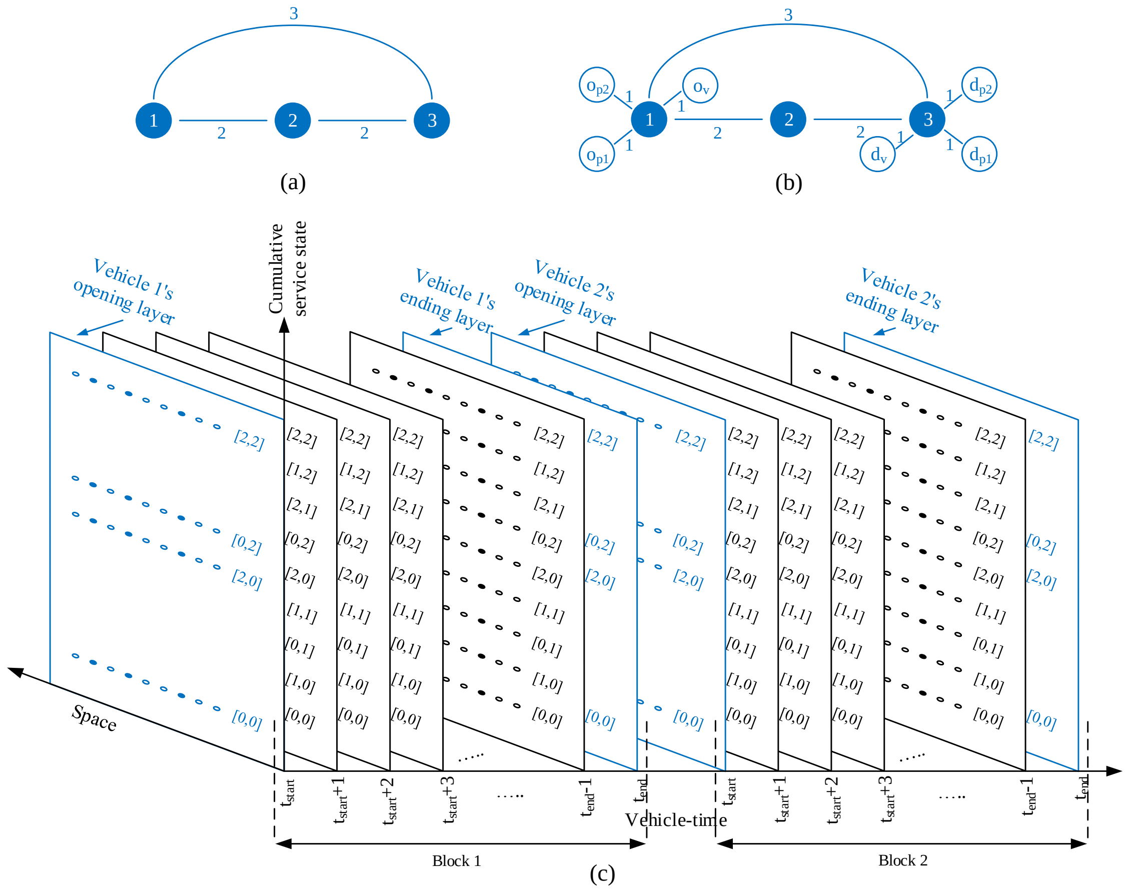


Fig. 4.4. (a) A three-node transportation network; (b) modified network with dummy nodes for two vehicles and two passengers; (c) three-dimensional service SST network.

Due to the complexity of the constructed hyper-network, our model is able to solve the PDPT to optimality (more precisely, pseudo-optimality due to the time discretization) for a limited number of OD pairs and vehicles. In order to handle a large set of OD pairs, we suggest the traditional cluster-first, route-second approach in which the large-sized primary problem is broken into a number of small-sized sub-problems, where the most compatible OD pairs are clustered together. In the next section, we will go through the details of the clustering phase, and afterwards, we explain how to conduct the PDPT on the constructed hyper-network.

4.3.4. Finding Potential Matchings to Cluster Transportation Requests

Clustering passengers based on their geographical proximity is broadly used either a priori or in parallel with the routing process. Cullen et al. (1981) proposed a cluster-first, route-second heuristic in which the clustering and routing sub-problems are solved by a CG technique. A few years later, Bodin and Sexton (1986) also developed another cluster-first, route-second heuristic for the PDPTW. Passengers are first partitioned into clusters, and then, by using the algorithm proposed by Sexton and Bodin (1985a, 1985b), the algorithm constructs a tour on each cluster. Finally, the algorithm swaps passengers between routes and performs route re-optimizations.

In fact, finding high-quality clusters without having some levels of routing information is a difficult task. That is why Dumas et al. (1989) introduced the concept of mini-clusters, where passengers with spatio-temporal closeness are clustered together. In their algorithm, a heuristic provides a set of mini-clusters. Then, a CG algorithm is used to optimally combine mini-clusters into vehicle routes. Finally, the algorithm performs route re-optimizations in order to obtain optimal routing and scheduling for each vehicle. Desrosiers et al. (1991) also proposed another way of constructing mini-clusters by a parallel insertion method based on spatio-temporal proximity of the requests. Ioachim et al. (1995) later applied an optimization-based technique instead of a heuristic for constructing mini-clusters. Pankratz (2005), Bard and Jarrah (2009), Qu and Bard (2012), and Masson et al. (2013) are a few out of many examples of studies which applied a clustering algorithm for the PDPTW.

In order to find the well-matched OD pairs, thanks to the explicit definition of the time dimension in our network, we utilize the three-dimensional space (XY plane)-time network representation and further apply an effective rule to explore all potential matchings.

We calculate the space-time distance between each OD pair's pickup location to other OD pairs' pickup spots. We do a similar calculation for each OD pair's drop-off location to other OD pairs' drop-off spots. In order to find these space-time distances, we initially consider the middle time of $o^m_{\omega_n(j)}$'s departure time window, i.e., $\frac{a_{o^m_{\omega_n(j)}}+b_{o^m_{\omega_n(j)}}}{2}$, as $o^m_{\omega_n(j)}$'s departure time, denoted by $t_{o^m_{\omega_n(j)}}$, and the middle time of $d^m_{\omega_n(j)}$'s arrival time window, i.e., $\frac{a_{d^m_{\omega_n(j)}}+b_{d^m_{\omega_n(j)}}}{2}$, as $d^m_{\omega_n(j)}$'s arrival time, denoted by $t_{d^m_{\omega_n(j)}}$. We also set β_1 as the weight of the geographical distance (\$/mile) and β_2 as the weight of the temporal distance (\$/min) to weight the time and space dissimilarities in a uniform way in the space-time distance calculation. Since different values of β_1 and β_2 result in different clusters, we play with these values to generate variant clusters. Suppose $f(OD^m_{\omega_n(j)}, OD^{m'}_{\omega_{n'}(j')})$ is the space-time distance between $o^m_{\omega_n(j)}$ and $o^{m'}_{\omega_{n'}(j')}$, and $h(OD^m_{\omega_n(j)}, OD^{m'}_{\omega_{n'}(j')})$ is the space-time distance between $d^m_{\omega_n(j)}$ and $d^{m'}_{\omega_{n'}(j')}$. Then, $f(OD^m_{\omega_n(j)}, OD^{m'}_{\omega_{n'}(j')})$ and $h(OD^m_{\omega_n(j)}, OD^{m'}_{\omega_{n'}(j')})$ are calculated as follows:

$$f\left(OD^{m}_{\omega_n(j)}, OD^{m'}_{\omega_{n'}(j')}\right) = \beta_1 \times \sqrt{(x_{o^{m}_{\omega_n(j)}} - x_{o^{m'}_{\omega_{n'}(j')}})^2 + (y_{o^{m}_{\omega_n(j)}} - y_{o^{m'}_{\omega_{n'}(j')}})^2} + \beta_2 \times \left| t_{o^{m}_{\omega_n(j)}} - t_{o^{m'}_{\omega_{n'}(j')}} \right| \tag{4.1}$$

$$h\left(OD^{m}_{\omega_n(j)}, OD^{m'}_{\omega_{n'}(j')}\right) = \beta_1 \times \sqrt{(x_{d^{m}_{\omega_n(j)}} - x_{d^{m'}_{\omega_{n'}(j')}})^2 + (y_{d^{m}_{\omega_n(j)}} - y_{d^{m'}_{\omega_{n'}(j')}})^2} + \beta_2 \times \left| t_{d^{m}_{\omega_n(j)}} - t_{d^{m'}_{\omega_{n'}(j')}} \right| \tag{4.2}$$

where $x_{.}$ and $y_{.}$ are the x-coordinate and y-coordinate of the corresponding spot. Note that in the formulations mentioned above, the first term calculates the geographical distance, while the second term computes the temporal distance between two nodes.

After calculating these two values, we calculate $r(OD^{m}_{\omega_n(j)}, OD^{m'}_{\omega_{n'}(j')}) = max\{f\left(OD^{m}_{\omega_n(j)}, OD^{m'}_{\omega_{n'}(j')}\right), h\left(OD^{m}_{\omega_n(j)}, OD^{m'}_{\omega_{n'}(j')}\right)\}$ as the measure of the dissimilarity between $OD^{m}_{\omega_n(j)}$ and $OD^{m'}_{\omega_{n'}(j')}$. Fig. 4.5 illustrates an example in which three OD pairs have a potential matching and may be defined in the same cluster.

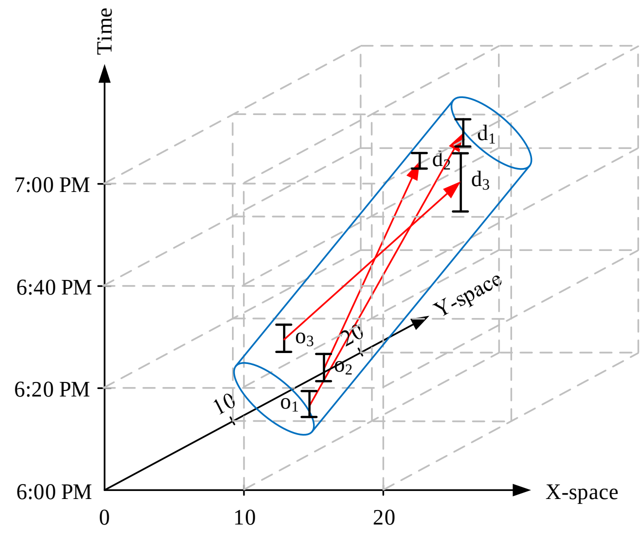


Fig. 4.5. An example of three OD pairs with potential matching.

In this step, each OD pair is considered as an individual cluster. As a result, we can obtain the level of dissimilarity between $OD^m_{\omega_n(j)}$ and cluster q by $r(OD^m_{\omega_n(j)}, q)$. It is clear that the dissimilarity between $OD^m_{\omega_n(j)}$ and its corresponding cluster is equal to 0. Let α denote the maximum number of OD pairs solvable in each cluster by our dynamic programing algorithm. Note that we have applied several dominance rules in our DP algorithm to improve the value of α. The highest value of α solvable by our DP algorithm is 35 OD pairs. Variable y_q equals 1 if cluster q exists, and 0 otherwise. Variable $z(OD^m_{\omega_n(j)}, q)$ is also equal to 1 if $OD^m_{\omega_n(j)}$ is assigned to cluster q, and 0 otherwise. We intend to not only minimize the mismatches between OD pairs of each cluster, i.e., $\sum_q \sum_j \sum_n \sum_m \{r(OD^m_{\omega_n(j)}, q) \times z(OD^m_{\omega_n(j)}, q)\}$, but also minimize the total number of clusters, i.e., $(\sum_q y_q)$. Therefore, we set ζ_1 as the weight of the former term and ζ_2 as the weight of the latter to weight these two factors in a uniform way in the objective function of the clustering problem. Since different values of ζ_1 and ζ_2 result in different clusters, we adjust these values to generate variant clusters.

Constraints (4.4) express that each cluster must contain up to α number of OD pairs. Moreover, each OD pair must be assigned to exactly one cluster (constraint (4.5)). Therefore, we will have:

$$Min\left\{\zeta_1 \times \sum_q \sum_j \sum_n \sum_m \left\{r(OD^m_{\omega_n(j)}, q) \times z(OD^m_{\omega_n(j)}, q)\right\} + \zeta_2 \times \sum_q My_q\right\} \qquad (4.3)$$

subject to:

$$\sum_j \sum_n \sum_m z(OD^m_{\omega_n(j)}, q) \leq \alpha y_q \qquad \forall q \qquad (4.4)$$

$$\sum_q z(OD^m_{\omega_n(j)}, q) = 1 \qquad \forall OD^m_{\omega_n(j)} \qquad (4.5)$$

$$z(OD^m_{\omega_n(j)}, q) \in \{0,1\};\ y_q \in \{0,1\} \quad \forall q, OD^m_{\omega_n(j)} \qquad (4.6)$$

The above integer programming problem can be solved by any commercial solver such as CPLEX, GAMS, or GUROBI. In our experiments, we use GAMS Distribution 23.00.

4.3.5. Routing Inside the Clusters

After clustering the OD pairs, we need to set initial prices for them to encourage vehicles to depart from their depots to serve OD pairs and collect the profits. The initial price of $OD^m_{\omega_n(j)}$ can be set as a ratio of its transportation cost, when $OD^m_{\omega_n(j)}$ is served by ride-hailing (taxi). Note that in order to make this service profitable for any form of coordinated transportation service, we consider the cost of the ride-hailing mode of transportation, which is the most expensive mode of transportation for the requester.

Moreover, a ratio value (mentioned above) greater than 1 is ideally implemented for a conservative approach.

The objective function of the PDPT may vary from one study to another based on the main focus of the problem. For example, some studies aim to satisfy all demands with fewer vehicles, while others attempt to maximize the number of requests that can be served by a fixed number of vehicles. Therefore, the former problem is minimizing costs subject to full demand satisfaction, while the latter is maximizing satisfied demand subject to vehicle availability. In fact, the latter case is more practical.

One of the ways by which the lack of vehicles can be handled is surge pricing. Surge pricing is how rideshare companies aim to control supply and demand, and it happens when there is a high demand for vehicles (i.e., lots of passengers are looking for a ride in the same area) while there are not enough vehicles to satisfy all of the passengers. The goal of surge pricing is to incentivize vehicles to perform trips during the busiest hours of the day. Surge pricing fixes this excess demand by applying a multiplier on every fare, therefore raising prices by certain percentages. As a result, some passengers will opt to not pay the higher fare, making more vehicles available for those passengers who are willing to pay the added expense. Surge pricing can happen at any time of the day, but it is most common during rush hour, bad weather, holidays, and weekends - all times when there is a sense of urgency to get a ride.

Another way to handle a deficit in supply is by serving some of the demand with the available vehicles and then using extra vehicles (e.g. virtual vehicles (taxis) with higher operating cost) if necessary. Note that operating costs may include fuel, maintenance, depreciation, insurance costs, and more importantly, the cost of hiring full-time or part-time drivers. Other objective functions observed in the literature include, but are not limited to: minimization of difference between actual and desired delivery times (e.g. Bodin and Sexton, 1986), minimization of differences between actual and shortest possible ride times (e.g. Bodin and Sexton, 1986), minimization of total route duration (e.g. Dumas et al. 1989; Desrosiers et al. 1991; Ioachim et al. 1995), minimization of total route length (e.g. Cordeau and Laporte, 2003), minimization of vehicles' idle time (e.g. Diana and Dessouky, 2004), minimization of user inconvenience (e.g. Coslovich et al. 2006; Melachrinoudis et al. 2007), or a weighted combination of those mentioned above.

At this stage, we assume that the total number of vehicles in each cluster is known, and the goal is to satisfy as many OD pairs as possible. Note that the vehicles serving OD pairs at this stage are not real and have been defined just for the sake of OD pairs' routing inside each cluster. We assume that all these hypothetical vehicles are homogenous. Let $\bar{o}_q$ and $\bar{d}_q$ denote the average point (with the average XY coordinates) of all OD pairs' origins and destinations in cluster q, respectively. Then, we assume that the starting and ending depots of all vehicles in cluster q are located at the nearest physical transportation node to points $\bar{o}_q$ and $\bar{d}_q$, respectively. We also need to guarantee that the hypothetical vehicles in cluster q have enough time to reach all OD pairs' origins and destinations. In other words, not serving an OD pair in cluster q is not due to the OD pair's inaccessibility to the depots. That is why we assume that all vehicles in cluster q start their routes from "the least earliest departure time of all OD pairs in cluster q minus a sufficient tolerant time interval", denoted by t_{start}^{q}, and end them at "the largest latest arrival time of all OD pairs in cluster q plus a sufficient tolerant time interval", denoted by t_{end}^{q}. The sufficient tolerant time interval has been set as 20 minutes in our computational experiments.

After defining the time horizon as well as the starting and ending depots for hypothetical vehicles, the question may arise: how many vehicles should be assigned in each cluster? The answer is that for the first iteration, we set the number of vehicles in each cluster by the following method. We assume that the capacity of all hypothetical vehicles in all clusters is 4. Then, we assume that 75% of the seats are occupied. Since all hypothetical vehicles fleet for a one-way trip from the starting depot to the ending depot, we can assume that each hypothetical vehicle can serve 3 passengers during its whole trip. That is why, in the first iteration, the total number of hypothetical vehicles in cluster q is obtained by $\left\lceil \frac{total\ number\ of\ OD\ pairs\ in\ cluster\ q}{3} \right\rceil$. Then, this value is tuned within our algorithm by dynamic system performance obtained from a cumulative flow count diagram which will be explained in Section 4.3.10.

So far, we clustered OD pairs and assigned a number of hypothetical vehicles to serve the OD pairs inside each cluster. Then, we construct a hyper-network for each cluster, in which each hypothetical vehicle starts its route from o_v at time t_{start}^{q} and, depending on the price of OD pairs, serves as many OD pairs as possible. Finally, the vehicle must end its route to d_v at time t_{end}^{q}, and transmit the information related to the cumulative service state to the next vehicle. This is exactly a time-dependent state-dependent least cost path problem whose mathematical model is provided in the next section.

Note that in our proposed hyper-network structure, all hypothetical vehicles have been assumed to be homogenous; therefore, there is no preference for assigning vehicles to OD pairs. We will discuss this point in Section 4.3.11 and explain that several hyper-paths are similar and should not be scanned more than once. Since the hypothetical vehicles perform in a serial fashion (not in a parallel manner) inside each cluster, there is no competition among vehicles for serving OD pairs. In fact, each vehicle just tries to serve as many remaining and unserved OD pairs as possible.

The serial structure for vehicle-time blocks helps us to overcome the symmetry issue which is common in most combinatorial optimization problems. To briefly explain this issue, suppose vehicles v and v' are identical in terms of starting and ending depots, work shift, and capacity. Despite the fact that, from a practical point of view, it does not matter whether passenger j is served by vehicle v or v', the computational procedure spends plenty of time exploring the vertexes of these two vehicles' network separately. As a result, many regions which are symmetric to the parts that have been already examined are unnecessarily scanned.

One common and effective method of handling symmetries is to introduce symmetry breaking constraints to the main problem, which directs the system to not search within symmetric solutions (Walsh 2012 and Raviv et al. 2013). In addition to symmetry-breaking constraints, CG reformulation is a widely used technique for eliminating symmetry. In fact, this is one of the major benefits of CG approaches in VRPTW and related problems. In this chapter, by the aid of our passengers' cumulative service patterns and well-structured hyper-network, we are able to implicitly impose the symmetry breaking constraints to the problem. To sum up, in each cluster, transmitting the information related to the cumulative service state of OD pairs from one block to another and our unique definition for feasible state transitions prevent future vehicles from approaching those OD pairs which have already been served in previous blocks. This is exactly what is needed to prevent the symmetry issue.

4.3.6. Time-discretized Multi-commodity Network Flow Programming Model for the PDPT in Each Cluster

Based on the constructed hyper-network that can capture vehicles' capacity constraints, as well as desired departure and arrival time windows and precedence constraints, we now start constructing a multi-commodity network flow programing model for the local clusters derived from the original large scale real-world instances. Table 4.1 lists the notations for the sets, indices, and parameters in the pickup and delivery problem with time windows (with/without transfers). We use i, t, s to represent the space-time-service state vertex, and the corresponding arc which is i, i', t, t', s, s'. The model uses binary variables $x_{i,i',t,t',s,s',v,v'}$ equal to 1 if arc (i, i', t, t', s, s') is used by vehicle ($v \in V^q$), and 0 otherwise. Note that in general, $v = v'$ unless vehicle v is involved with transmitting the information related to s from vertex $\left(d_v, t_{end}^q\right)$ on its own block to vertex (o_v, t_{start}^q) on the next block (vehicle v'). The objective is to find a set of minimum-cost vehicle routes to satisfy as many OD pairs as possible. Then, the problem can be mathematically modeled as follows:

Table 4.1
Sets, indices, and parameters in our proposed model for the PDPT.

Symbol	Definition
V^q	Set of hypothetical vehicles in cluster q, where $V^q = \{v_1^q, v_2^q, \dots, v_{\lvert V^q \rvert}^q\}$
S^q	Set of feasible states in the exterior layers of hypothetical vehicles in cluster q
(i, i')	Index of the physical link between adjacent nodes i and i'
$TT(i, i', t)$	Link travel time from node i to node i' starting at time t
t_{start}^q	Hypothetical vehicles' beginning time in cluster q
t_{end}^q	Hypothetical vehicles' ending time in cluster q
s, s'	The OD pairs' cumulative service states at vertexes (i, t) and (i', t')
Cap_v	Maximum capacity of vehicle v
$Cap_{i,i',t}$	Maximum road outflow capacity on the link from node i to node i' starting at time t
$o_{\omega_n(j)}^m$	Dummy node corresponding to passenger j's origin in mth OD pair of $\omega_n(j)$
$d_{\omega_n(j)}^m$	Dummy node corresponding to passenger j's destination in mth OD pair of $\omega_n(j)$
$[a_{o_{\omega_n(j)}^m}, b_{o_{\omega_n(j)}^m}]$	Departure time window for $o_{\omega_n(j)}^m$
$[a_{d_{\omega_n(j)}^m}, b_{d_{\omega_n(j)}^m}]$	Arrival time window for $d_{\omega_n(j)}^m$
o_v	Dummy node for the starting depot of hypothetical vehicles in cluster q
d_v	Dummy node for the ending depot of hypothetical vehicles in cluster q
$c_{i,i',t,t',s,s'}(v)$	Routing cost of arc (i, i', t, t', s, s') traveled by hypothetical vehicle v

$$Min \ \sum_{(v,i,i',t,t',s,s')} \{c_{i,i',t,t',s,s'}(v) \times x_{i,i',t,t',s,s',v,v}\} \tag{4.7}$$

subject to:

$$\sum_{i',t',s'} x_{o_v,i',t_{start}^q,t',\phi,s',v_1^q,v_1^q} = 1 \tag{4.8}$$

$$\sum_{i,t,s,s':s' \in S^q} x_{i,d_v,t,t_{end}^q,s,s',v_{\lvert V^q \rvert}^q,v_{\lvert V^q \rvert}^q} = 1 \tag{4.9}$$

$$\sum_{v',i',t',s'} x_{i,i',t,t',s,s',v,v'} - \sum_{v',i',t',s'} x_{i',i,t',t,s',s,v',v} = 0 \quad \forall i,t,s,v \tag{4.10}$$

$$\sum_{v,t',s,s'} x_{i,i',t,t',s,s',v,v} \leq Cap_{i,i',t} \qquad \forall (i,i'); v \in V^q; \ t \in [t_{start}^q, t_{end}^q] \tag{4.11}$$

$$x_{i,i',t,t',s,s',v,v'} \in \{0,1\} \qquad \forall v,v',i,t,s,i',t',s' \tag{4.12}$$

The objective function (4.7) minimizes total routing costs. Constraints (4.8) to (4.10) ensure flow balance on every vertex in the hyper-network. Constraint (4.11) guarantees that the road capacity is not violated. Constraint (4.12) defines the binary decision variables.

Note that if link (i,i') is fully occupied by a queue of vehicles, the queue length on that link should be less than or equal to $L_{i,i'} \times k_{jam}$, where $L_{i,i'}$ is the length of link (i,i') in terms of lane-miles and k_{jam} is jam density in terms of the number of vehicles per mile per lane. This constraint can be written as follows:

$$\sum_{v,s,s',0\leq\delta\leq t} x_{i,i',\delta,\delta+FFTT(i,i'),s,s',v,v} - \sum_{v,s',s'',0\leq\delta\leq t} x_{i',i'',\delta,\delta+FFTT(i',i''),s',s'',v,v} \leq L_{i,i'} \times k_{jam}\ \forall(i,i');\ t \in [t^q_{start}, t^q_{end}] \quad (4.13)$$

where δ is an integer time index between 0 and t, and $FFTT(i,i')$ is the free flow travel time on link (i,i'). The first term of this inequality (starting from the left) is the cumulative vehicle arrivals at link (i,i') at time t, $A(i,i',t)$, and the second term, is the cumulative vehicle departures from link (i,i') at time t, $D(i,i',t)$. We can also implement a cumulative flow count diagram and simplified kinematic-wave-based modeling approach to handle delays caused by the propagation of backward shock waves. Therefore, we will have:

$$A(i,i',t) - D(i,i',t - BWTT(i,i')) \leq L_{i,i'} \times k_{jam} \quad \forall(i,i');\ t \in [t^q_{start}, t^q_{end}] \quad (4.14)$$

where $BWTT(i,i')$ is the travel time needed for a backward wave to propagate through link (i,i') assuming a simplified triangular Q-K model. Since constraints (4.13) and (4.14) are out of scope of this chapter, we did not consider them in the main body of our model. Interested readers can find further details about these constraints in the original paper by Newell (1993) and in the recent papers by Zhou and Taylor (2014) and Li et al. (2015).

4.3.7. Our Motives for Building Hyper-networks

In this chapter, we intend to embed various constraints of the pickup and delivery problem with time windows (with/without transfers) on a three-dimensional SST network in which time and load are explicitly added as new dimensions to the physical transportation network. We show that our hyper-network structure not only handles large-scale transportation networks with links whose routing cost (travel time) may vary over the time of day (based on the real-time traffic conditions), but also performs on the networks in which the routing cost of links is load-dependent (e.g. HOV or HOT lanes).

In addition, the distinctive structure of our proposed multi-dimensional network allows us to mathematically model different forms of coordinated transportation service. In fact, introducing passengers' cumulative service state as an independent dimension to the space-time network enables us to distinguish the ways by which a passenger can be served. We will further apply a Lagrangian heuristic to determine the price of each OD pair considering the way by which it is supposed to be fulfilled (e.g. through Lagrangian multipliers). The Lagrangian heuristic evaluates the price of each OD pair and guides a fast search.

Note that our proposed SST network representation is able to solve the problem to optimality (more precisely, pseudo-optimality due to the time discretization) for a limited number of passengers due to the exponential order of passengers' cumulative service state. Therefore, in order to handle a real-world transportation network with a large set of passengers, we must split the large-sized primary problem into a number of small-sized sub-problems in which the most compatible trips are clustered together. In order to find well-matched passengers, we utilize the three-dimensional space (XY plane)-time network representation and apply a rational rule to explore all potential matchings. To define passengers' cumulative service patterns within each cluster, we utilize the path representation schema for the TSP proposed by Bellman (1962) and Held and Karp (1962). We further relax the group of hard constraints by which we guarantee that each passenger is served exactly once. As a result, the problem is converted to a state-dependent time-dependent least cost path problem which can be solved by computationally-efficient algorithms already proposed for solving the least cost path problem. Here, we develop a forward DP solution-based approach across multiple vehicles to reach optimality (more precisely, pseudo-optimality due to the time discretization) within a cluster. As a final point, by introducing passengers' cumulative service patterns to the problem, we are able to tackle the symmetry issue which is common in combinatorial problems. We have comprehensively discussed this point in Section 4.3.5.

4.3.8. Routing Outside the Clusters

Our clustering procedure may raise the question of whether a vehicle may operate in more than one cluster. The answer is 'yes'. In fact, if the space-time vertex at which $v_{v'}^{q'}$ picks up its first passenger is reachable for the space-time vertex at which v_v^q drops off its last passenger, then the service tasks of $v_{v'}^{q'}$ and v_v^q can be performed by a single vehicle. Therefore, vehicles will be utilized more during the day. Fig. 4.6 illustrates this procedure by an example in which seven passengers are clustered in three groups and served by a single vehicle.

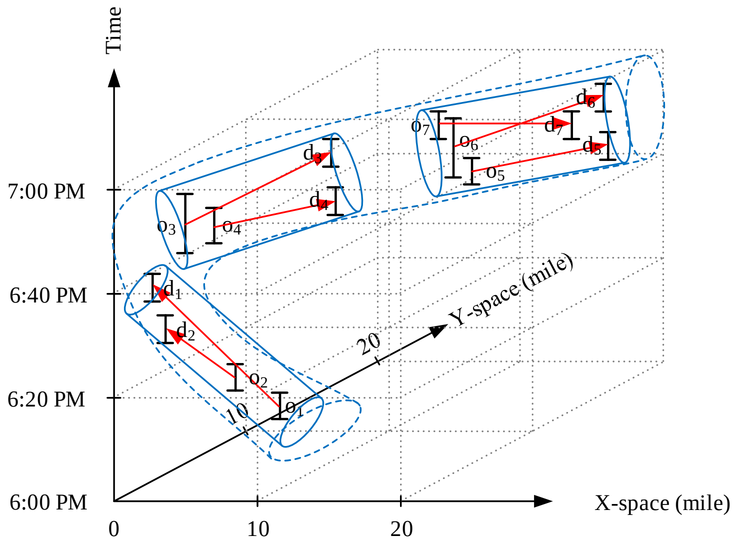


Fig. 4.6. An illustration of three clusters performed by one vehicle.

At this stage, we work with the set of heterogonous real vehicles distributed in the transportation network. These real vehicles whose origin/destination depots, work shift, and capacity are known beforehand and are provided by service providers. In this phase, we need to find the optimal chain of work pieces/tasks that can be performed by each real vehicle. Note that each work piece has already been completed by a hypothetical vehicle. We assume that all real vehicles are used and no one remains idle. We also assume that if the space-time vertex at which $v_{v'}^{q'}$ picks up its first passenger is reachable for the space-time vertex at which v_v^q drops off its last passenger, then the work piece completed by v_v^q

is connected to the work piece done by $v_{v'}^{q'}$ by link $(v_v^q, v_{v'}^{q'})$. The cost of the link connected from v_v^q's work piece to $v_{v'}^{q'}$'s work piece is defined as follows: {The cost of routing (including transportation and waiting costs) from the space-time vertex at which v_v^q drops off its last passenger to the space-time vertex at which $v_{v'}^{q'}$ picks up its first passenger, plus the profit that the real vehicle gains by taking work piece of vehicle $v_{v'}^{q'}$}. In order to simplify the network, we assume that if vehicle v_v^q needs to wait more that 30 minutes to start vehicle $v_{v'}^{q'}$'s task, then these two work pieces should not be connected. Suppose o_v^R and d_v^R denote the origin and destination depots of real vehicle v. Then, variable $x(u,w)$ is equal to 1 if link (u,w) is selected, and otherwise is 0. Index u can be an origin depot of a real vehicle or a hypothetical vehicle's work piece from a cluster, while w is a destination depot of a real vehicle or a hypothetical vehicle's work piece from a cluster. Let $c(u,w)$ denote the cost of link (u,w). The objective is minimizing the cost while assigning the real vehicles to the chains of work pieces. This will be obtained by considering objective function (4.15) and flow balance constraints (4.16)-(4.18). Moreover, each work piece must be selected at most once (constraints (4.19)).

$$Min \sum_{u,w} c(u,w) \times x(u,w) \tag{4.15}$$

subject to:

$$\sum_w x(o_v^R, w) = 1 \quad \forall o_v^R \tag{4.16}$$

$$\sum_u x(u, d_v^R) = 1 \quad \forall d_v^R \tag{4.17}$$

$$\sum_w x(u,w) - \sum_w x(w,u) = 0 \quad \forall u; u \neq o_v^R, d_v^R \tag{4.18}$$

$$\sum_w x(u,w) \leq 1 \quad \forall u, u \neq o_v^R, d_v^R \tag{4.19}$$

$x(u,w) \in \{0,1\} \qquad \forall u,w$ (4.20)

From a CG solution framework perspective (Dumas et al., 1991), our approach only examines the chains containing *optimal* routes of multiple vehicles, so called "*long*" columns, in comparison with the commonly used single-vehicle "*short*" columns in a generic CG scheme. Thus, our algorithms can be viewed as an adaption of CG algorithm that operates on a small set of long columns with passenger-to-vehicle assignment-routing solutions.

4.3.9. Lagrangian Heuristic for OD Pairs' Price Adjustment

After solving the optimization problem in Section 4.3.8 by any commercial solver such as CPLEX, GAMS, or GUROBI, we may find 3 different situations for passenger j's request:

(1) The request has been successfully served by exactly one way.

(2) The request has not been served by any ways completely. In this case, the request may be not touched by any vehicle at all, or maybe some OD pairs from one or more than one way have been served by some vehicles, but complete journey from passenger j's origin to destination has not occurred.

(3) The request has been completely served by more than one single way.

Since the goal is to reach situation (1) for as many passengers as possible in the long run, adjusting the price of OD pairs in each iteration is a necessity. Here, for every passenger, we define a set of cuts crossing particular pickup links to reach this target, which are called "leg-covering cuts". Cuts are usually defined for pruning, such as the definition of cuts in a branch-and-cut algorithm, but here a leg-covering cut is a partition of the node set N into two subsets S and $\bar{S}$, where N is the set of physical transportation nodes and dummy nodes, S is a set of particular dummy nodes corresponding passenger j's pickup action and $\bar{S} = N - S$. This definition of cuts is similar to what we have for flows and cuts in the maximum flow problem.

First of all, we identify the way by which passenger j is served with the highest number of transfer points. Assume this way contains τ number of transfers, so $\tau + 1$ leg-covering cuts must be defined. Then, for every single stage that passenger j can be picked up, a leg-covering cut is defined to guarantee that passenger j is only served by one OD pair. These cuts not only ensure passenger j is served by a single way, but also prevent half-finishing ways. Let $\Xi(j)$ denote the set of leg-covering cuts defined for passenger j. Let $\xi_l(j)$, $(\xi_l(j) \in \Xi(j))$ is the l^{th} cut crossing pickup links of passenger j's OD pairs. Suppose $A_{\xi_l(j)}$ is a set of pickup links that are cut by $\xi_l(j)$. Then, constraints (4.21) ensure that passenger j is served by only one way and prevent half-finishing ways.

$$\sum_{v,(i,i',t,t',s,s') \in A_{\xi_l(j)}} x_{i,i',t,t',s,s',v,v} = 1 \qquad \forall \xi_l(j) \in \Xi(j) \tag{4.21}$$

Back to the example illustrated in Fig. 4.2, where way $\omega_3(j)$ is the way with the highest number of transfers (i.e., 2). So, we need to define three leg-covering cuts that are $A_{\xi_1(j)} = \{(o^1_{\omega_1(j)}, O), (o^1_{\omega_2(j)}, O), (o^1_{\omega_3(j)}, O)\}$, $A_{\xi_2(j)} = \{(o^1_{\omega_1(j)}, O), (o^2_{\omega_2(j)}, T_A), (o^2_{\omega_3(j)}, T_B)\}$, and $A_{\xi_3(j)} = \{(o^1_{\omega_1(j)}, O), (o^2_{\omega_2(j)}, T_A), (o^3_{\omega_3(j)}, T_C)\}$ shown in Fig. 4.7.

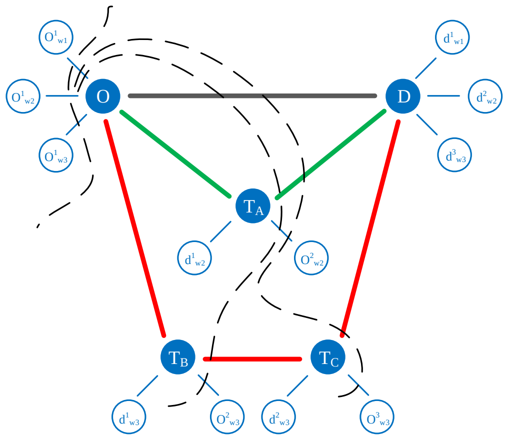


Fig. 4.7. Three leg-covering cuts to guarantee that passenger j is served by exactly one way.

Finally, we tune the price of every leg-covering cut for each passenger, and subsequently adjust the price of each OD pair involved with the corresponding leg-covering cut by using the sub-gradient method. Let k denote the iteration number, and $\theta^k(\xi_l(j))$ and $\lambda^k(\xi_l(j))$ denote the step size and Lagrangian multiplier corresponding to cut $\xi_l(j)$ at iteration k, respectively. First, we initialize $\theta^0(\xi_l(j))$ to a base profit, where base profit is the whole profit obtained by selecting arcs belonging to set $A_{\xi_l(j)}$. Second, we calculate the sub-gradient of $\xi_l(j)$, denoted by $\nabla L_{\lambda^k(\xi_l(j))}$, using equation (4.22). Third, we update the Lagrangian multiplier corresponding to $\xi_l(j)$ for the next iteration with equation (4.23). Finally, we update the step size for the next iteration by equation (4.24). After calculating $\lambda^{k+1}\left(\xi_l(j)\right)$, we distribute the updated price of cut $\xi_l(j)$ among all OD pairs corresponding to this cut based on their current price.

$$\nabla L_{\lambda^k(\xi_l(j))} = \{\sum_{v,(i,i',t,t',s,s')\in A_{\xi_l(j)}} x_{i,i',t,t',s,s',v,v} - 1\} \tag{4.22}$$

$$\lambda^{k+1}\left(\xi_l(j)\right) = \lambda^k(\xi_l(j)) + \theta^k\left(\xi_l(j)\right) \times \nabla L_{\lambda^k(\xi_l(j))} \tag{4.23}$$

$$\theta^{k+1}(\xi_l(j)) = \frac{\theta^0(\xi_l(j))}{k+1} \tag{4.24}$$

4.3.10. Dynamic System Performance Improvement Using Macroscopic Cumulative Flow Count Diagrams

Continuous-time approximation is an efficient technique for modeling complex logistics problems. Wang and Regan (2009) studied the convergence of a time window discretization method for the traveling salesman problem with time windows, which is initially introduced in Wang and Regan (2002) to show that iteratively refining the discretization converges to the optimal solution. Recently, Boland et al. (2017a) and (2017b) have suggested partially time-expanded networks which are iteratively refined until optimality is reached for continuous-time service network design problems and the traveling salesman problem with time windows, respectively.

Cumulative flow count diagrams are effective in describing the service process in the queueing system. The service process represents the required time and resources to serve a passenger. The performance of the queueing system is defined by the arrival process, service process, and queue discipline. We use time-dependent cumulative flow counts for each agent (i.e., passenger) to capture the arrival and departure of traveling objects. Fig. 4.8 illustrates the graphs corresponding to passengers' cumulative arrival, on-board, and departure count diagrams. The graph indicated by grey color is the passengers' cumulative arrival count to the system at time t, the graph indicated by blue color is the passengers' cumulative on-board count at time t, and the graph indicated by red color is the passengers' cumulative departure count from the system at time t. The region bounded by the passengers' cumulative arrival and on-board count diagrams represents the total waiting time for passengers to be picked up. In addition, the area contained by the passengers’ cumulative on-board and departure count diagrams represents the total service time.

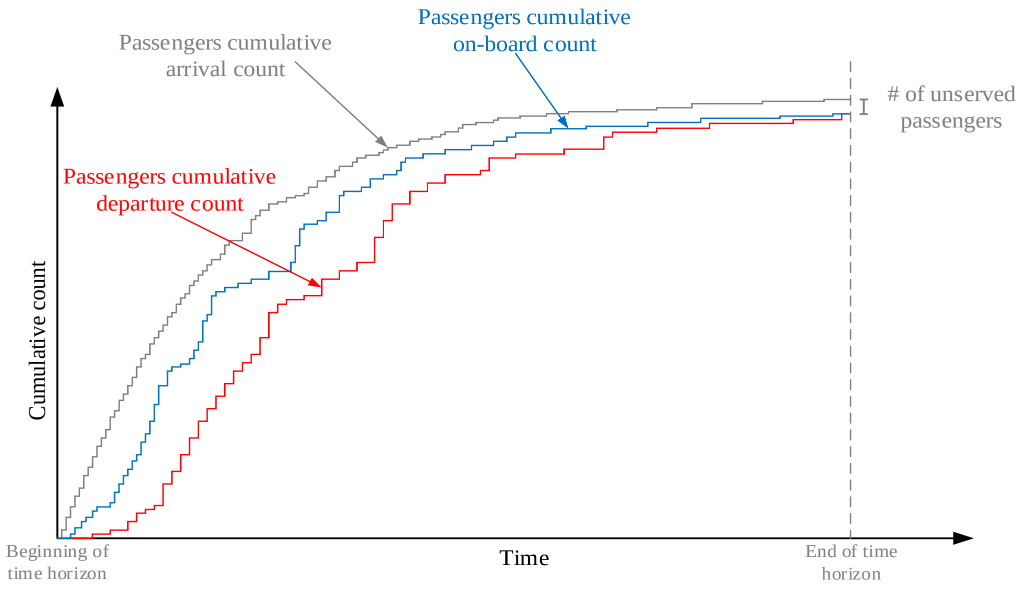


Fig. 4.8. Cumulative arrival, departure, and on board count diagrams.

Meng and Zhou (2014) used cumulative variables to describe the arrivals and departures of trains to allocate the time-space resources in rail timetabling optimization problems. Recently, Shi and Zhou (2015) have applied cumulative flow counts to optimize the system waiting time in the rail yard operation problem and describe classification tracks as a queueing system to capture the spatial capacity constraint. In this chapter, we use the cumulative flow count diagram to improve the efficiency of real vehicles. We define thresholds for passengers’ waiting and service times to measure the system performance. We check the performance of the system (passengers waiting and riding times) within each time horizon and identify time zones at which system performance is poor. Then, we increase the total number of hypothetical vehicles in those clusters operating in these time zones. By increasing the number of hypothetical vehicles in these clusters, we increase the chance of serving OD pairs corresponding to these clusters.

4.3.11. On the Computational Complexity of the DP Algorithm used for Each Cluster

Several efficient algorithms have been suggested to solve the time-dependent shortest path problem on a network with time-dependent arc costs (Ziliaskopoulos and Mahmassani 1993 and Chabini 1998 in deterministic networks; Miller-Hooks and Mahmassani 1998 and 2000 in stochastic networks). We have used a time-dependent state-dependent DP algorithm to solve the least-cost path problem obtained from Section 4.3.6.

Assume that the unit of time is one minute. Let $L_{i,t,s}(v)$ denote the label of vertex (i,t,s) in vehicle v's block, $TT_{i,i',t}$ denote the travel time of link (i,i') leaving from node i at time t, and the term "pred" stands for the predecessor. Algorithm 1, described below, presents the proposed time-dependent forward DP approach. The condition "$L_{i,t,s}(v) + c_{i,i',t,t',s,s'}(v) < L_{i',t',s'}(v')$" corresponds to the Bellman optimality condition.

```
// Algorithm 4.1: Time-dependent forward DP algorithm for each cluster $q$
  for each vehicle $v$, $v \in V^q$ do
  begin
    // initialization
    $L_{.,.,.}(.) := +\infty$;
    node pred of vertex $(.,.,.,.)$ := $-1$;
    time pred of vertex $(.,.,.,.)$ := $-1$;
    state pred of vertex $(.,.,.,.)$ := $-1$;
    vehicle pred of vertex $(.,.,.,.)$ := $-1$;
    for each time $t \in [t^q_{start}, t^q_{end}]$ do
    begin
```

```
      for each link $(i, i')$ do
      begin
        for each state $s$ do
        begin
          derive downstream state $s'$ based on the feasible state transition on link $(i, i')$
          derive arrival time $t' = t + TT_{i,i',t}$;
          if $(L_{i,t,s}(v) + c_{i,i',t,t',s,s'}(v) < L_{i',t',s'}(v'))$
          begin
            $L_{i',t',s'}(v') := L_{i,t,s}(v) + c_{i,i',t,t',s,s'}(v)$ ; // label update
            node pred of vertex $(v', i', t', s') := i$;
            time pred of vertex $(v', i', t', s') := t$;
            state pred of vertex $(v', i', t', s') := s$;
            vehicle pred of vertex $(v', i', t', s') := v$;
          end;
        end; // for each link
      end; // for each state
    end; // for each time
  end; // for each vehicle
```

Let $\mathcal{N}$ denote the set of nodes including both physical transportation and dummy nodes, $\mathcal{A}$ denote the set of links, and $\mathcal{T}$ denote the set of time stamps covering all vehicles' time horizons. According to our definition for cumulative service states, the total number of cumulative service states for n_q number of OD pairs in cluster q is 3^{n_q} because each OD pair's cumulative service state can be 0, 1, or 2. Therefore, the space complexity of our proposed DP algorithm for each cluster is $O(\alpha 3^{\alpha}|\mathcal{T}||\mathcal{A}|)$, where α is the maximum number of passengers solvable by the DP algorithm for one cluster. This statement can be interpreted as the maximum number of steps to be performed in the four for-loop structure, corresponding to the sequential loops for vehicles, time, service states, and links for each cluster.

Our experiments were performed on an Intel Workstation running two Xeon E5-2680 processors clocked at 2.80 GHz with 20 cores and 192GB RAM running Windows Server 2008 x64 Edition. Let's assume that $|\mathcal{T}| = 10^3$ and $|\mathcal{A}| = 10^3$. In the innermost loop of

Algorithm 4.1, five data are supposed to be recorded: $L_{i',t',s'}(v')$, node predecessor of vertex (v', i', t', s'), time predecessor of vertex (v', i', t', s'), state predecessor of vertex (v', i', t', s'), and vehicle predecessor of vertex (v', i', t', s'). As a result, $5\alpha 3^{\alpha} \times 10^{6}$ bytes of memory are required. In order to find the maximum value of α for any machine that runs our algorithm, it is sufficient to find the solution for this inequality: $5\alpha 3^{\alpha} \times 10^{6} \leq$ *total available memory*. For the machine we are running our experiments on, 192GB RAM is available; therefore, the "theoretical" maximum value of α is 7. Note that due to the existence of time and state dimensions in our model, the calculated α is not very large. In the following paragraph, we will explain that, in practice, the actual value of α solvable by our exact DP on our machine is larger than 7 (i.e., $\alpha \approx 35$).

In fact, not all 3^{α} OD pairs' cumulative service states are examined for each vehicle of a cluster, and the actual number is much smaller than this. We have shown this point for the example presented in Fig. 4.4. All hypothetical vehicles in cluster q are the same in terms of starting and ending depots as well as their work shift. As a result, according to Fig. 4.9, the least cost path traveled by vehicle v_1 from state [0,0] to [2,0] (blue route in block 1) is exactly the same as the least cost path traveled by vehicle v_2 from state [0,2] to [2,2] (blue route in block 2). Similarly, the least cost path traveled by vehicle v_1 from state [0,0] to [0,2] (red route in block 1) is exactly the same as the least cost path traveled by vehicle v_2 from state [2,0] to [2,2] (red route in block 2). To sum up, despite the fact that our proposed three-dimensional network is quite large, many of the shortest paths from one landmark node to another in latter blocks have already been evaluated in prior blocks.

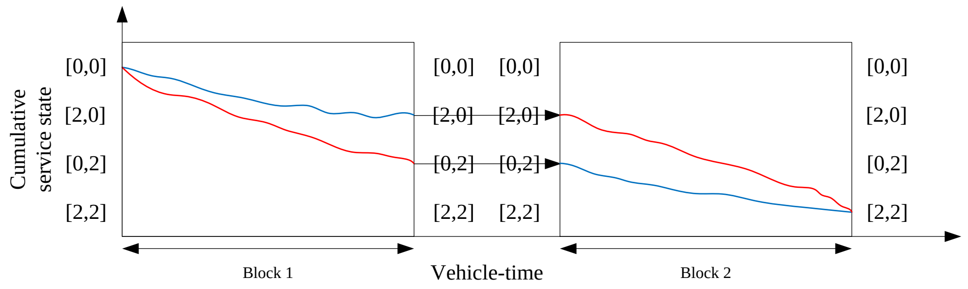


Fig. 4.9. Least cost paths from states [2,0] and [0,2] to state [2,2] in block 2 have already been calculated in block 1.

According to Algorithm 4.1, the hyper-network is created before DP implementation. This means that, before DP starts scanning vertexes, for each time t and each node i, all states corresponding to the OD pairs of cluster q must be generated. Sometimes the total number of feasible states for (i, t) is quite large, such that the size of the hyper-network becomes enormous. In this case, we suggest creating the network dynamically within the scanning process. Dynamically generating the hyper-network reduces the search space considerably. We also suggest using a beam search algorithm to keep a limited number of states for vertexes built from node i and time t at each level when the number of candidate states is large.

Basically, the beam search algorithm reduces the search space by setting dominance rules and solves practical problems within reasonable time and memory. The beam search algorithm is a greedy search algorithm which uses breadth-first search to explore the search tree. In addition, only a limited number of promising nodes (in our case, nodes are states) are kept at each level of the search tree, and other nodes are pruned off. In this algorithm, beam width refers to the number of promising nodes left at each level. In general, beam width restricts the memory required to perform the tree search. Note that fewer nodes are cut off with greater beam width. The beam search algorithm finally generates a representative set of non-dominated or promising solutions.

In this chapter, four criteria regarding the performance of a vehicle routing system are proposed as evaluation rules during the node selecting procedure to be used in the beam search process. Let $C^{p}_{Arrival}(t)$, $C^{p}_{On-board}(t)$, $C^{p}_{Departure}(t)$, $C^{v}_{Arrival}(t)$, and $C^{v}_{Departure}(t)$ denote passengers' cumulative arrival count to the system at time t, passengers' cumulative on-board count at time t, passengers' cumulative departure count from the system at time t, vehicles' cumulative arrival count to the system at time t, and vehicles' cumulative departure count from the system at time t, respectively. Then, our four criteria will be presented as follows:

(1) The total number of unserved passengers at time t should be minimized. This criteria can be calculated by $C^{p}_{Arrival}(t) - C^{p}_{Departure}(t)$.

(2) In order to improve service quality, the total waiting time of passengers receiving service until time t should be minimized. This criteria can be calculated by $\sum_{t'=t^{q}_{start}}^{t}[C^{p}_{Arrival}(t') - C^{p}_{On-board}(t')]$.

(3) The total service time of passengers until time t should also be minimized to improve passengers' service quality. This criteria can be calculated by $\sum_{t'=t^{q}_{start}}^{t}[C^{p}_{On-board}(t') - C^{p}_{Departure}(t')]$.

(4) The total processing time for vehicles until time t indicates the efficiency of the system to some degree. This criteria can be represented by $\sum_{t'=t^{q}_{start}}^{t}[C^{v}_{Arrival}(t') - C^{v}_{Departure}(t')]$.

In our experiments, two different approaches have been considered to define these dominance rules. The first approach is defining a utility function which is a weighted combination of all aforementioned criteria. Then, the top twenty states with the highest utility functions will be selected for further exploration, and other states will be pruned off. Note that we set twenty as our beam width after many trial and errors. In the second approach, we keep the top twenty states with the highest collected profit obtained from picking up OD pairs. In other words, states in which more OD pairs have been picked up have higher priority to be selected.

Here we provide an example to show vehicles' processing time as a measure of dynamic system performance. Assume a three-node transportation network in which two passengers with different origins and destinations have called for service (Fig. 4.10). Suppose passenger 1 wants to go from node 1 to node 3, while passenger 2 wants to go from node 3 to node 1. Vehicle v_1 located at node 1 and vehicle v_2 located at node 3 are available to deliver passengers. Note that each vehicle must return to its origin depot after finishing its tasks. In this example, each passenger can be served by two different ways: (1) by a non-stop trip, (2) by a one-stop trip with connection at node 2. Therefore, six OD pairs may be defined for this problem:

$OD^1_{w_1(1)}, OD^1_{w_1(2)}, OD^1_{w_2(1)}, OD^2_{w_2(1)}, OD^1_{w_2(2)}, OD^2_{w_2(2)}$. We have shown these six OD pairs by a, b, c, d, e, and f in Figure 4.10.

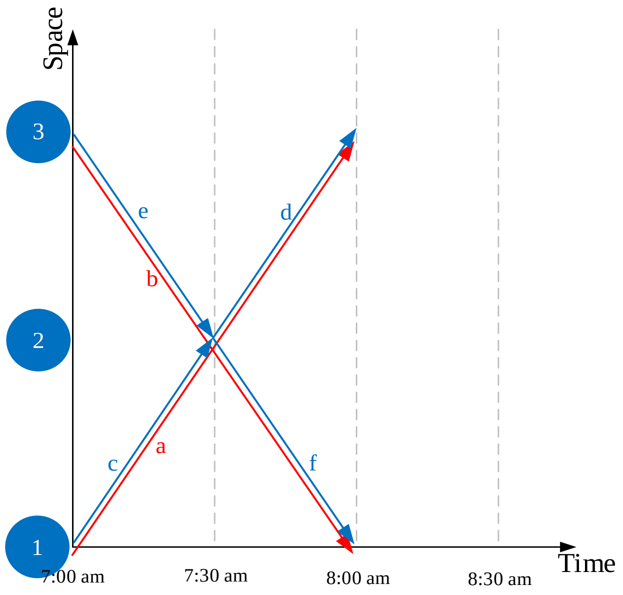


Fig. 4.10. Six OD pairs corresponding to the non-stop and one-stop trips for two passengers.

Dummy nodes corresponding to passengers' origins and destinations, as well as vehicles' origin and destination depots are added to the physical transportation network (Fig. 4.11). Suppose all six OD pairs have been clustered in one group.

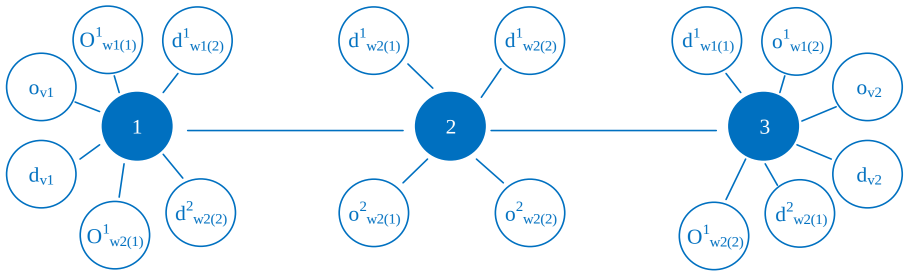


Fig. 4.11. Three-node transportation network with added dummy nodes.

Figure 4.12 shows the space-vehicle-time path by which each passenger is served by a non-stop trip. This path is as follows: $o_{v_1} \to 1 \to o^1_{w_1(1)} \to 1 \to 2 \to 3 \to d^1_{w_1(1)} \to 3 \to 2 \to 1 \to d_{v_1} \to o_{v_2} \to 3 \to o^1_{w_1(2)} \to 3 \to 2 \to 1 \to d^1_{w_1(2)} \to 1 \to 2 \to 3 \to d_{v_2}$. The state corresponding to the vertex at which the state transition occurs has been shown in

Figure 4.12 as well. Note that the state written in this graph is presented by a sequence of numbers, each representative of the cumulative service state for OD pairs a, b, c, d, e, and f. If passengers are served by non-stop trips, deadheading for vehicles occurs. In other words, each vehicle must travel 1 hour without carrying any passengers to return to its depot. In this case, the total processing time for each vehicle is equal to 2 hours.

Fig. 4.13 shows the space-vehicle-time path by which each passenger is served by a one-stop trip. This path is $o_{v_1} \rightarrow 1 \rightarrow o^1_{w_2(1)} \rightarrow 1 \rightarrow 2 \rightarrow d^1_{w_2(1)} \rightarrow 2 \rightarrow o^2_{w_2(2)} \rightarrow 2 \rightarrow 1 \rightarrow d^2_{w_2(2)} \rightarrow 1 \rightarrow d_{v_1} \rightarrow o_{v_2} \rightarrow 3 \rightarrow o^1_{w_2(2)} \rightarrow 3 \rightarrow 2 \rightarrow d^1_{w_2(2)} \rightarrow 2 \rightarrow o^2_{w_2(1)} \rightarrow 2 \rightarrow 3 \rightarrow d^2_{w_2(1)} \rightarrow 3 \rightarrow d_{v_2}$. If passengers are delivered by one-stop trips, deadheading for vehicles does not occur. In this case, the total processing time for each vehicle is equal to 1 hour.

We have used this example to show how the connection between microscopic cumulative service states and the macroscopic cumulative task count can help us to measure the processing time of each vehicle and subsequently the performance of the whole system. This evaluation rule is checked during the node selecting procedure in the beam search process.

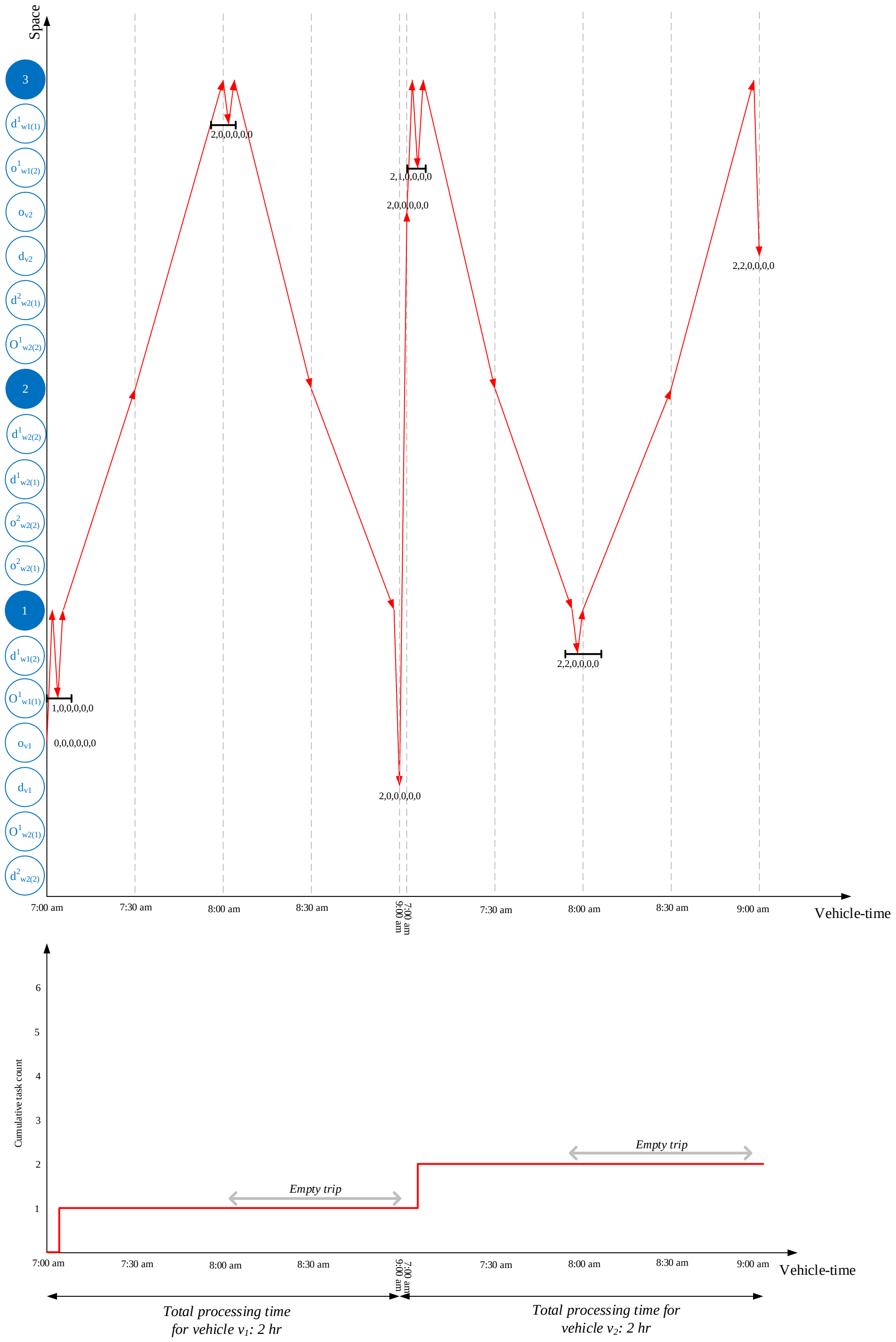


Fig. 4.12. The microscopic cumulative service state representation for pickup and delivery without transfer in Fig. 4.11 is connected to the corresponding macroscopic cumulative task count graph to measure the dynamic system performance.

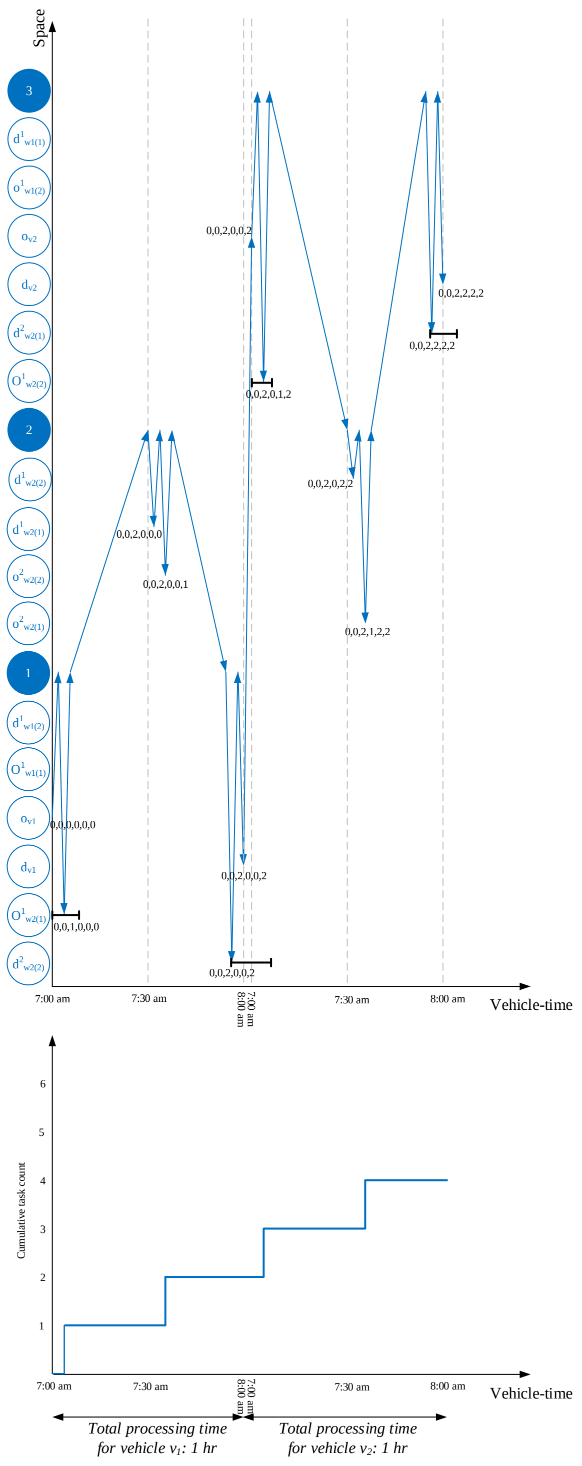


Fig. 4.13. The microscopic cumulative service state representation for pickup and delivery with transfer in Fig. 4.11 is connected to the corresponding macroscopic cumulative task count graph to measure the dynamic system performance.

4.4. Computational Experiments

The time-dependent state-dependent DP described in this chapter was coded in C++ platforms, while OD pairs' clustering and work pieces routing were solved by GAMS Distribution 23.00. The experiments were performed on an Intel Workstation running two Xeon E5-2680 processors clocked at 2.80 GHz with 20 cores and 192GB RAM running Windows Server 2008 x64 Edition. In this section, we initially explain the implementation of our model on a three-node transportation network. Then, we examine our proposed model on instances applied by Ropke and Pisinger (2006) which is publicly available at http://www.diku.dk/~sropke/, followed by the data provided by Cainiao Network available at https://tianchi.shuju.aliyun.com/datalab/index.htm to demonstrate the computational efficiency of our developed algorithm. The complete C++ implementation of the proposed DP algorithm and data set for the three-node example are available at https://github.com/xzhou99/Agent-Plus.

4.4.1. Three-node Transportation Network

The notations used for the implementation of our model for this example are first introduced in Table 4.2. Our proposed hyper-networks integrate physical transportation networks and service networks together. The nodes corresponding to the vertexes of a hyper-network can be categorized into three types: (1) passenger nodes (service nodes), (2) vehicle depots, and (3) transportation nodes. In the example provided below, service nodes, vehicle depots, and transportation nodes have been numbered from 10 to 999, from 1000 to 1999, and from 2000 to 2999, respectively. The links of hyper-networks include physical transportation links and service links representing service actions. In Table 4.2, four types of service actions have been defined as follows: PO, DD, PT and DT.

Table 4.2
Notations used for the implementation of our model for the three-node transportation network example.

Symbol	Definition
w, w'	Arrays that record the cumulative service state of passengers
p	Passenger's index
$w[p]$	An element of array w indicating passenger p's cumulative service state
v	Vehicle's index
T	Length of time horizon
t, t'	Time interval indexes
l	Link's index
$TT[l]$	Travel time of link l
PO	A type of service link that represents pickup action from origin
DD	A type of service link that represents drop-off action at destination
PT	A type of service link that represents pickup action from a transfer point
DT	A type of service link that represents drop-off action at transfer point
$from\_state\_code$	Cumulative service state code at the tail node of a link
to_state_code	Cumulative service state code at the head node of a link
$Pflag$	Used to distinguish dummy nodes corresponding passengers' origin/destination from physical nodes
$Vflag$	Used to distinguish dummy nodes corresponding vehicles' depots from physical nodes
$time\_window\_be$	Beginning time of the time window at a node
$time\_window\_en$	Ending time of the time window at a node
$cutoff\_time$	Cutoff time at a node
$location\_code$	Location code of the node, ranges from 0 to 9

When a service action occurs, the existing cumulative service state w will change to a new service state w'. All w values will be recorded in state set W. In addition, the value of the location code for passenger nodes ranges from 0 to 9, where "0" represents passenger's origin node, "1" means that the passenger is being transported by a vehicle, and "9" indicates the passenger's destination node. "2", "3", "4", "5", "6", "7" and "8" represent the passenger's possible transfer locations. Service state transitions are defined in Table 4.3.

Table 4.3
Service state transition codes and their descriptions.

$from\_state\_code$	to_state_code	Service link type
0	1	Pickup from origin (PO)
1	2 3 4 5 6 7 8	Drop-off at a transfer point (DT)
1	9	Drop off at the destination (DD)
2 3 4 5 6 7 8	1	Pickup from a transfer location (PT)

A passenger must be picked up within his particular departure time window and dropped off within his specific arrival time window. In addition, each vehicle has particular time windows for the departure from its starting depot and the arrival at its ending depot. We also define the "$cutoff\_time$" at a passenger's transfer location, which indicates the latest time at which the passenger can be picked up from that particular transfer point.

Physical transportation nodes and links of the three-node transportation network have been shown in Figure 4.14(a). Note that in this figure, the term "TN" stands for transportation node.

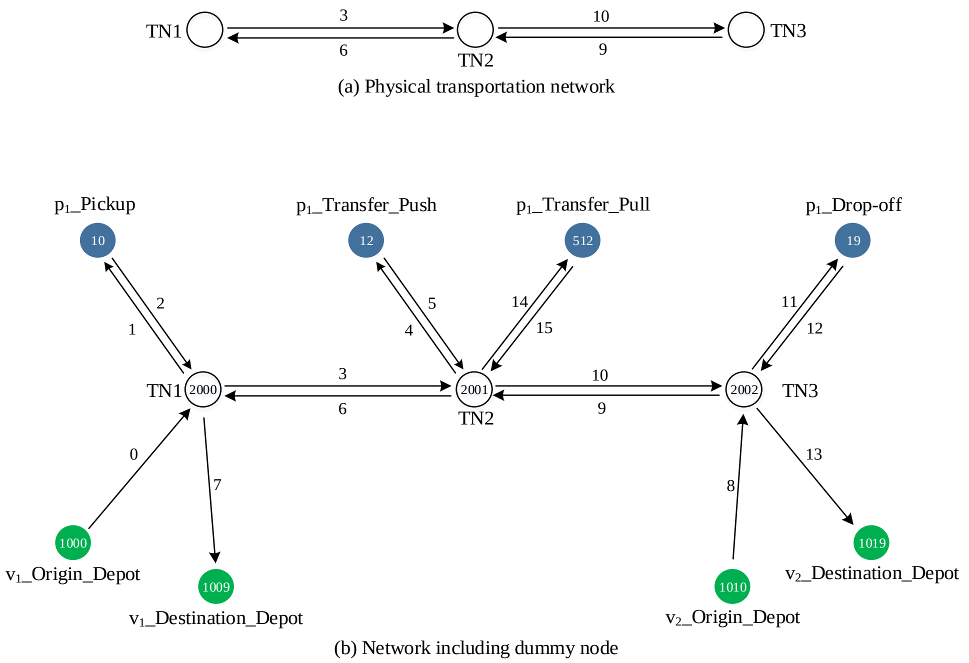


Fig. 4.14. (a) Three-node transportation network; (b) Network containing dummy nodes.

In this example, passenger p_1 wants to travel from node TN1 to node TN3 and needs to be transferred at node TN2. It is supposed that vehicle v_1 transports him from TN1 to TN2, while v_2 is responsible for transporting the passenger to their final destination (i.e., node TN3).

In order to capture service actions in this example clearly, solid service nodes are added to the physical transportation network to illustrate the concept of a transfer through "push" and "pull" actions. These nodes are connected to the corresponding physical nodes by service links. Nodes and links are identified by an ID number based on their types. According to Figure 4.14(b), service nodes/passenger nodes are numbered from 10 to 999, vehicle depots are numbered from 1000 to 1999, and transportation nodes are numbered from 2000 to 2999. Node 10 represents the pick-up location and node 19 represents the drop-off spot, while nodes 12 and 512 indicate the transfer location where the passenger gets out of vehicle v_1 and gets into vehicle v_2, respectively. That is why we say that vehicle v_1 "pushes" or drops off passenger p_1 at node 12 and v_2 "pulls" or picks him up from node 512.

Note that nodes 12 and 512 are the service nodes, at which the passenger transfers to another vehicle; therefore nodes 12 and 512 are actually located at the same location. However, in order to update the service state of passenger p_1, we must split the transfer location into two nodes such that pickup and drop-off actions will be recognizable. Meanwhile, the vehicle's depot must be split into two nodes, although the starting and ending depots of a vehicle are located in the same spot. By doing this, the vehicles' departure and arrival time windows at their starting and ending depots can be imposed independently. The input data corresponding to different nodes of the network demonstrated in Figure 4.14(b) is given in Table 4.4.

Table 4.4
The input data corresponding to different nodes of the network demonstrated in Fig. 4.14(b).

$Node\_No$	$Pflag$	$Vflag$	$time\_window\_begin$	$time\_window\_end$	$cutoff\_time$	$location\_code$
10	1	-1	10	20	20	0
12	1	-1	0	35	35	2
512	1	-1	35	10000	35	2
19	1	-1	0	80	80	9
1000	-1	1	0	10000	10000	0
1009	-1	1	45	50	50	9
1010	-1	2	50	10000	10000	0
1019	-1	2	85	90	90	9
2000	-1	-1	0	10000	10000	-1
2001	-1	-1	0	10000	10000	-1
2002	-1	-1	0	10000	10000	-1

According to Table 4.4, the $Pflag$ of a node is equal to the passenger ID (i.e. 1, 2, 3, 4, etc.), if and only if the node is either the corresponding passenger's origin/destination or pickup/drop-off locations at a transfer point. Otherwise, $Pflag$ is equal to -1. The

$Vflag$ of a node is also equal to a vehicle ID (i.e. 1, 2, 3, 4, etc.), if and only if the node is either the corresponding vehicle's origin depot or destination depot. Otherwise, $Vflag$ is equal to -1.

$time\_window\_begin$ and $time\_window\_end$ denote the starting and ending time of a node's time window. Note that if $time\_window\_end$ equals 10000, then we only need to make sure that the arrival time of a passenger/vehicle at that node is greater than or equal to the time corresponding to $time\_window\_begin$. $cutoff\_time$ has been defined to ensure that the passenger does not arrive late at the transfer node. Note that this parameter is only valid for service nodes.

$location\_code$ denotes the service status of the passenger/vehicle at the corresponding node. For example, $location\_code$ at node 12 is equal to 2, which means that the passenger has been dropped off at node 12 to be transferred to another vehicle. The input data corresponding to different links of the network demonstrated in Figure 4.14(b) is given in Table 4.5.

Links are numbered from 0, and their direction can be defined from $FromNodeNo$ to $ToNodeNo$. $LinkServiceType$ is not used in the program, but it describes changes in the service state. According to Fig. 4.14(b), the type of link 2 (i.e., pointing from node 10 to node 2000) is "pickup," so $FromStateCode$ and $ToStateCode$ equal to 0 and 1, respectively. This means that the passenger gets into the vehicle at node 10. Similarly, the type of link 4 is "push" (dropping off the passenger at the transfer point); therefore, $FromStateCode$ and $ToStateCode$ corresponding to this link are 1 and 2, respectively. One can conclude that the values of $FromStateCode$ and $ToStateCode$ are determined by the corresponding $LinkServiceType$. In addition, if $LinkServiceType$ of a link is null, then both the $FromStateCode$ and $ToStateCode$ of the link are equal to -1. Furthermore, $LinkServiceType$ is converted from a string type parameter to an integer by $LinkServiceCode$. Finally, $LinkLength$ and $SpeedLimit$ are given as the length and speed limit of a link, by which we are able to calculate the minimum time needed to move through the corresponding link in minutes.

Table 4.5
The input data corresponding to different links of the network demonstrated in Fig. 4.14(b).

LinkNo.	From Node No	ToNode No	LinkService Type	LinkService Code	LinkService PaxCode	FromState Code	ToState Code	LinkLength	Speed Limit
0	1000	2000	Null	0	0	-1	-1	1	12
1	2000	10	Null	0	0	-1	-1	1	6
2	10	2000	Pickup	1	1	0	1	1	6
3	2000	2001	Null	0	0	-1	-1	1	12
4	2001	12	Push	4	1	1	2	1	12
5	12	2001	Null	0	0	-1	-1	1	12
6	2001	2000	Null	0	0	-1	-1	1	12
7	2000	1009	Null	0	0	-1	-1	1	12
8	1010	2002	Null	0	0	-1	-1	1	12
9	2002	2001	Null	0	0	-1	-1	1	12
10	2001	512	Null	0	0	-1	-1	1	12
11	512	2001	Pull	3	1	2	1	1	12
12	2001	2002	Null	0	0	-1	-1	1	12
13	2002	19	drop-off	2	1	1	9	1	12
14	19	2002	Null	0	0	-1	-1	1	12
15	2002	1019	Null	0	0	-1	-1	1	12

4.4.2. Ropke and Pisinger (2006) Standard Data Set

The Ropke and Pisinger (2006) data set is the modified version of instances initially introduced by Savelsbergh and Sol (1998). In this data set, passenger p's origin and destination are denoted by node p and node $n + p$, respectively, where n is the total number of passengers in the system. In addition, the coordinates (x and y) of passengers' origin and destination are randomly generated and uniformly distributed over a $[0,50] \times$

[0,50] square. A single depot is located in [25,25]. The load of each passenger is randomly generated from $[5, Cap_v]$, where Cap_v is the maximum capacity of the vehicles (in these instances the vehicles' capacity is assumed to be the same). Moreover, [0,1000] is considered as the vehicles' time horizon (vehicles' time horizon is assumed to be identical). Feasible departure/arrival time windows are also randomly generated for each passenger. Fig. 4.15 illustrates the position of passengers' origin and destination and the routes of vehicles v_1-v_4 in data set AA30.

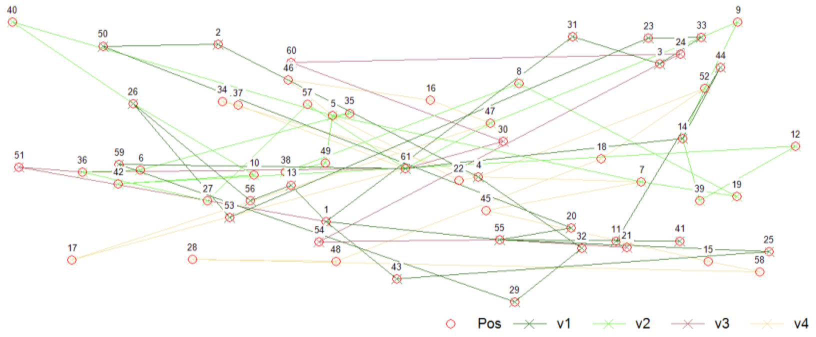


Fig. 4.15. Position of passengers' origin and destination and route of vehicles v_1-v_4 in data set AA30.

Six groups of instances are examined by considering different values of vehicles' capacity, different length of departure/arrival time windows, and different passengers' load. The values of vehicles' capacity in instances AA, BB, CC, and DD are 15, 20, 15, and 20, respectively. The lengths of passengers' departure/arrival time windows are 60, 60, 120, and 120, respectively. In addition, as mentioned before, in these four instances, the load of each passenger is randomly generated from $[5, Cap_v]$. In instances XX and YY, the value of vehicles' capacity is 15, while the length of passengers' departure/arrival time windows are 60 and 120, respectively. In addition, the load of each passenger is assumed to be 1. In instances XX and YY, due to the large value of vehicles' capacity (i.e. 15) in comparison to the load of each passenger (i.e. 1), more levels of complexity are expected.

To the best of our knowledge, very few papers have published the results of instances XX and YY. Table 4.6 presents the results obtained from running our algorithm on Ropke and Pisinger (2006) instances. The two alphabetical letters in the instance names are representative of vehicles' capacity, length of passengers' time windows, and load of passengers, while the double-digit number after the alphabetical letters demonstrates the total number of passengers in that data set. According to Table 4.6, in most instances, our heuristic-based algorithm performs better than the heuristic proposed by Ropke and Pisinger (2006) in terms of the number of vehicles (as the primary objective) and routing cost (as the secondary objective). However, from a computation time perspective, it seems that the heuristic by Ropke and Pisinger (2006) performs slightly better than ours.

Table 4.6
Results obtained from running our algorithm on Ropke and Pisinger (2006) instances.

Name	# of groups	# of Vehicles	# of Vehicles (Ropke and Pisinger (2006))	Routing cost	UB of routing cost (Ropke and Pisinger (2006))	Computation time (sec)	Computation time (sec) (Ropke and Pisinger (2006))
AA30	5	4	5	41,316	51,317.40	25.6	27
AA35	5	5	5	51,506	51,343.53	44.2	33
AA40	6	6	6	61,759	61,609.44	44.67	41.4
AA45	7	6	6	62,029	61,693.01	52.57	49.8
AA50	8	6	7	62,213	71,932.03	54.25	58.8
AA55	8	8	8	82,405	82,185.31	95.63	64.2
AA60	9	8	9	82,629	92,366.70	94.44	76.8
AA65	10	7	8	72,783	82,331.12	98.3	87.6
AA70	10	8	11	82,950	112,458.28	143.7	98.4
AA75	11	8	9	83,044	92,529.42	141.82	112.8
BB30	5	4	5	41,267	51,193.62	28.4	28.2
BB35	5	5	6	51,580	61,400.07	38.2	32.4
BB40	6	5	5	51,785	51,421.35	60.83	44.4
BB45	7	6	6	61,950	61,787.28	71.86	49.2
BB50	8	7	7	72,164	71,889.75	89.5	58.8
BB55	8	8	8	82,483	82,080.73	122.75	64.2
BB60	9	11	10	112,988	102,323.77	122.22	73.8
BB65	10	10	8	103,211	82,623.98	122.9	85.2
BB70	10	11	9	113,534	92,647.75	160.4	100.8
BB75	11	11	9	113,558	92,476.30	154.18	112.8
CC30	5	5	5	51,358	51,145.18	44.2	28.2
CC35	5	5	5	51,578	51,235.64	51.8	34.2
CC40	6	5	6	51,695	61,473.91	49.5	43.2
CC45	7	5	8	51,955	81,408.89	53.57	49.8
CC50	8	7	6	72,154	61,936.27	51.38	63.6
CC55	8	10	6	102,460	61,930.55	90.88	71.4
CC60	9	8	7	82,546	72,104.00	84.89	82.8
CC65	10	9	8	92,803	82,326.62	102.9	90
CC70	10	9	9	92,963	92,613.68	149.3	102
CC75	11	9	9	93,220	92,711.74	149	112.8
DD30	5	4	6	41,426	61,040.10	41.4	27.6
DD35	5	5	7	51,614	71,308.04	68.2	33.6
DD40	6	5	6	51,851	61,531.68	68	43.2
DD45	7	5	8	51,960	81,601.63	72.86	48
DD50	8	6	7	62,131	71,761.23	70.25	60
DD55	8	6	7	62,358	72,051.95	121.38	69
DD60	9	7	8	72,521	82,308.08	113.78	78.6
DD65	10	7	8	72,825	82,200.77	120.6	90
DD70	10	8	8	83,034	82,631.56	173.6	102
DD75	11	9	9	93,255	92,970.84	165.45	109.8
XX30	5	4	-	41,093	-	101.4	-
XX35	5	5	-	51,313	-	174.6	-
XX40	6	5	-	51,540	-	166.33	-
XX45	7	7	-	71,719	-	156.29	-
XX50	8	6	-	61,707	-	126.88	-
XX55	8	6	-	61,839	-	254.25	-
XX60	9	6	-	62,033	-	299.44	-
XX65	10	6	-	62,531	-	286.4	-
XX70	10	7	-	72,775	-	400.1	-
XX75	11	8	-	82,960	-	388.73	-
YY30	5	4	-	41,195	-	76.2	-
YY35	5	5	-	51,363	-	187.6	-
YY40	6	6	-	61,608	-	161.33	-
YY45	7	6	-	61,806	-	176.14	-
YY50	8	6	-	61,966	-	203.88	-
YY55	8	7	-	72,121	-	274.13	-
YY60	9	6	-	62,321	-	276.56	-
YY65	10	7	-	72,464	-	327.3	-
YY70	10	7	-	72,586	-	419.6	-
YY75	11	9	-	92,679	-	397.64	-

Basically, in most heuristics used for solving vehicle routing problems, a set of initial solutions are constructed and then improved by intensifying the search in the neighborhood of each solution. In large neighborhood search heuristics, instead of using small “standard moves”, i.e., moving a request from one route to another or exchanging two requests, we use very large moves that potentially can rearrange up to 30%-40% of all requests in a single iteration. The consequence of doing this is that the computation time needed for performing and evaluating the moves becomes much larger compared to the smaller moves. The number of solutions evaluated by the proposed heuristic per time unit is only a fraction of the solutions that could be evaluated by a standard heuristic. Nevertheless, very good performance is observed in the computational tests, since this method prevents getting stuck on a local optimum.

Our approach, on the other hand, clusters OD pairs based on their space-time closeness, finds the near-*optimal* route inside each cluster (constructs near-optimal routes for pieces of a path), and then finds the *optimal* chains of pieces that can be performed by vehicles outside the clusters. In fact, two out of three phases of our algorithm provide near-optimal solutions. In addition, at the beginning of each iteration, by changing the values of β_1 as the weight of space affinity, β_2 as the weight of time closeness, ζ_1 as the weight of spatio-temporal dissimilarities, and ζ_2 as the weight of total number of clusters, we may generate new clusters which can be slightly or very different from the previous ones. In other words, both standard and ALNS heuristics may occur in the first phase of our algorithm.

4.4.3. Large-scale Network and Test Data Set from Cainiao's Last Mile Rush Competition

We also tested our algorithm on the data set proposed by Cainiao Network, also called Last_Mile_Rush, to address the last mile delivery problem. Cainiao Network is the logistics company launched by Chinese e-commerce giant Alibaba Group in 2013. In this data set, there are two types of packages: e-commerce packages and intra-city online-to-offline (O2O) packages. In the case of e-commerce packages, couriers pick up packages from local branches of express companies and deliver them to individual passengers. In the case of intra-city O2O packages, couriers pick up packages from O2O shops and deliver them to the passengers, each of whom has particular pickup and delivery time windows. Several assumptions have been considered in this data set and are presented as follows:

1. One express company with 124 local branches serves all e-commerce delivery requests in Shanghai. Note that there is no overlapping between the service ranges of any two local branches.
2. All e-commerce packages arrive at local branches before 8:00am and must be delivered to individual passengers before 8:00pm.
3. All couriers start working at 8:00am. They are allowed to deliver both types of packages at the same time.
4. Type "A" Demand: There are 9,214 points of interest (POIs), each of which has a particular latitude and longitude. Every POI is representative of the location of an

order/passenger, which can be either an e-commerce or intra-city O2O order, and may contain one or more than one package.

5. Each POI is served by only one local branch since there is no overlapping between the service ranges of any two local branches.

6. No courier is allowed to carry more than 140 packages at a time.

7. If a passenger requests multiple e-commerce packages, all should be delivered to the corresponding POI at one time.

8. Type "B" Demand: There are 598 O2O shops in Shanghai. Similar to e-commerce packages, if a passenger requests multiple O2O packages, all should be delivered to the corresponding POI at one time.

9. Each O2O order has particular pickup and delivery time windows. Couriers must pick up O2O orders from O2O shops at the designated pickup time and deliver them to the O2O passengers no later than the designated delivery time.

10. If a courier arrives at a delivery point of an O2O order earlier than its designated delivery time, he does not need to wait and is allowed to leave the package(s) there.

11. The data set contains the latitude (lat) and longitude (lng) of all local branches, POIs, and O2O shops. The distance between any two locations can be calculated by equation (4.25).

$$distance = 2R.arcsin(\sqrt{sin^2\left(\frac{\pi}{180}\Delta lat\right) + \cos\left(\frac{\pi}{180}lat_A\right).\cos\left(\frac{\pi}{180}lat_B\right).sin^2(\frac{\pi}{180}\Delta lng)} \qquad (4.25)$$

, where (lat_A, lng_A) and (lat_B, lng_B) are the latitude and longitude of nodes A and B, respectively; $\Delta lat = \frac{lat_A - lat_B}{2}$; $\Delta lng = \frac{lng_A - lng_B}{2}$; and $R = 6{,}378{,}137\ m$.

12. The traveling speed of all couriers is assumed to be 15 km per hour. There are 1,000 available couriers in this data set. Each courier may remain idle or serve in one or more than one local branch.

13. Couriers start working exactly at 8:00am from local branches and end at the last POI.

14. The process time of a POI's request is also calculated by equation (4.26).

$$T = 3\sqrt{n} + 5 \tag{4.26}$$

, where n is the number of packages in the POI's order.

This data set contains several attributes related to the location of local branches, POIs, and O2O shops, as well as information related to the e-commerce and intra-city O2O orders, all of which are presented in Table 4.7.

Table 4.7
Given information for the location of local branches, POIs, O2O shops, e-commerce, intra-city O2O orders, and couriers.

Information related to the location of local branches	
Information	Description
site_id	Local branch code (e.g. A001)
Lng	Longitude of a local branch
Lat	Latitude of a local branch
Information related to the location of POIS	
Information	Description
spot_id	POI id (e.g. B0001)
Lng	Longitude of a POI
Lat	Latitude of a POI
Information related to the location of O2O shops	
Information	Description
shop_id	O2O shop id (e.g. S001)
Lng	Longitude of an O2O shop
lat	Latitude of an O2O shop
Information related to the e-commerce packages	
Information	Description
order_id	e-commerce order id
spot_id	POI id

site_id	Local branch id
Num	Number of packages in the order to be delivered from the given local branch to the given POI
Information related to the intra-city O2O packages	
Information	Description
order_id	Intra city O2O order id
spot_id	POI id
shop_id	O2O shop id
pickup_time	Pickup time at an O2O shop
delivery_time	Delivery time to an O2O passenger
Num	Number of packages in the order to be delivered from the given O2O shop to the given POI
Information related to the couriers	
Information	Description
courier_id	Courier id (e.g. D0001)

The total number of e-commerce orders (type “A” tasks) and O2O orders (type “B” tasks) is 9,214 and 598, respectively, while at most 1000 vehicles are available to perform the tasks. There is always a trade-off between minimizing the total cost (total travel time) and maximizing the total number of tasks (both types) that are performed. In order to find a balance between these two different objectives, we offer an evaluation function containing two terms: (1) total travel time (TT) with the weight of 1; and (2) the total number of tasks that are performed with the weight of α. Parameter α is tuned by running the algorithm for different values of α. In addition, in order to guide the search, we can assign different weights/priorities for type “A” or type ”B” tasks that haven been picked up and delivered successfully, as well as type “A” or type “B” tasks that have been only picked up but not delivered. Therefore, the evaluation function can be defined as follows:

$$Min\,\{TT + \alpha\big(\beta_A^{P\&D} n_A^{P\&D} + \beta_B^{P\&D} n_B^{P\&D} + \beta_A^{P} n_A^{P} + \beta_B^{P} n_B^{P}\big)\} \tag{4.27}$$

, where TT is the total travel time (total travel cost); α is the weight of total number of tasks that are performed (complete pickup and delivery or just pickup); $\beta_A^{P\&D}$ is the weight/priority of type “A” tasks that haven been picked up and delivered successfully; $n_A^{P\&D}$ is the total number of type “A” tasks that haven been picked up and delivered successfully; $\beta_B^{P\&D}$ is the weight/priority of type “B” tasks that haven been picked up and delivered successfully; $n_B^{P\&D}$ is the total number of type “B” tasks that haven been picked up and delivered successfully; β_A^{P} is the weight/priority of type “A” tasks that haven been only picked up but not delivered; n_A^{P} is the total number of type “A” tasks that haven been picked up but not delivered; β_B^{P} is the weight/priority of type “B” tasks that haven been only picked up but not delivered; and n_B^{P} is the total number of type “B” tasks that haven been picked up but not delivered.

The above evaluation function can be utilized as a selection criterion for the set of states. Note that the results are very sensitive to parameters $\beta_A^{P\&D}$, $\beta_B^{P\&D}$, β_A^{P}, and β_B^{P}, whose values are varied by the distribution of OD demands and the characteristics of types “A” and “B” tasks. Therefore, the values of these parameters must be determined by a large number of experiments for each group of tasks.

We have selected group 31 (out of all 111 groups of tasks) to explain how we have adjusted these parameters for a group of tasks. Our attempts for tuning the parameters for group 31 are presented in Appendix C. Note that we performed the same procedure for all 111 groups of tasks. We also have used the Fibonacci sequence (max number of states) as the width of the beam search algorithm to reduce the search region.

All local branches, POIs, O2O shops, and OD demands in the Cainiao Network have been shown in Fig. 4.16(a). As shown in Table 4.8, we have used five different parameters for the beam width to implement our beam search algorithm. According to Table 4.8, it can be concluded that case IV provides the most favorable solution among other cases, since in this case all tasks (both types “A” and “B” tasks) have been performed by 893 vehicles in 3,764.51 seconds. Comparing the results of cases I and II to case IV, types “A” and “B” tasks are incomplete (IC) in the prior two cases. Comparing the results of case III to case IV, the total cost and the total number of vehicles needed in this case is much larger than those in case IV. Finally, comparing the results from case V to case IV, the random access memory (RAM) usage and CPU time taken for case V are roughly twice those used for case IV, while the total number of vehicles needed and total cost for both cases are almost the same. Therefore, we have selected case IV and shown the delivery routes of all couriers in Fig. 4.16(b).

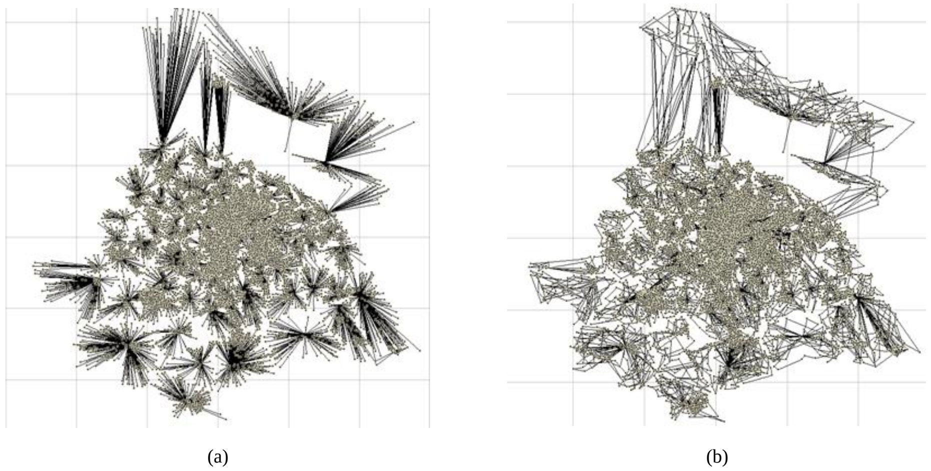


Fig. 4.16. (a) Cainiao network with all local branches, POIs, O2O Shops, and OD demands; (b) Delivery routes of all Couriers for Case IV in Table 4.8.

Table 4.8
Results obtained from running our algorithm on the Cainiao network.

Case No.	Beam width	RAM usage (GB)	CPU time (sec)	Total cost (min)	Total # of tasks completed (A+B)	# of vehicles
I	6,765	19.84	431.78	459337	6908+433 (IC)	755
II	10,946	48.09	888.68	389440	8936+497 (IC)	876
III	17,711	84.93	1829.06	364270	9214+598	954
IV	**28,657**	**140.96**	**3764.51**	**297200**	**9214+598**	**893**
V	46,368	220.23	7748.42	296934	9214+598	892

4.5. Conclusions

In this research, by extending the work pioneered by Bellman (1962), Held and Karp (1962) and Psaraftis (1980) on using the dynamic programing method to solve the TSP and VRP, we embed many complex constraints of the pickup and delivery problem with transfers on a three-dimensional SST network. In this hyper-network construction process, elements of time and load are explicitly added as new dimensions to the physical transportation network. To address the issue of the curse of dimensionality, we demonstrate a consistent transition from the microscopic cumulative service states to macroscopic cumulative flow count diagrams, which can be used to effectively estimate the overall dynamic system performance and guide the search. We also split the large-sized primary VRPPDTW into a number of small-sized sub-problems in which OD pairs with the most compatibility are clustered together. We use a time-dependent, state-dependent forward DP algorithm to solve the time-dependent state-dependent least-cost assignment-path problem for the local clusters derived from the original PDPT. At the end, extensive computational results over the standard instances used by Ropke and Pisinger (2006) and real-world large scale data set proposed by Cainiao Network with about 10,000 delivery orders were performed to examine the effectiveness and computational efficiency of our developed algorithm.

Future work may concentrate on building a computational engine to establish a wrapper for the DP algorithms with the inputs of a transportation network and possible state transition matrixes, and the output of various vehicle-path assignment and routing solutions. Another interesting extension of our SST framework can be building a more practically useful and robust model with some levels of travelers/carriers' behavior in better passengers' and vehicles' clustering, as opposed to simple and efficient trade-offs between time and distance.

CHAPTER 5

ACCESSIBILITY WITH TIME AND RESOURCE CONSTRAINTS: COMPUTING HYPER-PRISMS FOR SUSTAINABLE TRANSPORTATION PLANNING

5.1. Introduction

Accessibility is a theoretical and analytical concept in transportation, urban, and social sciences. Accessibility, or the freedom to participate in activities, obtain resources, and interact with others in a given environment, is a fundamental reason why cities and transportation exist (Bartholomew 2009). As a result, accessibility measures and models have become core components of many advanced transportation and urban models, and a common basis for evaluating the performance of transportation systems, policies, plans and projects (Handy 2005).

A common accessibility measure is the *space-time prism* (STP). A STP is a measure of potential mobility for an individual: it is the envelope of all possible travel paths between two locations in geo-space and time, considering (1) the given maximum travel speed on each direction, and (2) the time budget which is defined by (a) the times when the individual is required to be (or observed) at both locations, and (b) any stationary time required for activity participation during the time interval. The STP is an elegant and sensitive accessibility measure: it captures the interface among the locations and timing of activities in an environment, individual differences in scheduling constraints, and the ability of the transportation systems in meeting an individual's mobility and accessibility needs (Hägerstrand 1970). STPs have been applied widely as measures of individual and joint accessibility in transportation, urban, and social sciences, both directly and indirectly as components of accessibility benefit measures based on spatial interaction and economic welfare theory (e.g., Burns 1979; Dong et al. 2006; Ettema and Timmermans 2007; Kwan 1998; Kwan 1999; Miller 1999; and Neutens et al. 2010).

Although STPs and NTPs are powerful measures of individual accessibility, they focus on time as the main constraint limiting accessibility. They do not capture other, non–temporal resource constraints that may limit accessibility. These constraints can include private resource limitations such as finite fuel capacity in conventional vehicles, the need to recharge batteries in electric vehicles (EVs) or limited monetary budgets for distance-based fares. Non-temporal resource constraints can also include common resources such as placing limits on vehicle emissions to preserve clean air.

This chapter extends the concept of STPs and NTPs to other resource limitations in addition to time. *Resource hyper-prisms* (RHPs) are prisms that capture incorporate private and/or common resource budgets in addition to time budgets. These prisms are the envelope of all possible space-time paths between two locations and times that satisfy speed limits, a time budget and other resource constraints on mobility. We present a mathematical program approach for network-based RHPs and a solution algorithm based on forward and backward DP.

We discuss several potential applications of RHPs and show our solution methods using a sketch network of Chicago to analyze accessibility considering carbon emission budget. We also use parts of Washington D.C. and Baltimore networks to illustrate realistic scenarios involving EVs and recharging stations along highways or in towns. By using our proposed RHP approach, not only we can track the level of resource (electricity) at any time and location to guarantee that the EVs do not run out of power while on the road, but also evaluate the impact of recharging stations location on the accessibility of the users. To the best of our knowledge, our framework is the first work studied this problem from this perspective. There are a number of research focused on this problem before, although they did not see the resource as an explicit dimension for the space-time network. For example, Liu et al. (2017) propose a mathematical model for constrained energy-efficient time-aware routing problem, denoted as CEETAR and solve it by an approximate dynamic programing, but not considering energy as an explicit dimension.

The remainder of this chapter is organized as follows. The next section provides a brief literature review on existing time geography frameworks. Section 5.3 describes in detail, the construction of a STR hyper-network as the foundation of our method. The fourth section presents a mathematical formulation to specify the borders of a RHP, followed by a DP solution approach. Section 5.5 provides results from the application of the proposed algorithm to the large-scale Chicago sketch transportation network as well as parts of Washington D.C. and Baltimore networks. Discussion, concluding remarks, and directions for future research form the sixth and final section of this chapter. We also provide the computational experiments on small-scale six-node transportation network and medium-scale Sioux Falls network in Appendix D for those researchers willing to reproduce the results.

5.2. Literature Review

Accessibility is defined as the ease of reaching desired destinations, activities, or services within an environment. People-based accessibility measures recognize the individuals' different activity schedules and available transportation resources, and therefore are sensitive to individuals' social and economic background (Miller 2005). A STP is an envelope of all feasible space-time paths between two activity locations, given the scheduled times for activities and the maximum moving speed (Miller 1991). Fig. 5.1 shows a space-time prism for an individual who wants to travel from his origin to destination. According to Fig. 5.1, the travel speed can be different in various directions. In this figure, the red circle on the XY plane shows the boundary of the accessible region for the individual based on his time restrictions.

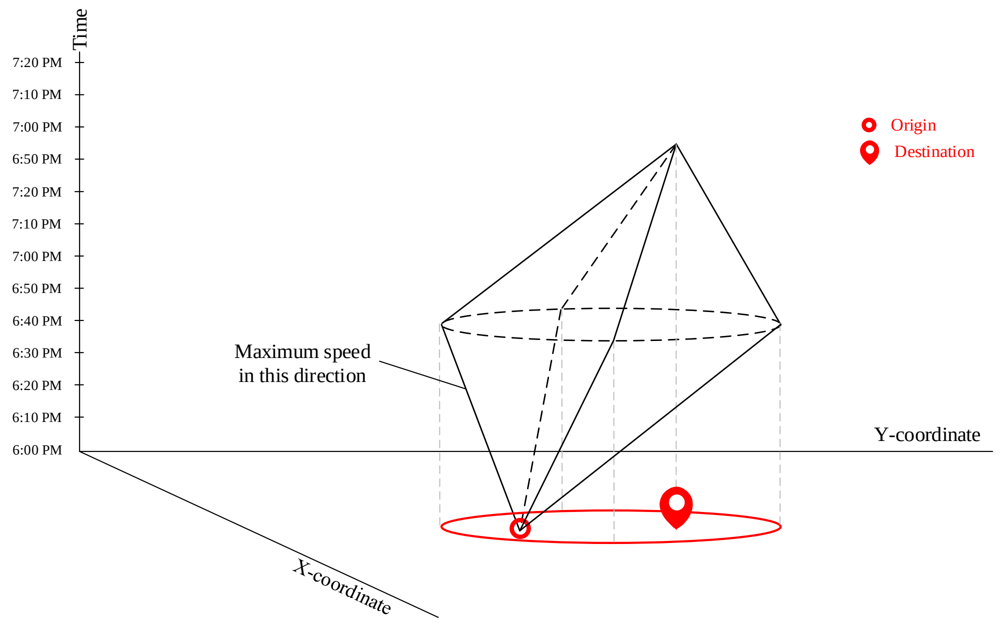


Fig. 5.1. A space-time prism for an individual who wants to travel from his origin to destination.

An NTP, as its name implies, is the STP confined by spatial networks (Kuijpers et al. 2010). NTPs have been widely used to study individuals' travel behaviors and space-time accessibilities (e.g. Horner and Wood 2014; Kwan 1999; Raubal et al. 2004; Tong et al. 2015; Widener et al. 2015).

STPs and NTPs have been introduced to explain how spatio–temporal constraints impose upon individuals' day-to-day activities and trip decisions and only consider time as a scarce resource. However, the resource constraints have attracted little attention. In fact, other resource constraints can determine space-time accessibility, such as limits or "budgets" for energy, emissions, or monetary expenses and may have significant impacts on transportation planning and traveling behaviors.

Road capacities, toll fare budgets, energy provisions, emission constraints, and available shared-mobility services are a few examples of resource constraints that have been studied in fields related to time-geography. For instance, the minimum travel time between locations is a key component to define NTP, and studies have proved that the road capacity, as a resource constraint, is critical in determining the network reliability and travel time between locations (Chen et al. 2002; Sullivan et al. 2010). The impacts of road capacity can be mitigated by identifying critical locations and setting toll fares, because individuals also have monetary budget on tolls (Brownstone and Small 2005; Irnich and Desaulniers 2005; Osenga 2005). Another example is the resource constraints of automobile vehicles, such as available recharging stations for EVs (e.g. Adler et al. 2016; Schneider et al. 2014) and perceptions/policies on carbon emission limits (e.g. Javid et al. 2014; Schwanen et al. 2012). The ever-increasing popularity of share-mobility also introduces new constraints such as the number of seats left for passengers and the available pickup/drop-off locations (e.g. Mahmoudi and Zhou 2016; Spieser et al 2014). Hence, considering these resource constraints can refine prisms as accessibility measures, and make prisms more practically useful. In this chapter, we mainly focus on constraints associated with the fuel consumption for EVs and emission budget for gasoline passenger cars.

About electricity consumption budget constraints for EVs, several studies have focused on EVs' charging stations network design to meet the needs of current and future EV fleets. These studies aim at locating charging facilities near the urban activity centers of EV owners (e.g. home, shopping malls, and workplaces) for their short distance trips to maximize the overall accessibility of the EV owners (e.g. Dashora et al., 2010, Frade et al., 2011; Sweda and Klabjan, 2011; Chen et al., 2013). To the best of our knowledge, most existing studies along this direction have focused on short distance trips (less than 100 miles round trip), while charging seems more important for long distance trips than short distance ones. In fact, with EVs, the concern is not city-level accessibility, since people rarely drive 130 miles in a typical day. Moreover, they can just plug in their EVs at home during the night. But limited EV ranges are a big concern when driving between cities. Where should these stations be placed to not only we can get between big cities, but also accessibility to other cities and towns along the way? Recently, Nie and Ghamami, 2013 developed an optimization model to study travel by EVs along a long corridor, in which the objective is to select the charging power at each station and the number of stations needed along the corridor to meet a given level of service such that the total social cost is minimized. In this chapter, we develop scenarios based on EVs and state/region level to study regional accessibility with charging stations along highways or in towns.

About emission budget constraints for gasoline cars, since highway vehicles account for 72 percent of total transportation emissions (Greene and Schafer, 2003), they have been the main focus of environmental protection and transportation agencies to reduce greenhouse gas emissions. Various strategies have been considered for potential vehicular emission reduction such as low-carbon alternative fuels, energy efficiency improvements, increasing the operating efficiency of the transportation system, and reducing travel. Several studies have been also conducted to estimate greenhouse emissions based on different input factors such as travel-related factors (e.g. distance travelled and speed) (An and Ross, 1993; Humphrey, 1996); highway network characteristics and conditions (e.g. geometries and road surface conditions) (Baker, 1994); vehicle-related factors (e.g. weight, engine type, and age) (Murrel, 1980); and other factors (e.g. ambient temperature and wind speeds) (Ahn, 1998). Motor Vehicle Emission Simulator (MOVES) is an air pollution emission estimator designed by the US Environmental Protection Agency (USEPA) in which vehicle emission rates are described as a combination of two factors: the emission source and the vehicle operating mode. Emission sources are classified in bins by vehicle characteristics such as vehicle types, fuel/engine technologies, ages, model years, engine size, and average weight fraction. The vehicle operating modes refer to vehicle operating conditions and are classified in bins of second-by-second vehicle activity characteristics, represented as vehicle specific power (VSP) which is a function of vehicle speed, road grade, and acceleration (USEPA, 2012). Recently, Zhou et al. 2015 have coupled mesoscopic simulation-based dynamic network loading framework DTALite with a simplified version of the EPA-MOVES, MOVES Lite, to evaluate the traffic dynamics and vehicle emission/fuel consumption impacts of different traffic management strategies. In this chapter, we do not aim to focus on the greenhouse emission reduction strategies or the input factors affecting the level of emission; instead, we intend to analyze an individual accessibility based on the emission budget constraints limiting his daily travel. We compute accessibility based on a time budget and an emissions budget, and then, develop scenarios based on changing the emissions budgets to address urban air quality concerns.

Apart from limitation of STPs and NTPs in dealing with the various types of resource constraints, the emerging paradigm toward sustainable transportation planning motivates a spectrum of new efforts and applications such as energy efficient vehicles, emission control zones, and ride-sharing designs (Banister 2008; Chan 2007; Chu and Majumdar 2012; Thomas 2009). They bring both challenges and opportunities to the study of people-based accessibility based on STPs and NTPs. This research defines a STR network that captures both network structures and available resources.

5.3. Problem Statement

In this section, we briefly discuss about the three-dimensional space-time (ST) networks, and then, explain how to extend it to a STR network. The classical space-time network has been developed to track the location of each individual at any time. The space consists of two dimensions (X and Y coordinates), by which the location of physical transportation nodes (e.g. intersections or freeway merge points) can be specified. The time horizon also has been discretized into a series of time intervals with the same time length. Fig. 5.2 shows a space-time network in which an individual departs from his origin located at $[x_1, y_1] = [20,10]$ at time 6:00PM and reaches his destination at $[x_2, y_2] = [0,10]$ at time 6:20PM. Note that in this particular example, we assume a unit of time has ten minutes length.

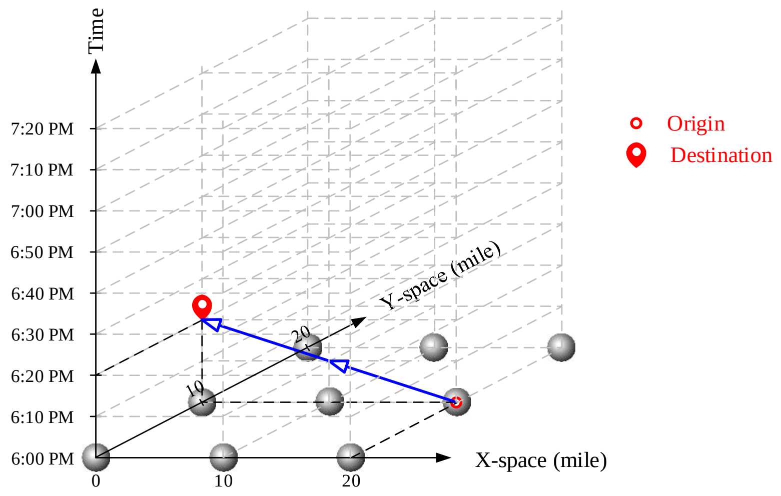


Fig. 5.2. A space–time network in which an individual departs from origin at 6:00PM and reaches destination at 6:20PM.

5.3.1. Resource Discretization

In order to illustrate STR networks, we initially map the two-dimensional space to a one-dimensional space, at which all nodes are positioned in a row. We also keep the time dimension as defined in ST networks. Thus far, we have a discretized space-time graph. Then, the resource dimension should be added to the two-dimensional space-time graph.

Suppose the carbon monoxide CO emission budget as a resource constraint in gasoline passenger cars. According to Eq. (5.1) (Stein and Walker, 2003), the CO emission rate (Ω) depends on travel time or speed. Note that in this equation, Ω is the CO emission rate in terms of grams of CO per vehicle per second and v is the vehicle's speed in terms of miles per hour (mph). In order to add the resource dimension to the ST graph, the resource should be presented in a discretized version. As shown in Fig. 5.3, the lowest and highest resource consumption rates (i.e. 0.18 and 1.88) occur in 39.23 mph (roughly 40 mph) and 120 mph speed, respectively. Since one minute has been assumed as the unit of time in this research, we need to convert resource consumption rate per hour to resource consumption rate per minute. Based on the range of resource consumption rates mentioned above, CO emission rate per minute is a number between 0.003 and 0.03. For the Chicago sketch network, the free-flow speed is assumed to be 25 mph, and according to Eq. (5.1), the fuel consumption rate at 25 mph speed is equal to 0.238 per hour or 0.004 per minute. Therefore, 0.004 is assumed as the unit of resource (i.e. emission) in this data set.

$$\Omega = -0.064 + 0.0056v + 0.00026(v - 50)^2 \tag{5.1}$$

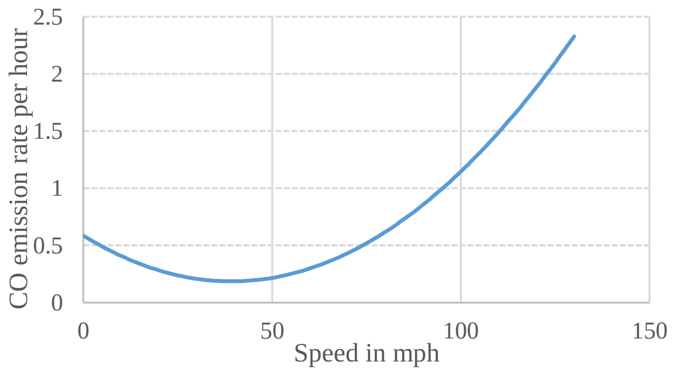


Fig. 5.3. CO emission rate per hour in different speeds.

According to Eq. (5.1), CO emission rate per minute is a function of vehicle speed, while as discussed in Section 5.2, in order to estimate the emission rate more precisely, travel-related factors, highway network characteristics and conditions, vehicle-related factors, and other factors may need to be considered. Motor vehicle emission simulator (MOVES) is an air pollution emissions estimation software designed by the US Environmental Protection Agency (USEPA). The emission model in MOVES estimates emissions for a wide range of on-road vehicles (e.g., cars, trucks, motorcycles, and buses). In this estimation approach, vehicle emission rates are defined as a combination of two factors: (1) the emission source and (2) the vehicle operating mode. Emission sources are categorized in bins by vehicle characteristics such as vehicle types, fuel/engine technologies, ages, model years, engine size, and average weight fraction. Operating modes refer to vehicle operating conditions and are categorized in bins of second-by-second vehicle activity characteristics, represented as VSP. VSP is a function of vehicle speed, road grade, and acceleration which accounts for kinetic energy, rolling resistance, aerodynamic drag, and gravity. The equation to calculate VSP adopted in MOVES (USEPA, 2012) is expressed as follows:

$$VSP = \frac{A}{M} \times v + \frac{B}{M} \times v^2 + \frac{C}{M} \times v^3 + (a + \sin(\phi)) \times v \tag{5.2}$$

, where A (metric ton), B (metric ton/(*m/s*)), and C (metric ton/$(m/s)^2$) refer to the rolling term, rotating term and drag term, respectively; M is the vehicle mass (metric ton); v is the vehicle speed (*m/s*); a denotes the vehicle acceleration (m/s^2); and ϕ is the road grade. Then, the calculated VSPs are categorized by speed and VSP ranges. A detailed classification is provided in Table 5.1.

Table 5.1
Definition of MOVES operating mode bins by speed and VSP ranges (USEPA, 2012)

0 mph < $v \le$ 25 mph		25 mph < $v \le$ 50 mph		v > 50 mph	
Operating Mode ID	Description	Operating Mode ID	Description	Operating Mode ID	Description
11	VSP < 0	21	VSP < 0	NA	NA
12	0 ≤ VSP < 3	22	0 ≤ VSP < 3	NA	NA
13	3 ≤ VSP < 6	23	3 ≤ VSP < 6	33	VSP < 6
14	6 ≤ VSP < 9	24	6 ≤ VSP < 9	35	6 ≤ VSP < 12
15	9 ≤ VSP < 12	25	9 ≤ VSP <12	NA	NA
16	12 ≤ VSP	27	12 ≤ VSP < 18	37	12 ≤ VSP < 18
Others:		28	18 ≤ VSP <24	38	18 ≤ VSP < 24
0	Braking	29	24 ≤ VSP <30	39	24 ≤ VSP < 30
1	Idling	30	30 ≤ VSP	40	30 ≤ VSP
NA = Not Applicable					

Table 5.2 presents average emission rate (e.g. CO2, NOx, CO, HC (g/h)) for zero-age passenger cars (Frey and Liu, 2013). In this case, a resource constraint can be defined as a restriction on the daily emission (e.g. CO2) budget for an agent (e.g. passenger car).

Table 5.2
Average emission rate for zero-age passenger cars (Frey and Liu, 2013)

Operating Mode ID	Energy (KJ/h)	CO2 (g/h)	NOx (g/h)	CO (g/h)	HC (g/h)
0	49206	3536	0.05	2.37	0.04
1	45521	3271	0.01	4.06	0.00
…	…	…	…	…	…
40	641649	46113	14.34	407.60	2.73

Suppose electricity consumption budget constraints as the resource constraints for EVs. A study conducted by the US Department of Energy (Electric Vehicle Operation Program, 1999) examined six different types of vehicles in urban versus highway driving under various conditions (e.g. headlight setting, auxiliary loads, and air conditioning). It found on average an EV can travel 2.5 miles for each kW h (kilo Watt hour) of energy. In this research, real-world data set corresponding to the geographical location of EV charging stations has been adopted. This data set is publicly available at https://www.afdc.energy.gov/data_download. We develop scenarios and consider a number of charging stations along highways and inside towns to study regional accessibility with charging stations along highways or in towns.

5.3.2. STR Hyper-network Construction

Finally, the obtained STR hyper-network helps us to track the level of resource and location of the individuals at any time. Fig. 5.4 shows the STR hyper-network corresponding to the example mentioned in Fig. 5.2. In this example, we assume that the individual starts his trip from his origin at time 6:00PM with resource (emission) level index $r = 0.000$, and respecting 0.005 emission rate per minute (the free–flow speed for all transportation links is assumed to be 60 mph in this example), he reaches his destination at 6:20PM with resource level index $r = 0.010$.

In this problem, the origin and destination of the individuals are given. What we are interested in is that how far an individual can travel based on his own time and resource constraints.

In order to address the aforementioned objective, we need to construct a high-dimensional network first. The first step for the network construction is defining the vertexes and arcs. A vertex in our STR network is recognized by a triplet of three different indexes: node index i, time index t, and resource level r. We also assume that vertex (i, t, r) is connected to vertex (i', t', r') through directional arc (i, i', t, t', r, r').

The travel time and emission production corresponding to arc (i, i', t, t', r, r') can be calculated by $(t' - t)$ and $(r' - r)$, respectively. Note that at 60 mph speed, $r' = r + 0.005(t' - t)$. The cost of each arc is a ratio of its emission production. In this research, the ratio is assumed to be 1.

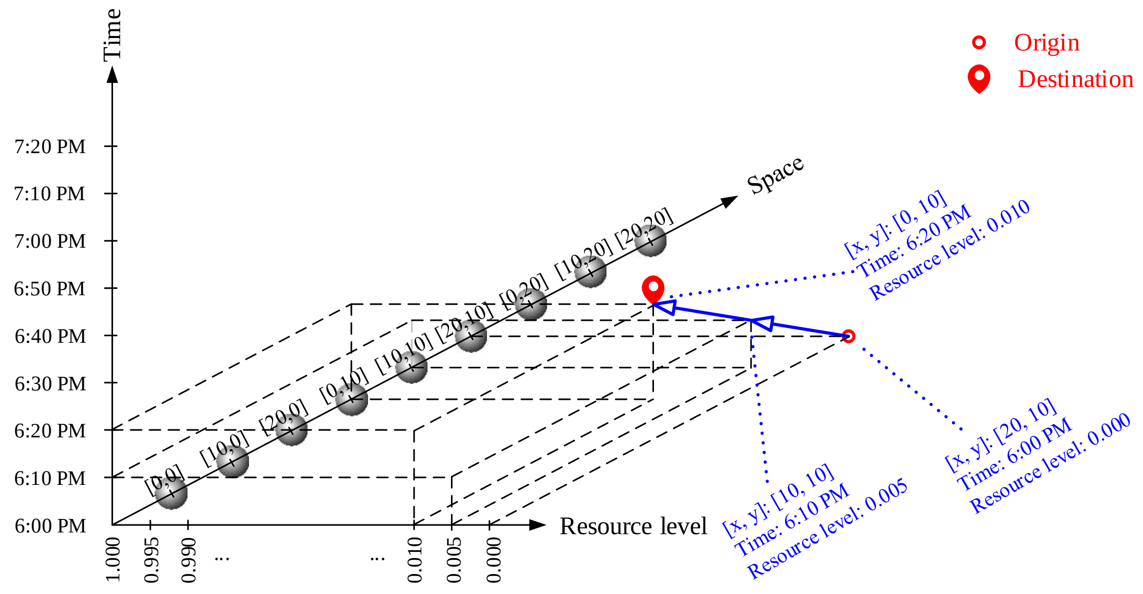


Fig. 5.4. Our proposed STR hyper-network in which the individual starts his trip from his origin at time 6:00 PM with resource level index $r = 0.000$, and reaches his destination at time 6:20 PM with $r = 0.010$.

Let us assume resource is the fuel. Note that everything mentioned in the following paragraphs is also correct for carbon emission. In order to find an individual's hyper-network, we assume that the individual starts his route from his origin spot at the beginning of the time horizon with the highest level of fuel, i.e. r_{max}, and ends it to his destination location at the end of the time horizon with fuel level r $(0 \leq r \leq r_{max})$. By considering this rule, each individual may explore the transportation network as much as possible with respect to his time and resource budgets. Waiting at any location and time with any level of fuel is allowed. The length of a waiting arc is assumed to be one minute.

We also assumed that the resource is not consumed while waiting. Therefore, the cost of a waiting arc is zero.

Since the level of resource at the end of the time horizon, i.e. $t = T,$ may vary from 0 to r_{max}, to be able to mathematically model this problem, we need to end the individual's route to a fixed vertex. Therefore, we define a virtual sink vertex at time $T + 1$ with $r = 0$, whom all feasible destination vertexes are connected to. Finally, to find the hyper-network, it is sufficient to find all vertexes that are reachable from both individual's origin and destination.

5.4. Mathematical Programming Approach for Solving Network-based Hyper-prisms

Back to our discussion about finding the hyper-prism, we initially use a forward DP algorithm to find the shortest path from the individual's origin vertex to the sink vertex. In fact, the problem mentioned above, is a time-dependent resource-dependent shortest path problem which can be solved by many computationally efficient algorithms already proposed in the literature for this optimization problem.

5.4.1. Mathematical Program

Table 5.3 lists the notations for the sets, indices, and parameters in this problem. The mathematical programming for this problem has been provided as follows:

Table 5.3
Indices, parameters, and variables in our proposed model for finding the hyper–prism.

Symbol	Definition
(i, i')	Index of the physical link between adjacent nodes i and i'
$TT(i, i', t)$	Link travel time from node i to node i' starting at time t
T	Ending time of the time horizon
r, r'	The corresponding resource level at vertexes (i, t) and (i', t')
o	Origin node
d	Destination node
ϕ	Sink node
$c_{i,i',t,t',r,r'}$	Routing cost of arc (i, i', t, t', r, r') which is equal to $r - r'$
$x_{i,i',t,t',r,r'}$	= 1 if the individual travels from vertex (i, t, r) to vertex (i', t', r'); 0 otherwise

The objective is to find the time–dependent resource–dependent least cost path.

$$Min\ \sum_{(i,i',t,t',r,r')} \{c_{i,i',t,t',r,r'} \times x_{i,i',t,t',r,r'}\} \tag{5.3}$$

subject to:

Flow balance constraint for the origin vertex:

$$\sum_{i',t',r'} x_{i,i',t,t',r,r'} = 1 \tag{5.4}$$

, where $i = o$, $t = 0$, and $r = r_{max}$.

Flow balance constraint for the sink vertex:

$$\sum_{r} x_{i,i',t,t',r,r'} = 1 \tag{5.5}$$

, where $= d$, $i' = \phi$, $t = T$, $t' = T + 1$, and $r' = 0$.

Flow balance constraint at intermediate vertexes

$$\sum_{i',t',r'} x_{i,i',t,t',r,r'} - \sum_{i',t',r'} x_{i',i,t',t,r',r} = 0 \qquad \forall i, t, r \tag{5.6}$$

Binary definitional constraint

$$x_{i,i',t,t',r,r'} \in \{0, 1\} \qquad \forall i, t, r, i', t', r' \tag{5.7}$$

5.4.2. Solution Algorithm: Forward and Backward DP

We initialize the label of all vertexes except origin vertex to infinity and move forward to update any vertex that is reachable for the origin vertex. Note that we initialize the label of origin vertex to zero. Then, we use a backward DP algorithm to find the shortest path from the sink vertex to the individual's origin. Again, we initialize the label of all vertexes except the sink vertex to infinity and move backward to update any vertex that is reachable for the sink vertex. Note that we initialize the label of the sink vertex to zero in this case. Finally, we determine the hyper-prism for the individual by taking those vertexes that are reachable for both the individual's origin and sink vertexes. The pseudo code corresponding this algorithm has been provided as follows. In this pseudo code, terms "pred", "succ", "max_fc", and "min_fl" stand for predecessor, successor, the maximum allowed fuel/energy consumption, and the minimum allowed fuel level.

```
// resource hyper-prism search algorithm

// step 1: forward DP algorithm
// initialization
     Forward_Label(origin vertex) := 0;
     Forward_Label(vertex(i,t,r)) := +∞;
     Node_pred(vertex(i,t,r)) := -1;
     Time_pred(vertex(i,t,r)) := -1;
     Resource_pred(vertex(i,t,r)) := -1;

// updating the forward labels
   For each time t
   Begin
     For each link l // i and i' are given; i & i' are equal for waiting
arcs
     Begin
       For each resource r // r >= min_fl
       Begin

         Derive t'; // based on i, i', t, and r
         Drive r'; // based on i, i', t, t', and r (r' must be >= min_fl)

         If Forward_Label(vertex(i,t,r)) + (r - r') <
         Forward_Label(vertex(i',t',r'))
```

```
        {
        Forward_Label(vertex(i',t',r') := Forward_Label(vertex(i,t,r)) +
        (r - r');
        Node_pred(vertex(i',t',r')) := i;
        Time_pred(vertex(i',t',r')) := t;
        Resource_pred(vertex(i',t',r')) := r;
        }
      End
    End
  End

// step 2: backward DP algorithm
// initialization
    Backward_Label(sink vertex) := 0;
    r_sink vertex := r_max;
    Backward_Label(vertex(i',t',r')) := +∞;
    Node_succ(vertex(i',t',r')) := -1;
    Time_succ(vertex(i',t',r')) := -1;
    Resource_succ(vertex(i',t',r')) := -1;

// updating the backward labels
  For each time t'
  Begin
    For each link l // i and i' are given; i & i' are equal for waiting
    arcs
    Begin
      For each resource r' // r' >= min_fl
      Begin

        Derive t; // based on i, i', t', and r'
        Drive r; // based on i, i', t, t', and r' (r must be >= min_fl)

        If Backward_Label(vertex(i',t',r')) + (r - r') <
        Backward_Label(vertex(i,t,r))
        {
        Backward_Label(vertex(i,t,r) := Backward_Label(vertex(i,t,r)) +
        (r - r');
        Node_succ(vertex(i',t',r')) := i;
        Time_succ(vertex(i',t',r')) := t;
        Resource_succ(vertex(i',t',r')) := r;
        }
      End
    End
  End


// step 3: calculating the hyper-prism
  For each node i
  Begin
     For each time t
     Begin
        For each resource r // r >= min_fl
        Begin
        If Forward_Label(vertex(i,t,r)) +
        Backward_Label(vertex(i,t,r)) <= max_fc
        {vertex(i,t,r) is in the hyper-prism}
        End
     End
  End
```

5.4.3. Search Region Reduction

Our experiments executed on an Intel Workstation running two Xeon E5–2680 processors clocked at 2.80 GHz with 20 cores and 192GB RAM running Windows Server 2008 x64 Edition. In order to tackle the curse of dimensionality in this algorithm, we use a few heuristics to reduce the search space as much as possible. For instance, we reduce the search space for the NTP by overlaying the corresponding STP on the network and restricting the NTP to the vertexes within the STP (Kuijpers and Othman, 2009), since the STP always overbounds the NTP. We also calculate the NTP first (by applying forward and backward DP algorithms), and then we extend the resource dimensions only for those vertexes belonging to the NTP. As a result, we do not apply forward and backward DP for the whole three-dimensional search space.

We also use another heuristic for search space contraction. Suppose after calculating the space-time prism, the least time needed to travel from origin node to a node in the network is τ; therefore, we do not need to construct the vertexes corresponding that node whose time index is less than τ. By considering this rational rule, we can skip several vertexes that can be constructed before time τ for the corresponding node. Similarly, suppose the least time required to travel from a node in the network to the sink node is τ'; so, the latest time one can depart from this node to reach the sink node is $T + 1 - \tau$. Note that in this expression, $T + 1$ refers to the time stamp we defined for the sink vertex. Again, by considering this rational rule, we can skip several vertexes that can be constructed after time $T + 1 - \tau$ for the corresponding node.

5.5. Computational Experiments

In this section, we develop scenarios on city level over Chicago sketch network and state/region level over parts of Washing, D.C. and Baltimore network. In the city level, we compute accessibility based on a time budget and an emissions budget; we further develop scenarios based on changing the emissions budgets. In the state/region level, we develop scenarios based on EVs and regional accessibility with charging stations along highways or in town. The complete C++ implementation of the proposed DP algorithm and data set for the Chicago sketch network are available at https://github.com/mmahmou2/Space-time-resource-Prism.

The time horizon has been discretized into a series of time intervals with the same time length. Without loss of generality, we can assume that a unit of time has one minute length. We have also provided our computational experiments over six-node transportation network with 13 transportation links as a small-scale network and Sioux Falls network with 24 nodes and 76 links as a medium-sized network with enough details in Appendix D for those researchers willing to reproduce the examples.

5.5.1. Illustration of Accessibility

Chicago sketch network includes 933 nodes and 2,967 links illustrated in Fig. 5.5. The free-flow speed is assumed to be 25 mph for all transportation links, the length of time horizon is 80 minutes, and nodes 421 and 424 are an individual's origin and destination, respectively. We assume that the index of minimum level of resource (emission) is 0.000. We also assumed that maximum emission production index (max_ep) is 0.320.

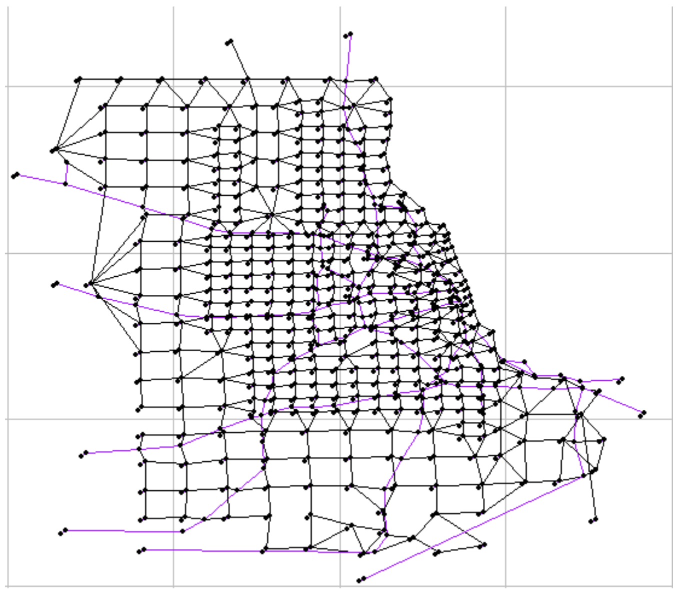


Fig. 5.5. Chicago sketch network with 933 nodes and 2,967 transportation links.

Fig. 5.6 illustrates the nodes that are accessible for both origin and destination. The following figures corresponding to the accessible regions have been plotted by google fusion tables. We have provided the procedure of creating the heatmaps by google fusion tables at https://github.com/mmahmou2/Space-time-resource-Prism. In this figure, nodes with more accessibility are shown by red color, while less accessible nodes are shown by green color. Note that a node is more accessible if and only if it is reachable in longer period of time and larger range of resource levels. The CPU time for running our algorithm on this network is 102.7 seconds.

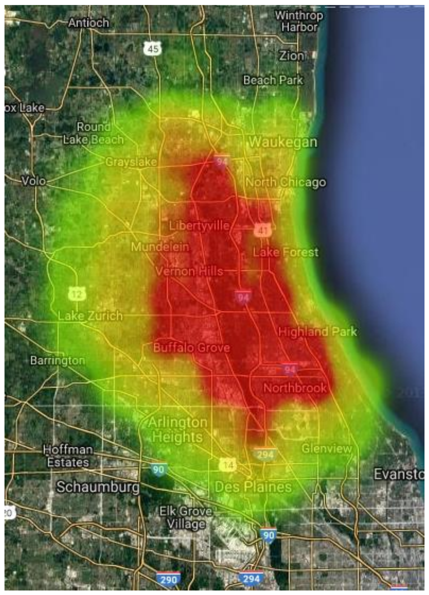
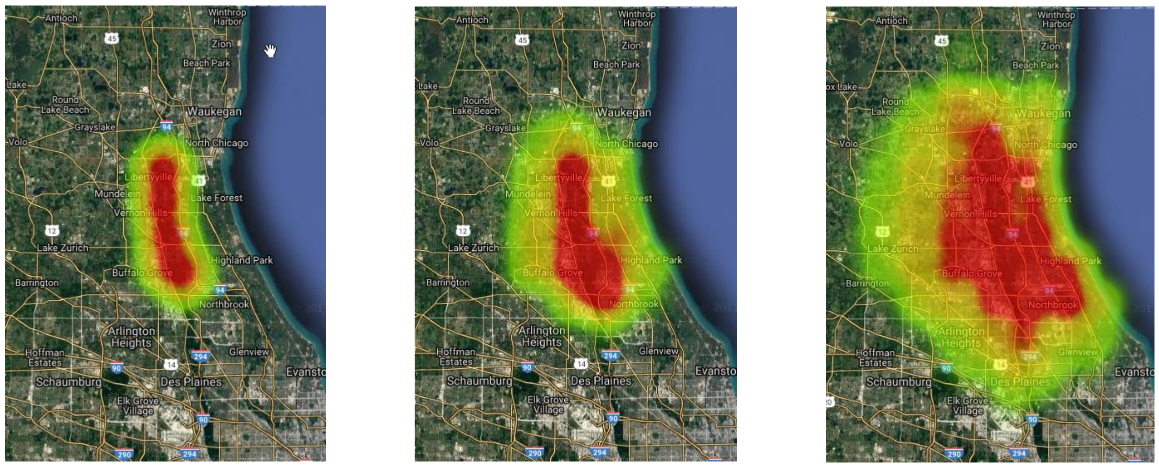


Fig. 5.6. Accessible nodes for both origin and destination respecting time and resource budgets.

5.5.2. Effect of Time and Resource Budget on Space-time Prisms

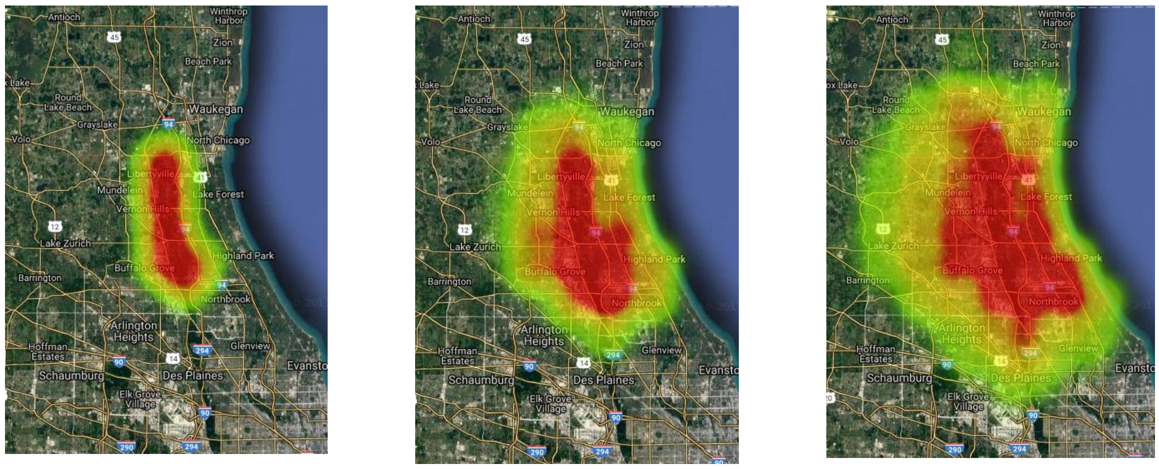
In Fig. 5.7, considering 0.320 as the maximum emission production index, we examine the effect of time budget on the space-time prism by testing our algorithm for 30, 50, and 80 minutes time budgets. The individual's origin and destination are similar to the ones used for the example above. It is expected that by increasing the value of time budget, more nodes be reachable for the individual.



(a) time budget = 30 minutes (b) time budget = 50 minutes (c) time budget = 80 minutes

Fig. 5.7. The effect of time budget on the STR network.

In Fig. 5.8, considering 80 minutes as the time budget, we examine the effect of resource budget on the space-time prism by testing our algorithm for 0.120, 0.200, and 0.320 resource budget (max_ep) indexes. Fig. 5.8 shows that by increasing the value of resource budget, more nodes are reachable for the individual.



(a) resource budget = 0.120 (b) resource budget = 0.200 (c) resource budget = 0.320

Fig. 5.8. The effect of resource budget on the STR network.

5.5.3. Regional Accessibility with EV Charging Stations along Highways or in Town

On average an EV can travel 2.5 miles for each kW h (kilo Watt hour) of energy. We use parts of Washington, D.C. and Baltimore networks (with 12,145 nodes and 30,697 links) illustrated in Fig. 5.9 for a use case of an individual with an EV to study regional accessibility with charging stations along highways or in towns to show how a RHP can measure the accessibility impacts of new EV charging stations. We extracted this network from the state-wide network used by Erdoğan et al. (2014) for their state-wide dynamic traffic routing model.

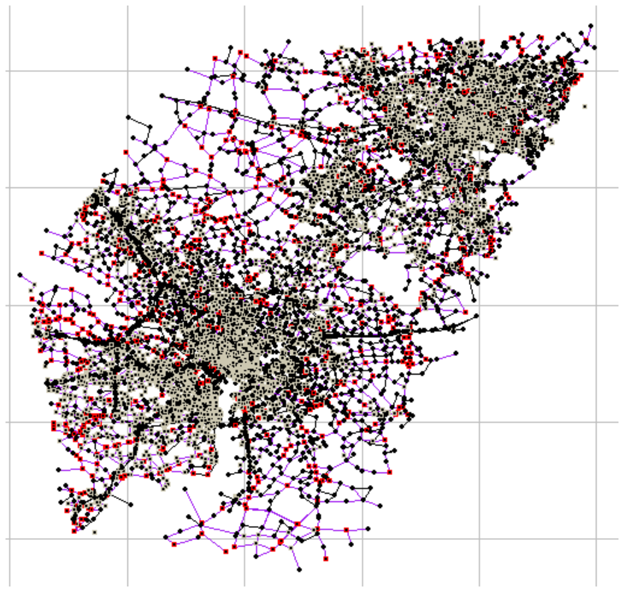


Fig. 5.9. Parts of Washington, D.C. and Baltimore networks (with 12,145 nodes and 30,697 links)

In our experiments, the length of time horizon is assumed to be 120 minutes. We also changed our C++ code to improve the computational efficiency for large-scale transportation networks. Fig. 5.10 illustrates the STR network for an individual with an EV when no charging station has been added to the network. In this network, 7,438 nodes are accessible for the individual whose origin and destination are located inside the city of Washington, which means that 61.24% of the whole network is accessible for the individual in this scenario. The CPU time for running our algorithm on this network is 236.6 seconds.

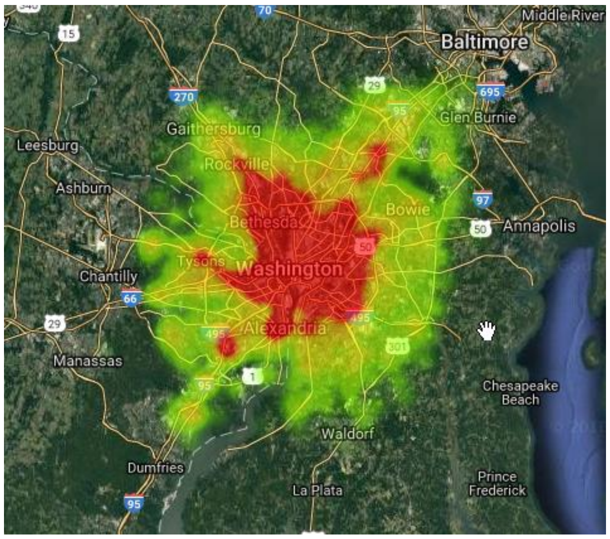


Fig. 5.10. STR network for an individual with an EV when no charging station has been added to the network.

We tested our algorithm for the same individual with the same origin and destination, while four charging stations inside Washington, D.C. have been added to the network. In this case, 7,648 nodes or equivalently, 62.97% of the whole network is accessible for the individual. We also tested our algorithm for the case in which four charging stations along the main corridor connecting Washington, D.C. to Baltimore have been added to the network. In this case, 8,247 nodes or equivalently, 67.90% of the network is accessible for the individual. Finally, we tested our algorithm for the case in which we combine the first two scenarios (four charging stations inside the town and four charging stations along the main corridor). The results show that 8,302 nodes are accessible in this scenario, which is 68.35% of the whole network.

According to our results, more accessibility for individuals is expected if charging stations are established along corridors and inside towns both; however, the installation cost of a charging station basically forces us to decide between the first two scenarios. According to our result, we may conclude that establishing EV along main corridor provides more accessibility for the users With EVs in comparison to establishing them inside the town. Moreover, as discussed before, the concern with EV is not city-level accessibility, since people rarely drive 130 miles in a typical day, and they can just plug in their EVs at home during the night.

5.5.4. Capturing Non–temporal Resource Budgets in SPTs and NTPs

STPs and NTPs only capture accessibility as constrained by space and time scheduling constraints and do not see other resource constraints that may limit accessibility. These resources can include private resources such as a battery that needs recharging, a mode of transportation that a traveler chooses for his trip, or amount of money that a traveler desires to spend for his trip as well as common resources such as clean air. The main contribution of this chapter is that our method is supposed to extend NTPs to capture both types of resource constraints. For example, in order to address how accessibility of an individual can be affected by choosing different transportation modes, we may need to define a dimension called “mode”. This dimension tracks the mode of transportation that the individual uses to travel from vertex X to Y. Therefore, a vertex in our space-time-resource-mode network is recognized by a quadruple of four different indexes: node index i, time index t, resource level r, and transportation mode index m.

Similar to what we defined for arcs in the three-dimensional STR networks, vertex (i, t, r, m) is connected to vertex (i', t', r', m') through directional arc $(i, i', t, t', r, r', m, m')$. Assume that a traveler is located at vertex (i, t, r). It is clear that using different modes of transportation to transport the individual from node i to i' may result in different t' and r'. Note that for any transportation arc, m and m' are the same, except for a transfer arc by which the individual changes the transportation mode. For example ride-sharing is a mode of transportation in which a passenger hires a driver to take him exactly where he needs to go, but he may share his ride with one or more than one passenger. It is obvious that using ride-sharing mode of transportation can lower the fuel consumption/emission per person per mile and may save the natural resources. Table 5.4 provides a list of transportation modes and their corresponding fuel consumption in terms of BUT (British Thermal Union) per passenger-mile and MJ (Mega Joule) per passenger-kilometer (Davis et al. 2011) for year 2009.

Table 5.4
List of transportation modes and their corresponding fuel consumption in 2009 reported by Davis et al. (2011).

Transportation mode	Average passengers per vehicle	BUT per passenger–mile	MJ per passenger–kilometer
Rail (intercity Amtrak)	20.9	2,435	1,596
Motorcycles	1.16	2,460	1.61
Rail (transit light and heavy)	24.5	2,516	1.649
Rail (commuter)	32.7	2,812	1.843
Air	99.3	2,826	1.853
Cars	1.55	3,538	2.319
Personal trucks	1.84	3,663	2.401
Buses	9.2	4,242	2.781
Taxi	1.55	15,645	10.257

To sum up, transportation mode can be treated as a kind of private resource in our proposed multi-dimensional hyper-prism, and individuals have this freedom to choose their transportation mode to increase their accessibility by saving their resources to some degree.

5.6. Conclusion

Although mobility has many benefits, it also has substantial negative impacts on local and global environments. Transportation is a major consumer of non-renewable natural resources, as well as a key source for greenhouse gas emissions and damaging air pollutants. Although space-time prisms have been introduced as the measure of individuals' accessibility to explain how spatio-temporal constraints impose upon individuals' day-to-day activities and trip decisions, the resource constraints have attracted little attention. In this research, in order to examine the effect of resource budget constraints on an individual's accessibility, we introduce a new dimension specifically for the resource to the classical space-time network. Having this, the level of resource for each individual at any location and time can be monitored. Fuel, money, carbon emission, and transportation mode are a few out of many types of resources in the real-world transportation systems. We applied a forward and backward resource-dependent time-dependent DP to find the resource hyper-prims for each individual. Finally, we tested our algorithm on small, medium, and large scale transportation network to show the solution optimality as well as computational efficiency of our developed algorithm.

The current work could be extended by considering different types of resources, particularly for the transportation mode, by which we can evaluate the effect of multi-modal transportation options on the accessibility of the travelers.

CHAPTER 6

HOW MANY TRIP REQUESTS COULD WE SUPPORT? AN ACTIVITY-TRAVEL BASED VEHICLE SCHEDULING APPROACH

6.1. Motivation and Background

The past decade has witnessed unprecedented advances in the auto industry, specifically in the domain of autonomous vehicle technologies. Several auto companies have forged new paths and introduced vehicles of the future that need minimal human intervention for their operation (Tesla Motors Team, 2015; Sherman, 2016). Ridesourcing, operated by TNCs such as Uber and Lyft, is another game changing technology introduced in recent times. TNCs aim to provide reliable and inexpensive personalized travel options that combine the best of personalized transport (for example, door-to-door travel), as well as transit services (where the users pay per trip and do not have to drive the vehicle themselves). Recent reports show that 12% of registered voters across the United States used ridesourcing services at least once in the past month (Morning Consult, 2015).

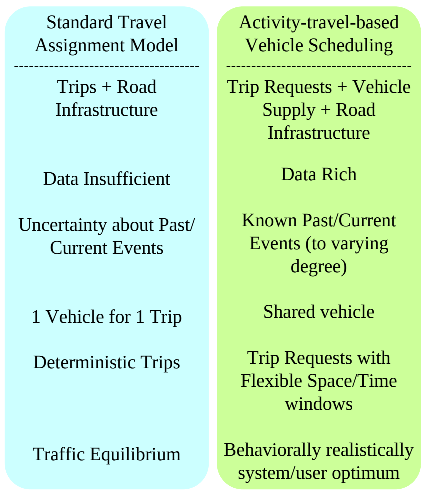
The rapidly growing popularity of TNCs coupled with autonomous vehicle technologies, could potentially redefine the way in which individuals schedule and execute their activities and also the way in which travel demand is managed by network operators. For the traveler, the freedom from having to drive could lead to more flexible activity schedules and increased productivity while travelling. On the other hand, network operators could handle demand by incentivizing/dis-incentivizing travel during a certain portion of the day (similar to surge pricing by Uber), or along a specific route. There is growing interest in the field to study incentive-based demand management strategies (for example, see Hu et al. 2014). It is therefore of critical importance to understand and accurately depict these transformative technologies and their implications for activity-travel patterns in travel demand model systems.

Great strides have been made in the past couple of decades in advancing travel demand modeling from the traditional 4-step travel demand models where demand and supply sides were considered static to present day state-of-the art integrated travel model systems. On the travel demand front, the profession has progressed from traditional trip-based methods to ABMs, which date back to the pioneering work of Kitamura (1988) (see Rasouli and Timmermans (2014) for a detailed synthesis on ABMs). ABMs view travel as derived demand, arising from the necessity of individuals to participate in various activities. This facilitates representing travel in a behaviorally realistic way in ABMs. On the other hand, network supply/simulation has progressed from static traffic assignment to DTA models that employ microscopic simulation and are capable of evaluating various traffic management strategies on the fly (Peeta and Ziliaskopoulos, 2001). There are ongoing efforts to tightly integrate the ABMs with DTA models with a view to accurately predict the impacts of dynamic pricing strategies and real-time information provision (Zockaie et al. 2015). While some of the integrated models operate in a sequential paradigm (exchange of information between the model components happens after a full iteration, for example Lin et al. 2008; Hao et al. 2010), others employ a tighter integration where information is exchanged on a more continuous basis (Balmer et al. 2009; Pendyala et al. 2012; Auld et al. 2016).

While the integrated models developed so far address modeling needs for the current array of travel options (modes, demand management strategies, etc.), they do not adequately handle emerging transportation technologies (ride-sharing services, autonomous vehicle technologies) that are increasingly penetrating the marketplace. For example, in an autonomous taxi fleet future, how would individuals go about scheduling their activities? How would the demand arising in such a situation impact network performance? Conversely, for a given fleet size, how many activities can a transportation networking company support? Integrated travel demand models of the present day would not be able to answer these questions for a variety of reasons.

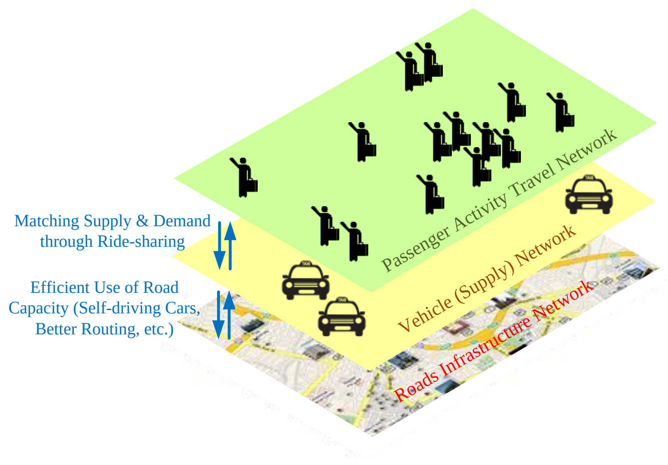
ABMs still operate based on zonal level information (such as skims, by time-of-day) provided by DTA models. The ABMs are oblivious to network logistics such as availability of ridesourcing options and incentives/disincentives customized to specific trips/travelers. On the other hand, the VRP, used to depict ride-sharing services in DTA models, view travel as disjoint trips that are independent of each other (Toth and Vigo, 2014). The solutions to VRP are typically optimization-based and lack a sound behavioral foundation. Solutions to VRP in the standard DTA models are often aimed at serving the maximum number of trip requests without taking into consideration the precedence constraints (or linkages) between the trips. Consider an individual's schedule comprising of three trips a) pick-up child, b) accompany child to the playground and c) take the child home. A VRP algorithm could produce a solution where trip requests for activities ‘b’ and ‘c’ are served, but in reality, activities ‘b’, and ‘c’ have a precedence constraint of engaging in activity ‘a’. This vital behavioral constraint is ignored in the VRP optimization techniques incorporated in DTA models.

Fig. 6.1(a) compares the characteristics of a standard dynamic traffic assignment system with the activity-travel based vehicle scheduling system. Due to the flexibility of the service offered by vehicle service providers and a vast variety of traveler’ behaviors (e.g. different levels of traveler flexibility in terms of departure and/or arrival time windows, trip cost budget, and ride synchronization), future dynamic transportation network models must consist of various layers of passengers and vehicle service providers interacting with one another (Fig. 6.1(b)).



Standard Travel Assignment Model	Activity-travel-based Vehicle Scheduling
Trips + Road Infrastructure	Trip Requests + Vehicle Supply + Road Infrastructure
Data Insufficient	Data Rich
Uncertainty about Past/ Current Events	Known Past/Current Events (to varying degree)
1 Vehicle for 1 Trip	Shared vehicle
Deterministic Trips	Trip Requests with Flexible Space/Time windows
Traffic Equilibrium	Behaviorally realistically system/user optimum

(a) Comparison between standard DTA and new vehicle scheduling system


Matching Supply & Demand through Ride-sharing
Efficient Use of Road Capacity (Self-driving Cars, Better Routing, etc.)
Passenger Activity Travel Network
Vehicle (Supply) Network
Roads Infrastructure Network

(b) Different layers of activity-travel vehicle scheduling system

Fig. 6.1. (a) a comparison between standard traffic assignment and proposed activity-travel vehicle scheduling system (adapted from Mahmassani, 2012); (b) layers of passengers’ requests, vehicles, and roads infrastructure network.

The objective of this chapter is two-fold i) add to the existing knowledge in the VRP domain by proposing an algorithm that incorporates behavioral realism into VRP; ii) facilitate the representation of emerging technologies in ABM-DTA integrated models by providing ABMs with a richer set of information made available by the proposed algorithm. The objective of the chapter is achieved by formulating and solving a time-discretized multi-commodity network flow model. The solution for the proposed algorithm takes into account the time-space constraints as well as activity hierarchy (precedence) constraints. It is envisioned that the proposed algorithm would enable the provision of a much richer set of information to ABMs, thus enabling ABMs to include ride-sharing/ride-hailing as an additional mode of travel. For example, if individuals are provided with a price (in the form of a Lagrangian multiplier) to undertake a trip, they can determine whether or not to engage in a discretionary activity based on the prevailing 'price' for the trip to get to that activity.

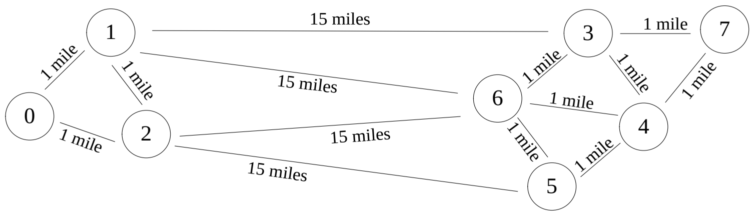
The remainder of the chapter is organized as follows. The next section describes in detail, the construction of the activity-travel graphs for passengers and SST networks for vehicles, and the third section presents the mathematical formulation of the time-discretized multi-commodity network flow model, as well as the solution approach. The fourth section provides results from the application of the proposed algorithm to the Phoenix subarea transportation network. Discussion, concluding remarks, and directions for future research form the fifth and final section of the chapter.

6.2. Network Construction for Passengers and Vehicles

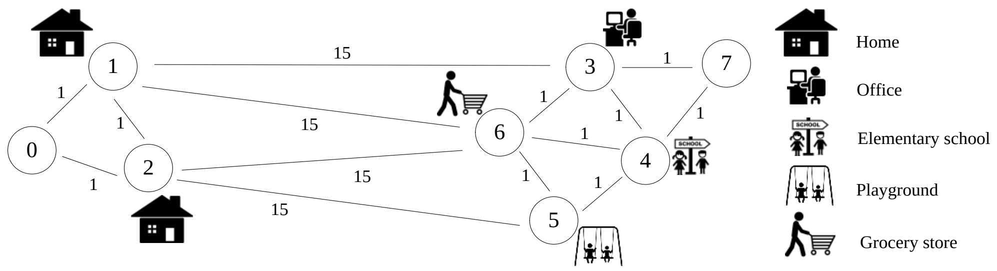
This section describes the construction of activity networks for passengers followed by the explanation of multi-dimensional SST network construction for vehicles. In a vehicle network, adding the ‘under-service trip requests state’ dimension helps in tracking the execution status of the passengers’ trip requests at any time. Interested readers are referred to a recent paper by Mahmoudi and Zhou (2016) for more details about how to construct a SST network.

6.2.1. Illustrative Example for Passenger Activity-travel Network Construction

This section details the construction of the graph containing passenger p’s activities using an example. Take an eight-node transportation network, illustrated in Fig. 6.2(a). Suppose two passengers, each with a specific origin and destination. Each passenger intends to perform a set of activities during the day (some of them are mandatory and others are optional). Table 6.1 presents the information related to the passengers’ trip requests.



(a) an eight-node transportation network



(b) activities location on the transportation network

Fig. 6.2. (a) an eight-node transportation network; (b) the transportation network in which the location of passengers’ activities has been specified.

The location of passengers' activities has been depicted on the eight-node transportation network in Fig. 6.2(b). In this example, passenger p_1 has a trip request from his home to office (working at office is a mandatory activity for p_1). Moreover, he would like to shop for groceries after work (note that this activity is optional). On the other hand, passenger p_2 has to drop off his kid at school first thing in the morning, and then go to work. Although both the activities in the morning are mandatory for passenger p_2, he has more flexible schedule (with discretionary activities) after work. He may (1) directly go home from office and assign the task of picking up the kid from school to his wife, (2) pick up the kid from school and go home, or (3) pick up the kid from school, take her to the playground and play with her for half an hour and then return home. In the latter alternative, taking the kid to the playground is dependent on picking her up from school.

Table 6.1
Information related to the trip requests of passengers

Passenger p_1			
Origin	Location		
Home	Node 1		
Destination	Location		
Home	Node 1		
Activity	Location	Type	Time window
Working at office	Node 3	Mandatory	[8:00 AM, 4:00 PM]
Shopping groceries	Node 6	Optional	[4:05 PM, 4:50 PM]
Passenger p_2			
Origin	Location		
Home	Node 2		
Destination	Location		
Home	Node 2		
Activity	Location	Type	Time window
Drop-off the kid at school	Node 4	Mandatory	[7:55 AM, 7:55 AM]
Working at office	Node 3	Mandatory	[8:00 AM, 4:00 PM]
Pick up the kid from school	Node 4	Optional	[4:10 PM, 4:10 PM]
Play with kid at playground	Node 5	Optional	[4:15 PM, 4:45 PM]

To construct the activity graph for each passenger, it is sufficient to arrange all possible trip requests respecting their type and time window. Fig. 6.3(a) and Fig. 6.3(b) presents the graphs of travel activities for passengers p_1 and p_2, respectively. In these graphs, each passenger should start and end his activity trip chain (or a tour) at the same location (home).

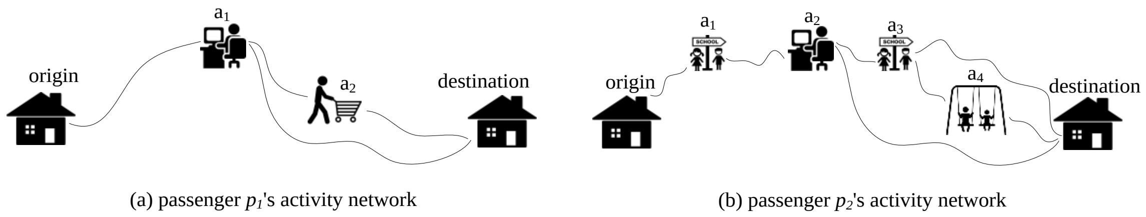

a1
origin
a2
destination
(a) passenger p1's activity network
a1
a2
a3
origin
a4
destination
(b) passenger p2's activity network

Fig 6.3. (a) passenger p_1's activity-travel graph; (b) passenger p_2's activity-travel graph.

6.2.2. Vehicle SST Network Construction

The construction of multi-dimensional SST networks for vehicles is explained in this section with the help of the example mentioned above. Note that in the SST network representation, an activity can only performed along the corresponding activity link. In order to consider the precedence constraints (e.g. drop-off a passenger at an activity location should occur before his pickup from there) as well as vehicle capacity constraints, a new dimension called as the "under-service trip requests state" is introduced. With the help of this definition, the execution status of passengers' trip requests in each vehicle can be tracked at any time within the vehicle time horizon. In the example mentioned above, we assume that the vehicle has 3 seats available for serving the passengers. Let $r(p, a, a')$ denote passenger p's trip request in which he leaves activity location a to perform activity a'. From Fig. 6.3(a), there might be four trip requests for passenger p_1, i.e. $r_{p_1}^1 = r(p_1, origin, a_1)$, $r_{p_1}^2 = r(p_1, a_1, a_2)$, $r_{p_1}^3 = r(p_1, a_1, destination)$, $r_{p_1}^4 = r(p_1, a_2, destination)$, while according to Fig. 6.3(b), seven trip requests can be possible for passenger p_2, i.e. $r_{p_2}^1 = r(p_2, origin, a_1)$, $r_{p_2}^2 = r(p_2, a_1, a_2)$, $r_{p_2}^3 = r(p_2, a_2, destination)$, $r_{p_2}^4 = r(p_2, a_2, a_3)$, $r_{p_2}^5 = r(p_2, a_3, destination)$, $r_{p_2}^6 = r(p_2, a_3, a_4)$, $r_{p_2}^7 = r(p_2, a_4, destination)$. The under-service trip requests state (denoted by w from now on) is explained with the help of Table 6.2. Table 6.2 presents the under-service trip requests state w at any node i and time t. Note that $w = [\_\,\_\,\_]$ is the null state in which the vehicle is not involved in any passenger's trip request. Figures 6.4(a) and 6.4(b) show the route of the passengers when they share their ride with each other. Fig. 6.5(a) also depicts the vehicles two-dimensional space-time network when two passengers are served through the ride-sharing.

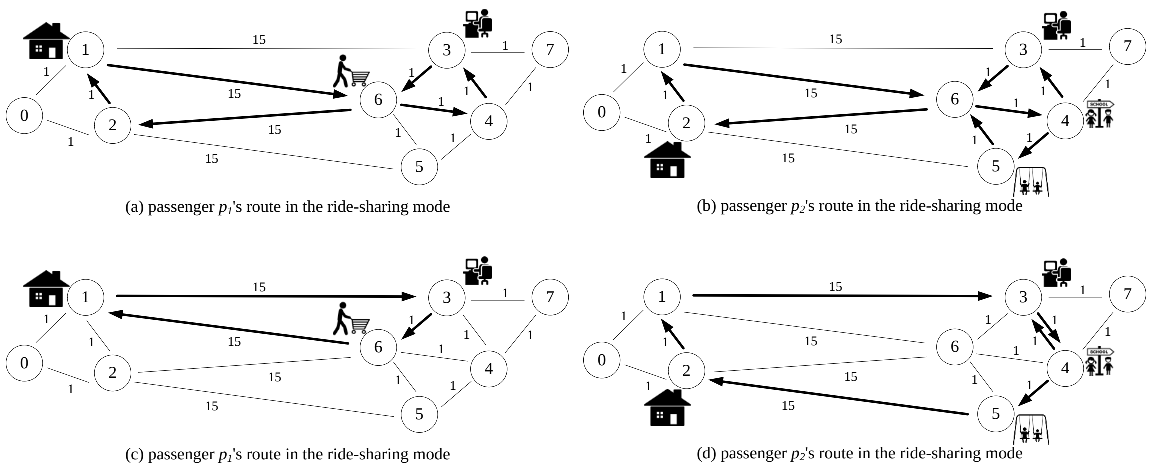

(a) passenger p_1's route in the ride-sharing mode
(b) passenger p_2's route in the ride-sharing mode
(c) passenger p_1's route in the ride-sharing mode
(d) passenger p_2's route in the ride-sharing mode
SCHOOL

Fig 6.4. (a) passenger p_1's route in the ride-sharing mode; (b) passenger p_2's route in the ride-sharing mode; (c) passenger p_1's route if he drives alone; (d) passenger p_2's route if he drives alone.

Table 6.2
State transitions and trips for the aforementioned example

Trips in the morning					
from node i	at time t	at state w	to node i'	at time t'	at state w'
Node 0	7:20 AM	$[\_\ \_\ \_]$	Node 2	7:25 AM	$[\_\ \_\ \_]$
Node 2	7:25 AM	$[\_\ \_\ \_]$	Node 2	7:25 AM	$[r^1_{p_2}\ r^1_{p_2}\ \_]$
Node 2	7:25 AM	$[r^1_{p_2}\ r^1_{p_2}\ \_]$	Node 1	7:30 AM	$[r^1_{p_2}\ r^1_{p_2}\ \_]$
Node 1	7:30 AM	$[r^1_{p_2}\ r^1_{p_2}\ \_]$	Node 1	7:30 AM	$[r^1_{p_2}\ r^1_{p_2}\ r^1_{p_1}]$
Node 1	7:30 AM	$[r^1_{p_2}\ r^1_{p_2}\ r^1_{p_1}]$	Node 6	7:50 AM	$[r^1_{p_2}\ r^1_{p_2}\ r^1_{p_1}]$
Node 6	7:50 AM	$[r^1_{p_2}\ r^1_{p_2}\ r^1_{p_1}]$	Node 4	7:55 AM	$[r^1_{p_2}\ r^1_{p_2}\ r^1_{p_1}]$
Node 4	7:55 AM	$[r^1_{p_2}\ r^1_{p_2}\ r^1_{p_1}]$	Node 4	7:55 AM	$[\_\ r^2_{p_2}\ r^1_{p_1}]$
Node 4	7:55 AM	$[\_\ r^2_{p_2}\ r^1_{p_1}]$	Node 3	8:00 AM	$[\_\ r^2_{p_2}\ r^1_{p_1}]$
Node 3	8:00 AM	$[\_\ r^2_{p_2}\ r^1_{p_1}]$	Node 3	8:00 AM	$[\_\ \_\ \_]$
Node 3	8:00 AM	$[\_\ \_\ \_]$	Node 7	8:05 AM	$[\_\ \_\ \_]$
Trips in the afternoon					
from node i	at time t	at state w	to node i'	at time t'	at state w'
Node 7	3:55 PM	$[\_\ \_\ \_]$	Node 3	4:00 PM	$[\_\ \_\ \_]$
Node 3	4:00 PM	$[\_\ \_\ \_]$	Node 3	4:00 PM	$[r^4_{p_2}\ r^2_{p_1}\ \_]$
Node 3	4:00 PM	$[r^4_{p_2}\ r^2_{p_1}\ \_]$	Node 6	4:05 PM	$[r^4_{p_2}\ r^2_{p_1}\ \_]$
Node 6	4:05 PM	$[r^4_{p_2}\ r^2_{p_1}\ \_]$	Node 6	4:05 PM	$[r^4_{p_2}\ \_\ \_]$
Node 6	4:05 PM	$[r^4_{p_2}\ \_\ \_]$	Node 4	4:10 PM	$[r^4_{p_2}\ \_\ \_]$
Node 4	4:10 PM	$[r^4_{p_2}\ \_\ \_]$	Node 4	4:10 PM	$[r^6_{p_2}\ r^6_{p_2}\ \_]$
Node 4	4:10 PM	$[r^6_{p_2}\ r^6_{p_2}\ \_]$	Node 5	4:15 PM	$[r^6_{p_2}\ r^6_{p_2}\ \_]$
Node 5	4:15 PM	$[r^6_{p_2}\ r^6_{p_2}\ \_]$	Node 5	4:15 PM	$[\_\ \_\ \_]$
Node 5	4:15 PM	$[\_\ \_\ \_]$	Node 5	4:45 PM	$[\_\ \_\ \_]$
Node 5	4:45 PM	$[\_\ \_\ \_]$	Node 5	4:45 PM	$[r^7_{p_2}\ r^7_{p_2}\ \_]$
Node 5	4:45 PM	$[r^7_{p_2}\ r^7_{p_2}\ \_]$	Node 6	4:50 PM	$[r^7_{p_2}\ r^7_{p_2}\ \_]$
Node 6	4:50 PM	$[r^7_{p_2}\ r^7_{p_2}\ \_]$	Node 6	4:50 PM	$[r^7_{p_2}\ r^7_{p_2}\ r^4_{p_1}]$
Node 6	4:50 PM	$[r^7_{p_2}\ r^7_{p_2}\ r^4_{p_1}]$	Node 2	5:10 PM	$[r^7_{p_2}\ r^7_{p_2}\ r^4_{p_1}]$
Node 2	5:10 PM	$[r^7_{p_2}\ r^7_{p_2}\ r^4_{p_1}]$	Node 2	5:10 PM	$[\_\ \_\ r^4_{p_1}]$
Node 2	5:10 PM	$[\_\ \_\ r^4_{p_1}]$	Node 1	5:15 PM	$[\_\ \_\ r^4_{p_1}]$
Node 1	5:15 PM	$[\_\ \_\ r^4_{p_1}]$	Node 1	5:15 PM	$[\_\ \_\ \_]$
Node 1	5:15 PM	$[\_\ \_\ \_]$	Node 0	5:20 PM	$[\_\ \_\ \_]$

By using the ride-sharing mode for the passengers' trip requests, several miles are deducted. Figures 6.4(c) and 6.4(d) illustrate the route of the passengers and Figures 6.5(b) and 6.5(c) depict the corresponding mileage for performing their respective activities by using their own vehicle. In these figures, passenger p_1 travels for 31 miles, whereas p_2 travels for 34 miles. Therefore, if each passenger drives separately, they spend 65 miles in total to perform their activities, while according to Fig. 6.5(a), if they share a ride with each other (for a portion of their tours), the vehicle only travels for 43 miles which is 22 miles less than driving alone.

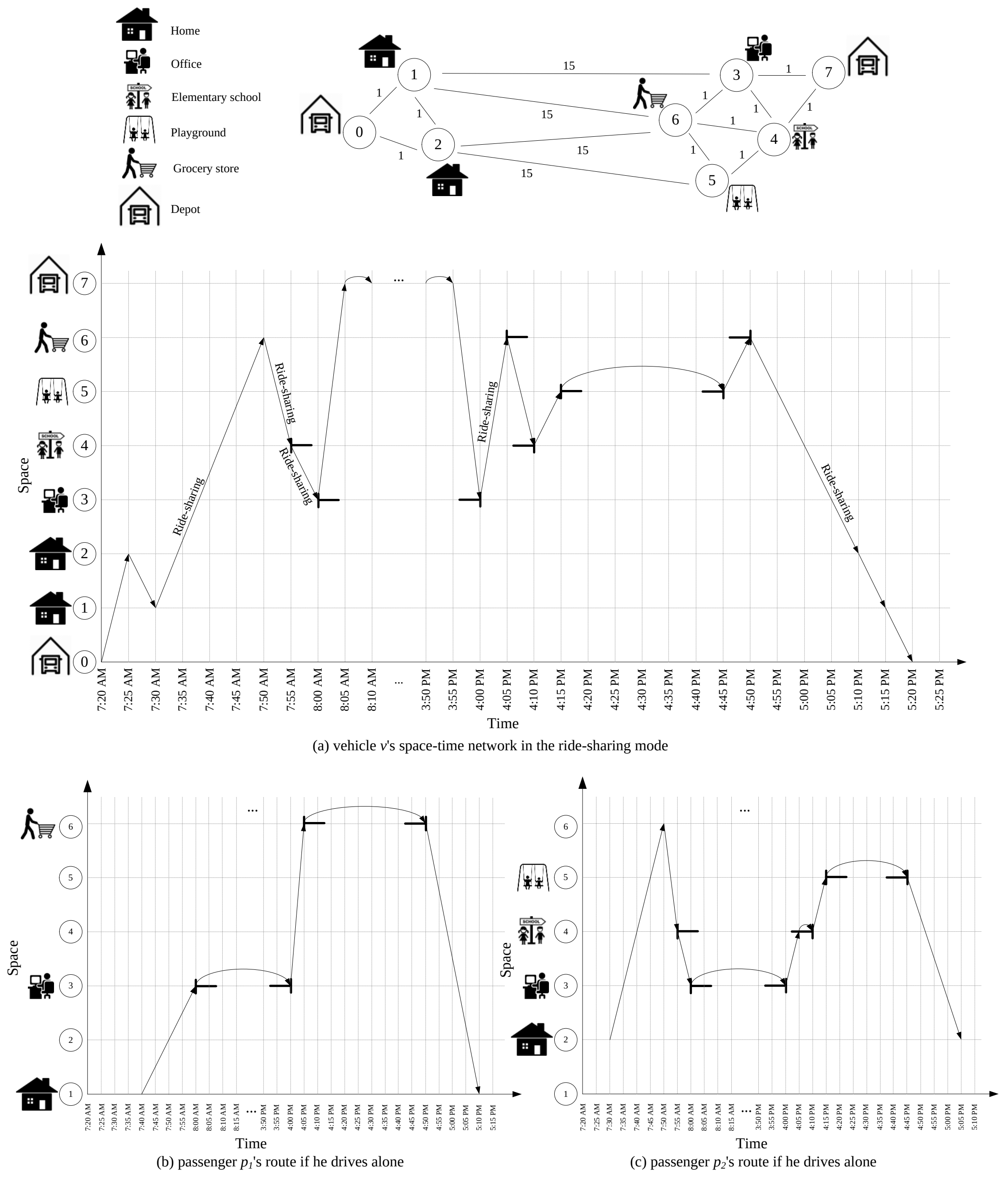

Home
Office
Elementary school
Playground
Grocery store
Depot
15
1
Ride-sharing
Space
Time
(a) vehicle v's space-time network in the ride-sharing mode
(b) passenger p_1's route if he drives alone
(c) passenger p_2's route if he drives alone

Fig. 6.5. (a) vehicle space-time network in the ride-sharing mode; (b) passenger p_1's route when he drives by his own car; (c) passenger p_2's route when he drives by his own car.

6.3. Time-discretized Multi-commodity Network Flow Programming Model

In this section, we initially present the mathematical programming of the proposed time-discretized multi-commodity network flow model. We further apply LR approach to relax the two groups of complicated constraints into the objective function. By doing so, the main problem is systematically decomposed to two sub-problems where sub-problem (1) is a typical least cost path problem and sub-problem (2) is a time-dependent state-dependent least cost path problem. Both sub-problems can be solved by computationally efficient algorithms for solving the shortest path problem, e.g. DP, label correcting algorithm, etc.

6.3.1. Mathematical Model

Mathematical formulation for the proposed time-discretized multi-commodity network flow model is presented in this section. This formulation not only guarantees that each activity (depending on whether it is mandatory or optional) is performed within its time window, but also ensures that the road as well as vehicle capacity constraints are not violated. Note that the roads' capacity constraints can be constructed based on the cumulative arrival and departure functions to reflect detailed traffic congestion propagation through simplified kinematic wave model (Newell, 1993). In this model, we assume that all vehicles have the same planning horizon, i.e. $[0, T]$. Moreover, to distinguish regular transportation nodes from passengers' origin, destination, activities location, as well as vehicles' origin and destination, we add dummy nodes corresponding each to the original transportation network. Each dummy node is only connected to its corresponding transportation node by a link. The travel time on this link can be interpreted as the service time if the added dummy node is related to a passenger's pickup/drop-off, and as preparation time if it is related to a vehicle's departure/arrival at a depot. Without loss of generality, this travel time is assumed to be a unit of time for all passengers and vehicles in this chapter. Table 6.3 lists the notations for the sets, indices, parameters, and variables in this model.

Table 6.3
Sets, Indexes, Parameters, and Variables Used in Our Proposed Model

Symbol	Definition
k	Vehicle index
p	Passenger index
w, w'	Under-service trip requests state indices in vehicles networks
(i, i')	Index of a physical link between adjacent nodes i and i'
$TT(i, i', t)$	Link travel time from node i to node i' starting at time t
(i, t, w), (i', t', w')	Indexes of SST vertices for vehicles SST network
(i, i', t, t', w, w')	Index of a space-time-state arc indicating vehicle v travels from node i at time t with under-service trip requests state w to node i' at time t' with under-service trip requests state w'
a, a'	Passengers' activity indices in passenger p's activities graph
$r(p, a, a')$	Trip request in which passenger p leaves activity location a to perform activity a'
$u(p, a, a')$	Utility gained from serving trip request $r(p, a, a')$
$\mu(i, i', t)$	Maximum road capacity per unit time interval on physical link (i, i') at time t
$\phi(p, a, a', i, i', t, t+1)$	$= 1$, if (i, t) is the dummy vertex from which passenger p calls a vehicle to be picked up for trip request $r(p, a, a')$ (trip $r(p, a, a')$ starts); $= 0$ otherwise
$\psi(p, a, a', i, i', t-1, t)$	$= 1$, if (i', t) is the dummy vertex at which passenger p is dropped off exactly when trip request $r(p, a, a')$ is completed; $= 0$ otherwise
$\Omega(p, a, a', i, t)$	Set of all feasible arcs from dummy vertex (i, t, w) to (i', t, w') in which state w' contains $r(p, a, a')$, while w does not (pickup).
$\Theta(p, a, a', i', t)$	Set of all feasible arcs from $(i, t-1, w)$ to dummy vertex (i', t, w') in which state w contains $r(p, a, a')$, while w' does not (drop-off).
$y(k, i, i', t, t', w, w')$	$= 1$ if arc (i, i', t, t', w, w') is used by vehicle k; $= 0$ otherwise
$x(p, a, a')$	$= 1$ if link (a, a') is traversed by passenger p; $= 0$ otherwise

Note that each vehicle k must start its route from the dummy node corresponding to its origin depot at time $t = 0$ with the null state. We call this vertex as super source vertex(k). In addition, vehicle k must ends its route at the dummy node corresponding to its destination depot at time $t = T$ with the null state. This vertex is called as super sink vertex(k). Finally, this time-discretized multi-commodity network flow problem can be formulated as follows:

$$Max\ \sum_{p,a,a'}[u(p,a,a').x(p,a,a')] \tag{6.1}$$

s.t.

Flow balance constraint at any node belongs to passenger p's activity network:

$$\sum_{a'} x(p,a,a') - \sum_{a'} x(p,a',a) = b \tag{6.2}$$

$b = +1$, if a: passenger p's origin; $b = -1$, if a: passenger p's destination; $b = 0$; otherwise.

Flow balance constraint at any vertex belongs to vehicle k's SST network:

$$\sum_{i',t',w'} y(k,i,i',t,t',w,w') - \sum_{i',t',w'} y(k,i',i,t',t,w',w) = b' \tag{6.3}$$

$b' = +1$, if (i,t,w): super source vertex(k); $b' = -1$, if (i,t,w): super sink vertex(k); $b' = 0$, otherwise.

Coupling constraint to link 'the execution of passenger p's pickup for the trip request $r(p,a,a')$' to 'corresponding pickup arc in vehicle k's SST network':

$$\sum_{k,(w,w')\in\Omega(p,a,a',i,t)} y(k,i,i',t,t+1,w,w') = \phi(p,a,a',i,i',t,t+1).x(p,a,a')$$

$$\forall\phi > 0 \tag{6.4}$$

Coupling constraint to link 'the execution of passenger p's drop-off exactly when trip request $r(p,a,a')$ is completed' to 'corresponding delivery arc in vehicle k's SST network:

$\sum_{k,(w,w')\in\Theta(p,a,a',i',t)} y(k,i,i',t-1,t,w,w') = \psi(p,a,a',i,i',t-1,t).x(p,a,a')$

$\forall\psi > 0$ (6.5)

Conceptual road outflow capacity constraint:

$\sum_{k}\sum_{t',w,w'} y(k,i,i',t,t',w,w') \leq \mu(i,i',t) \qquad \forall(i,i');\ t\in[0,T-1]$ (6.6)

Binary definitional constraint:

$x(p,a,a') \in \{0,1\} \qquad \forall p,a,a'$ (6.7)

$y(k,i,i',t,t',w,w') \in \{0,1\} \qquad \forall k,i,i',t,t',w,w'$ (6.8)

6.3.2. LR-based Solution Approach

Defining multi-dimensional decision variables $y(k,i',i',t',t',w,w')$ leads to computational challenges for the real-world data sets, which are addressed properly by specialized programs and an innovative solution framework. We reformulate the problem by relaxing the complex set of constraints (6.4), (6.5), and (6.6) into the objective function and introducing the corresponding Lagrangian multipliers, $\alpha(p,a,a')$, $\beta(p,a,a')$, and $\gamma(i,i',t)$ to construct the dualized Lagrangian function (6.9).

$L = Min\ \sum_{p,a,a'}[-u(p,a,a').x(p,a,a')] +$

$\sum_{p,a,a'}\alpha(p,a,a').[\sum_{k,(w,w')\in\Omega(p,a,a',i,t)} y(k,i,i',t,t+1,w,w') - \phi(p,a,a',i,i',t,t+$

$1).x(p,a,a')] + \sum_{p,a,a'}\beta(p,a,a').[\sum_{k,(w,w')\in\Theta(p,a,a',i',t)} y(k,i,i',t-1,t,w,w') -$

$\psi(p,a,a',i,i',t-1,t).x(p,a,a')] +$

$\sum_{i,i',t}\gamma(i,i',t).\left[\sum_{k}\sum_{t',w,w'} y(k,i',i',t',t',w,w') - \mu(i,i',t)\right]$ (6.9)

Based on a Lagrangian reformulation framework, the main problem can be transformed to two easy sub-problems, P_x and P_y, which can be solved independently with much computationally efficient effort (*15*).

Sub-problem P_x (6.10)
$Min\ (-U - A\Phi - B\Psi)X$
s.t.
Constraints (6.2) & (6.7)

Sub-problem P_y (6.11)
$Min\ (A + B + \Gamma)Y - \Gamma M$
s.t.
Constraints (6.3) & (6.8)

Sub-problem P_x is a typical least cost path problem, and P_y is a time-dependent state-dependent least cost path problem. Both sub-problems can be solved by computationally efficient algorithms, e.g. DP, label correcting algorithm, etc. In this research, we apply time-dependent state-dependent forward DP to solve these two sub-problems.

If individuals are provided with a price (in the form of lagrangian multipliers α and β) to undertake a trip, they can determine whether or not to engage in a discretionary activity based on the prevailing ‘price’ for the trip to get to that activity. In Section 6.4, the results from the application of the proposed algorithm on Phoenix subarea transportation network are provided.

6.4. Computational Experiments

The time-dependent state-dependent forward DP described in this chapter is coded in C++ platforms. The experiments were performed on an Intel Workstation running two

Xeon E5-2680 processors clocked at 2.80 GHz with 20 cores and 192 GB RAM running Windows Server 2008 x64 Edition. In addition, parallel computing and OpenMP technique are implemented for generating lower bound and upper bound at each iteration in the LR algorithm.

In this section, we examine our proposed model on sample data sets from the Phoenix subarea with 1162 transportation nodes and 3164 links, illustrated in Fig. 6.6, to demonstrate the computational efficiency, as well as, solution optimality of the proposed algorithm. Some sample trip requests (an OD pairs) are illustrated by directed dashed links, whereas vehicles' paths are shown by directed thick links with different colors from transportation links color.

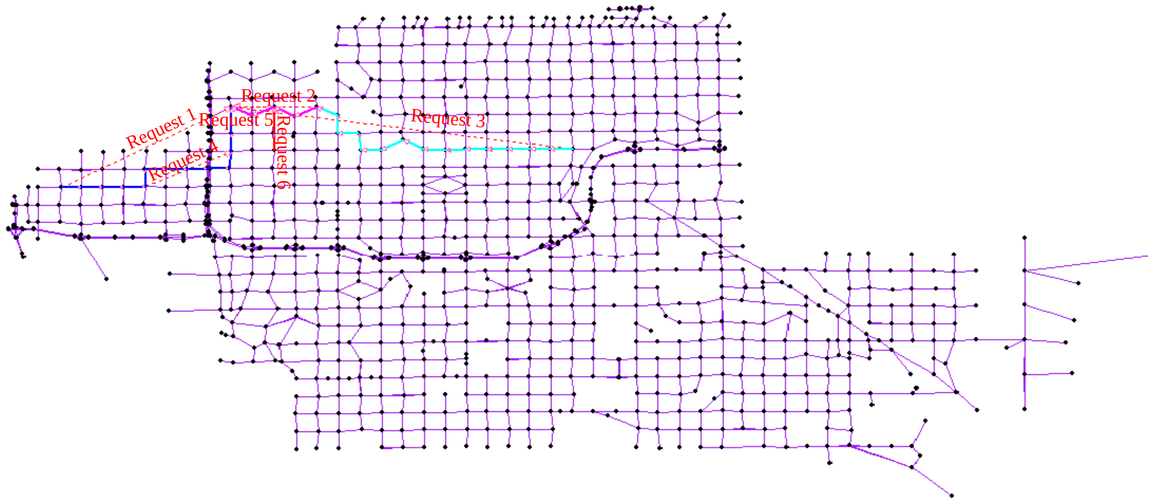


Fig. 6.6. Phoenix Metropolitan Chandler subarea transportation network with 6 trip requests.

It is assumed that the routing cost of a transportation arc is $22/h, while the waiting cost at a node is $15/h. The initial charge is assumed to be $7 for all passengers. The maximum capacity of the vehicles for service is 2 seats, and the length of time horizon is 2 hours (120 min). It is also assumed that a unit of time has 1 min length. The proposed multi-commodity flow programming model is first demonstrated by the general purpose optimization package GAMS (Rosenthal, 2015) in small transportation networks. For large-scale applications, we also create a time-dependent state-dependent shortest path computational engine by enhancing an open-source mesoscopic dynamic traffic assignment model namely DTALite (Zhou and Taylor, 2014). The resulting open-source project with GAMS and C++ source codes can be found at https://github.com/xzhou99/Agent-Plus.

Trip requests on the test network are generated from an open-source activity based travel demand modeling system called Open Source Activity Mobility Simulator (OpenAMOS) (Pendyala et al. 2012). We also pre-specify the locations of vehicle depots at major activity locations in the test network. Table 6.4 presents the summary of vehicle routing results for a few test cases, with the following observations. If the number of trip requests increases, we generally need more vehicles to satisfy the travel demand desires. The ratio of required fleet size and total request number dramatically varies, between about 20% and 66%, from depending on the underlying spatial and temporal patterns. Typically, a larger pool of travel requests could lead to better system vehicle use efficiency. Due to the fixed vehicle depot and time-window restrictions, there are still a few under-served trip requests in the given passenger activity-travel pattern. Different from commonly used heuristic algorithms, the developed algorithm aims to find the exact or close-to-optimal solution to the proposed optimization model. The desirable trip-to-vehicle assignment and detailed routing solution could take about 5 min to compute for a medium case with about 50 trip requests.

Table 6.4
Results for the Phoenix subarea with 1162 nodes and 3164 links

Test case	Number of trip requests	Number of vehicles required	Number of passengers not served	Running time (s)
1	6	4	0	8.41
2	10	5	0	25.5
3	15	10	1	51.7
4	37	20	1	255.9
5	48	9	0	308.2

We also explain the pricing mechanism by test case 1 with 6 trip requests. Fig. 6.7 demonstrates the Lagrangian multiplier corresponding each trip request along 5 iterations. According to Fig. 6.7, each multiplier ultimately converges to a specific value. This value can be literally interpreted as the ‘ultimate price’ of a trip to get to a specific travel activity. Through this pricing mechanism, vehicle service providers would be able to offer a reasonable bid to their customers.

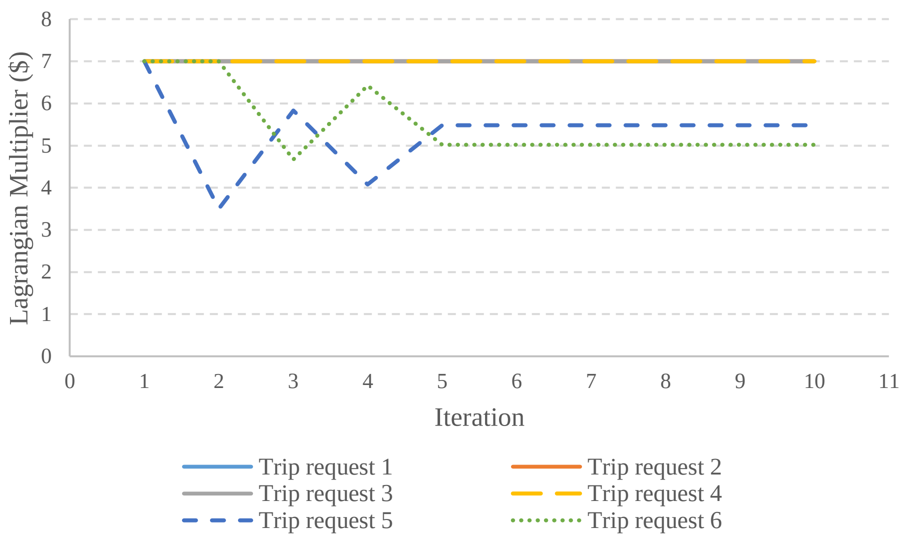


Fig. 6.7. Lagrangian multipliers along 10 iterations in test case 1.

6.5. Conclusions

Despite all advancements in the real-time traffic control, DTA modelers still seek for a robust framework to extend their existing model (1) from single-OD demand to trip chaining, and (2) from driving your own mode to shared-use vehicle systems. In this research, it is expected that with the help of the proposed algorithm, it would be possible to provide a much richer set of information to ABMs, thus enabling ABMs to include ride-sharing/ride-hailing as an additional mode of travel.

Future research directions include (1) how to schedule activity-travel requests at extremely large scales to meet temporally and spatially distributed traveler demand, (2) how to seamlessly integrate distributed computing, car platooning, and resource-oriented pricing and scheduling for better coordinated use of vehicles and road infrastructure resources. We hope this research line could offer a set of novel techniques on holistic behaviorally oriented traveler mobility optimization under the new environment of shared self-driving car networks.

CHAPTER 7

CONCLUSION

In Chapter 1, we introduce the pickup and delivery problem and extensions of this problem which have been studied in this dissertation. The objectives of this dissertation as well as overview of our proposed methods have been extensively discussed in this chapter.

Chapter 2 provides a comprehensive literature review for the pickup and delivery problem. In this chapter, we focus on the applications of this problem, as well as solution methods used to solve this optimization problem. In this chapter, we aim to emphasize that previous research has made a number of important contributions to the challenging pickup and delivery problem along different formulation or solution approaches. However, there are a number of modeling and algorithmic challenges for a large-scale deployment of a vehicle routing and scheduling algorithm, especially for regional networks with various road capacity and traffic delay constraints on freeway bottlenecks and signal timing on urban streets.

Chapter 3 first proposes a new time-discretized multi-commodity network flow model for the VRPPDTW based on the integration of vehicles' carrying states within space-time transportation networks, so as to allow a joint optimization of passenger-to-vehicle assignment and turn-by-turn routing in congested transportation networks. Our three-dimensional state-space-time network construct is able to comprehensively enumerate possible transportation states at any given time along vehicle space-time paths, and further allows a forward dynamic programming solution algorithm to solve the single vehicle VRPPDTW problem. By utilizing a Lagrangian relaxation approach, the primal multi-vehicle routing problem is decomposed to a sequence of single vehicle routing sub-problems, with Lagrangian multipliers for individual passengers' requests being updated by sub-gradient-based algorithms. We further discuss a number of search space reduction strategies and test our algorithms, implemented through a specialized program in C++, on medium-scale and large-scale transportation networks, namely the Chicago sketch and Phoenix regional networks.

By reformulating the PDPTW through space-time networks to consider time window requirements, our proposed approach can not only solve the vehicle routing and scheduling problem directly in large-scale transportation networks with time-dependent congestion, but also avoid the complex procedure to eliminate any sub-tour possibly existing in the optimal solution for many existing formulations. By further introducing virtual vehicle constructs, the proposed approach can fully incorporate the full set of interacting factors between passenger demand and limited vehicle capacity in this model to derive feasible solutions and practically important system-wide cost-benefit estimates for each request through a sub-gradient-based pricing method. This joint optimization and pricing procedure can assist transportation network service providers to quantify the operating costs of spatially and temporally distributed trip requests.

On a large-scale regional network, the capacity impact of optimized passenger-to-vehicle matching results can be further evaluated in mesoscopic dynamic traffic simulation packages such as an open-source Dynamic Traffic Assignment-Light weight

(DTALite) (Zhou and Taylor, 2014). Future work will concentrate on the development of the model for the following cases: (i) Passengers may desire different ride-sharing capacities (i.e. a passenger may desire to share his ride with up to only one passenger, whereas the other passenger may have no restriction about the number of passengers which share their ride with him). (ii) A passenger may desire to be or not to be served by a particular vehicle. (iii) A transportation request could contain a group of passengers who have the same origin, while they may or may not have the same destination. Alternatively, a transportation request could contain a group of passengers who have the same destination, while they may or may not have the same origin. In this case, we are interested in adding dummy nodes corresponding to passengers' origins and destinations more wisely and efficiently. In addition, in our future research, a comprehensive branch-and-bound algorithm should be included in our solution framework to fully address the complexity of assigning different vehicles to multiple passengers.

In Chapter 3, we focus on the pickup and delivery problem with time windows and synchronized transfers which is a challenging version of the vehicle routing problem. In order to tackle this tough problem, we add time dimension to physical transportation networks to not only track the location of vehicles at any time, but also consider passengers' preferred pickup and delivery time windows, synchronization time points, and precedence constraints to the problem. We also add another dimension so called "cumulative service state" to the constructed space-time graph to track the service status of requests at any time. The constructed hyper-network not only handles real-life transportation networks with time-dependent and load-dependent costs, but also is wellsuited for connecting microscopic cumulative service states to macroscopic cumulative flow count diagrams. We develop a continuous time approximation approach using cumulative arrival, departure, and on-board count diagrams to effectively assess the dynamic system performance and guide the search. In order to handle a large set of passengers, we develop the traditional cluster-first, route-second approach. We break the large-sized primary problem into a number of small-sized sub-problems in which the most compatible origin-destination pairs are clustered together. Then, we reach optimality (more precisely, pseudo-optimality due to the time discretization) for local clusters derived from a large set of passengers. Finally, we find the optimal chain of work piece which can be done by vehicles to improve the vehicles' efficiency. We perform the extensive experiments over the standard instances applied by Ropke and Pisinger (2006) and real-world large scale data set proposed by Cainiao network with about 10,000 delivery orders, to examine the effectiveness and computational efficiency of our developed algorithm.

Future work may concentrate on building a computational engine to establish a wrapper for the dynamic programming algorithms with the inputs of a transportation network and possible state transition matrixes, and the output of various vehicle-path assignment and routing solutions. Another interesting extension of our state-space-time framework can be building a more practically useful and robust model with some levels of travelers/carriers' behavior in better passengers' and vehicles' clustering, as oppose to simple and efficient trade-off between time and distance.

Accessibility is the ease of obtaining desired destinations, activities, or services in an environment. A common accessibility measure in basic and applied transportation science is the space-time prism (STP) and the network-time prisms (NTPs): these are the envelopes of all possible paths between two locations and times in planar space and transportation networks, respectively. STPs and NTPs focus on time as the scarce resource limiting accessibility. However, other resource constraints can constrain space-time accessibility, such as limits or "budgets" for energy, emissions, or monetary expenses. Chapter 4 extends NTPs to include other resource constraints in addition to time. Network-based *resource hyper-prisms* (RHPs) incorporate other resource constraints into NTP, capturing the trade-offs between time and other resources in determining space-time accessibility. We conceptualize RHPs as a constrained optimization problem and develop a forward and backward resource-dependent time-dependent dynamic programming to determine the boundaries of a RHP given time and other resource budgets. We illustrate our approach using the Chicago sketch network (with 933 nodes and 2,967 links) for a use case of an individual with a gasoline passenger car with limited carbon emission budget and using parts of Washington, D.C. and Baltimore networks (with 12,145 nodes and 30,697 links) for a use case of an individual with an electric vehicle (EV) to study regional accessibility with charging stations along highways or in towns to show how a RHP can measure the accessibility impacts of new EV charging stations. The current work could be extended by considering different types of resources, particularly for the transportation mode, by which we can evaluate the effect of multi-modal transportation options on the accessibility of the travelers.

It is vital to have integrated model systems that fully capture the interactions between supply and demand dimensions of travel to model the implications of advanced technologies and mobility services on traveler behavior. In Chapter 6, we introduce a new state dimension (called the ‘under-service trip request’ state) to the vehicle scheduling model in order to track the execution status of the trip requests at any time and transportation node. We also construct activity-travel graphs for passengers to detect the execution of the passenger’s activities. We further propose a time-discretized multi-commodity network flow model that not only guarantees that each activity request is systematically evaluated within its time window (depending on whether it is mandatory or optional), but also ensures that the road as well as vehicle capacity constraints are not violated. By introducing a mapping constraint between ‘passenger’s pickup/drop-off at an activity location’ and ‘under-service trip requests state in a vehicle network’ as linking constraints, passenger and vehicle networks can be seamlessly connected together. By dualizing this set of trip request constraints and the road capacity constraints into the objective function and utilizing a Lagrangian relaxation approach, the main problem is decomposed to two sub-problems which can be solved in parallel through computationally efficient algorithms for real-world transportation networks. Based on a standard optimization solver and C++, we developed an open-source activity-based vehicle routing engine, namely Agent+, using real-world Phoenix subarea network data sets and trip requests generated from activity-based modelling system OpenAMOS.

Despite all advancements in the real-time traffic control, DTA modelers still seek for a robust framework to extend their existing model (1) from single-OD demand to trip chaining, and (2) from driving your own mode to shared-use vehicle systems. In this research, it is expected that with the help of the proposed algorithm, it would be possible to provide a much richer set of information to ABMs, thus enabling ABMs to include ride-sharing/ride-hailing as an additional mode of travel.

Future research directions include (1) how to schedule activity-travel requests at extremely large scales to meet temporally and spatially distributed traveler demand, (2) how to seamlessly integrate distributed computing, car platooning, and resource-oriented pricing and scheduling for better coordinated use of vehicles and road infrastructure resources. We hope this research line could offer a set of novel techniques on holistic behaviorally oriented traveler mobility optimization under the new environment of shared self-driving car networks.

REFERENCES

Adler, J. D., Mirchandani, P. B., Xue, G., and Xia, M. (2016) The electric vehicle shortest-walk problem with battery exchanges. *Networks and Spatial Economics*, 16(1), 155–173.

Ahn, K. (1998) Microscopic Fuel Consumption and Emission Modeling. Master Thesis. Virginia Polytechnic Institute and State University, Virginia, United States.

Ai, T. J. and Kachitvichyanukul, V. (2009) A particle swarm optimization for the vehicle routing problem with simultaneous pickup and delivery, *Computers & Operations Research*, 36, 1693-1702.

Alfa, A. S. (1986) Scheduling of vehicles for transportation of elderly. *Transportation Planning and Technology*, 11, 203-212.

An, F. and Ross, M. (1993) Model of fuel economy with applications to driving cycles and traffic management. *Transportation Research Record*, 1416, 105–114.

Angelelli, E. and Mansini, R. (2002) The vehicle routing problem with time windows and simultaneous pick-up and delivery. In: Klose A, Speranza MG, VanWassenhove LN (eds) Quantitative Approaches to Distribution Logistics and Supply Chain Management. Springer, Berlin-Heidelberg, 249-267.

Attanasio, A., Cordeau, J. F., Ghiani, G., and Laporte, G. (2004) Parallel Tabu search heuristic for the dynamic multi-vehicle dial-a-ride problem, *Parallel Computing*, 30, 377-387.

Auld, J., Hope, M., Ley, H., Sokolov, V., Xu, B., and Zhang, K. (2016) POLARIS: Agent-based modeling framework development and implementation for integrated travel demand and network and operations simulations. *Transportation Research Part C: Emerging Technologies*, *64*, 101-116.

Azadian, F., Murat, A. E., Chinnam, R. B. (2012) Dynamic routing of time-sensitive air cargo using real-time information, *Transportation Research Part E: Logistics and Transportation Review*, 48 (1), 355-372.

Azi, N., Gendreau, M., Potvin, J.-Y. (2007) An exact algorithm for a single vehicle routing problem with time windows and multiple routes. *European Journal of Operational Research*, 178(3), 755-766.

Baker, M. A. (1994) Fuel consumption and emission models for evaluating traffic control and route Guidance strategies. Master Thesis. Queen's University, Kingston, Canada.

Baldacci, R., Bartolini, E., and Mingozzi, A. (2011) An exact algorithm for the pickup and delivery problem with time windows. *Operations Research*, 59(2) 414-426.

Balmer, M., Rieser, M., Meister, K., Charypar, D., Lefebvre, N., Nagel, K., and Axhausen, K. (2009) MATSim-T: Architecture and Simulation Times. *Multi-agent systems for traffic and transportation engineering*, 57-78.

Banister, D. (2008) The sustainable mobility paradigm. *Transport policy*, 15(2), 73–80.

Bard, J.F. and Jarrah, A. I. (2009) Large-scale constrained clustering for rationalizing pickup and delivery operations, *Transportation Research Part B: Methodological*, 43(5), 542-561.

Bartholomew, K. (2009) Cities and accessibility: The potential for carbon reductions and the need for national leadership. *Fordham Urban Law Journal* 36(2): 159–209.

Baugh, J., Kakivaya, G., and Stone, J. R. (1998) Intractability of the dial-a-ride problem and a multi-objective solution using simulated annealing. *Engineering Optimization*, 30, 91-123.

Bell, W., Dalberto, L., Fisher, M. L., Greenfield, A., Jaikumar, R., Kedia, P., Mack, R., and Prutzman, P. (1983) Improving the distribution of industrial gases with an on-line computerized routing and scheduling optimizer. *Interfaces*, 13, 4-23.

Bellman, R. (1962) Dynamic programming treatment of the traveling salesman problem. *Journal of the Association for Computing Machinery*, 9, 61-63.

Bent, R. and Henteryck, P. V. (2006) A two-stage hybrid algorithm for pickup and delivery vehicle routing problems with time windows, *Computers & Operations Research*, 33, 875-893.

Bianchessi, N. and Righini, G. (2007) Heuristic algorithms for the vehicle routing problem with simultaneous pick-up and delivery. *Computers & Operations Research*, 34(2), 578-594.

Bianco, L., Mingozzi, A., Ricciardelli, S., and Spadoni, M. (1994) Exact and heuristic procedures for the traveling salesman problem with precedence constraints, based on dynamic programming. *INFOR*, 32, 19-31.

Borndörfer, R., Klostermeier, F., Grötschel, M., and Küttner, C. (1997) Telebus Berlin: vehicle scheduling in a dial-a-ride system, Technical report SC 97-23, Konrad-Zuse-Zentrum für Informationstechnik, Berlin.

Bramel, J., and Simchi-Levi, D. (1995) A location based heuristic for general routing problems. *Operations Research*, 43, 649-660.

Brownstone, D. and Small, K. A. (2005) Valuing time and reliability: assessing the evidence from road pricing demonstrations. *Transportation Research Part A: Policy and Practice*, 39(4), 279–293.

Bodin, L.D. and Sexton, T. (1986) The multi-vehicle subscriber dial-a-ride problem. *TIMS Studies in Management Science*, 26, 73-86.

Boland, N., Hewitt, M., Marshall, L., Savelsbergh, M.W.P. (2017a) The continuous time service network design problem. *Operations Research*, article in advance, https://doi.org/10.1287/opre.2017.1624.

Boland, N., Hewitt, M., Vu, D.M., Savelsbergh, M.W.P. (2017b) Solving the traveling salesman problem with time windows through dynamically generated time-expanded networks. In *International Conference on AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems*, 254-262, Springer, Cham.

Burns, L. D. (1979) *Transportation, Temporal, and Spatial Components of Accessibility* (Lexington, MA: D. C. Heath).

Caris, A. and Janssens, G.K. (2009) A local search heuristic for the pre- and end-haulage of intermodal container terminals. *Computers & Operations Research*, 36, 2763-2772.

Carrabs, F., Cerulli, R., and Cordeau, J.-F. (2007) An additive branch-and-bound algorithm for the pickup and delivery traveling salesman problem with LIFO or FIFO loading. *INFOR*, 45, 223-238.

Chabini, I. (1998) Discrete dynamic shortest path problems in transportation applications: Complexity and algorithms with optimal run time. *Transportation Research Record: Journal of the Transportation Research Board*, 1645, 170-175.

Chan, C. C. (2007) The state of the art of electric, hybrid, and fuel cell vehicles. *Proceedings of the IEEE*, *95*(4), 704–718.

Chandra, R., Menon, R., Dagum, L., Kohr, D., Maydan, D., McDonald, J. (2000) *Parallel Programming in OpenMP*. Morgan Kaufmann.

Chemla, D., Meunier, F., and Wolfler Calvo, R. (2013). Bike sharing systems: solving the static rebalancing problem, *Discrete Optimization*, 10(2), 120-146.

Chen, M. C., Hsiao, Y. H., Reddy, R. H., Tiwari, M. K. (2016) The self-learning particle swarm optimization approach for routing pickup and delivery of multiple products with material handling in multiple cross-docks. *Transportation Research Part E: Logistics and Transportation Review*, 91, 208-226.

Chen, D., Khan, M., and Kockelman, K. M. (2013) The electric vehicle charging station location problem: a parking based assignment method for Seattle. In: Presented at 92nd Annual Meeting of the Transportation Research Board in Washington DC.

Chen, J. F. and Wu, T. H. (2005) Vehicle routing problem with simultaneous deliveries and pickups. *Journal of the Operational Research Society*, 57(5), 579-587.

Chen, A., Yang, H., Lo, H. K., and Tang, W. H. (2002) Capacity reliability of a road network: an assessment methodology and numerical results. *Transportation Research Part B: Methodological*, *36*(3), 225–252.

Cherkesly, M., Desaulniers, G., Irnich, S., and Laporte, G. (2016) Branch-price-and-cut algorithms for the pickup and delivery problem with time windows and multiple stacks. *European Journal of Operations Research*, 250(3), 782-793.

Cherkesly, M., Desaulniers, G., Laporte, G. (2014) A population-based metaheuristic for the pickup and delivery problem with time windows and lifo loading. Technical report, Les Cahiers du GERAD, G-2014-66, GERAD, Montréal.

Christiansen, M. (1999) Decomposition of a combined inventory routing and time constrained ship routing problem. *Transportation Science*, 33, 3-16.

Christiansen, M., Fagerholt, K., Nygreen, B., and Ronen, D. (2007) Maritime transportation, In: Barnhart, C. and Laporte, G. (eds), *Handbooks in Operations Research and Management Science: Transportation*, North-Holland, Amsterdam, the Netherlands, 14, 189-284.

Christiansen, M., Fagerholt, K., and Ronen, D. (2004) Ship routing and scheduling: status and perspectives, *Transportation Science*, 38(1): 1-18.

Christiansen, M. and Nygreen, B. (1998) A method for solving ship routing problems with inventory constraints. *Annals of Operations Research*, 81, 357-378.

Chu, S. and Majumdar, A. (2012) Opportunities and challenges for a sustainable energy future. *Nature*, *488*(7411), 294.

Clarke, G. and Wright, J. W. (1964) Scheduling of vehicles from a central depot to a number of delivery points, *Operations Research*, 12, 568-581.

Colorni, A., and Righini, G. (2001) Modeling and optimizing dynamic dial-a-ride problems. *International Transactions in Operational Research*, 8, 155-166.

Coltin, B. and Veloso, M. (2014) Online pickup and delivery planning with transfers for mobile robots, *IEEE International Conference on Robotics and Automation (ICRA)*, 5786-5791.

Contardo, C., Hemmelmayr, V., and Crainic, T.G. (2012) Lower and upper bounds for the two-echelon capacitated location-routing problem, *Computers & Operations Research*, 39(12), 3185-3199.

Cordeau, J.-F. (2006) A branch-and-cut algorithm for the dial-a-ride problem. *Operations Research*, 54(3): 573-586.

Cordeau, J.-F., Iori, M., Laporte, G., and Gonzál, J. J. S. (2010) A branch-and-cut algorithm for the pickup and delivery traveling salesman problem with LIFO loading, Networks, 55 (1), 46-59.

Cordeau, J.F. and Laporte, G. (2003). A tabu search heuristic for the static multi-vehicle dial-a-ride problem, *Transportation Research Part B*, 37, 579-594.

Cordeau, J.-F., Laporte, G., Potvin, J.-Y., and Savelsbergh, M.W.P. (2007) Transportation on demand. In C. Barnhart and G. Laporte, editors, Transportation, *Handbooks in Operations Research and Management Science*, Volume 14, pages 429-466. Elsevier, Amsterdam.

Cortés, C., Matamala, M., and Contardo, C. (2010) The pickup and delivery problem with transfers: Formulation and a branch-and-cut solution method. *European Journal of Operational Research*, 200(3), 711-724.

Coslovich, L., Pesenti, R., and Ukovich, W. (2006) A two-phase insertion technique of unexpected customers for a dynamic dial-a-ride problem. *European Journal of Operational Research*, 175, 1605-1615.

Côté, J., Archetti, C., Speranza, M., Gendreau, J., and Potvin, J.-Y. (2012) A branch-and-cut algorithm for the pickup and delivery traveling salesman problem with multiple stacks. *Networks* 60(4), 212-226.

Crispim, J. and Brandao, J. (2005) Metaheuristics applied to mixed and simultaneous extensions of vehicle routing problems with backhauls. *Journal of the Operational Research Society*, 56(11), 1296-1302.

Cullen, F., Jarvis, J., and Ratliff, D. (1981) Set partitioning based heuristics for interactive routing. *Networks*, 11, 125-144.

Daganzo, C.F. (2001) A theory of supply chains. 526 lecture notes in economics and mathematical systems, *Springer.*

Dantzig, G. B. and Ramser, J. H. (1959) The truck dispatching problem, *Management Science*, 6, 80-91.

Dashora, Y., Barnes, J., Combs, R. P. T., Hilliard, M., and Chinthavali, M. (2010) The PHEV charging infrastructure planning (PCIP) problem. *International Journal of Emerging Electric Power Systems*, 11 (2).

Davis S. C., Diegel, S. W., and Boundy, R. G. (2011) Transportation Energy Data Book: Edition 30. US Department of Energy.

Dell'Amico, M., Righini, G., and Salani M. (2006) A branch-and-price approach to the vehicle routing problem with simultaneous distribution and collection. *Transportation Science*, 40, 235-247.

Deng, A. M., Mau, C., and Zhou, Y.T. (2009) Optimizing research of an improved simulated annealing algorithm to soft time windows vehicle routing problem with pick-up and delivery, *Systems Engineering-Theory and Practice*. 29 (5), 186-192.

Desaulniers, G., Desrosiers, J., Erdmann, A., Solomon, M. M., and Soumis. F. (2002) The VRP with pickup and delivery. In Toth, P. and Vigo, D. editors, The Vehicle Routing Problem, *SIAM Monographs on Discrete Mathematics and Applications*, Chapter 9, SIAM, Philadelphia, 225-242.

Desrosiers, J., Dumas, Y., and Soumis, F. (1986) A dynamic programming solution of the large-scale single-vehicle dial-a-ride problem with time windows. *American Journal of Mathematical and Management Sciences*, 6, 301-325.

Desrosiers, J., Dumas, Y., and Soumis, F. (1988) The multiple vehicle dial-a-ride problem, in J.R. Daduna, A. Wren (eds.) Computer Aided Transit Scheduling, Proceedings of the Fourth International Workshop on Computer-Aided Scheduling of Public Transport, Lecture Notes in Economics and Mathematical Systems 308, Springer Verlag, 15-27.

Desrosiers, J., Dumas, Y., Soumis, F., Taillefer, S., and Villeneuve, D. (1991) An algorithm for mini-clustering in handicapped transport. Les Cahiers du GERAD, G-91-02, HEC Montréal.

Dethloff, J. (2002) Relation between vehicle routing problems: an insertion heuristic for the vehicle routing problem with simultaneous delivery and pick-up applied to the vehicle routing problem with backhauls, *Journal of the Operational Research Society*, 53, 115-118.

Diana, M., and Dessouky, M. M. (2004) A new regret insertion heuristic for solving large-scale dial-a-ride problems with time windows. *Transportation Research Part B: Methodological*, 38, 539-557.

Dong, X., Ben–Akiva, M., Bowman, J., and Walker, J. (2006) Moving from trip–based to activity–based measures of accessibility. *Transportation Research A*, 4(2), 163–180.

Drexl, M. (2012a) Synchronization in vehicle routing - A survey of VRPs with multiple synchronization constraints. *Transportation Science*, 297-316.

Drexl, M. (2012b) Rich vehicle routing in theory and practice. *Logistics Research*, 5, 47-63.

Drexl, M. (2013) Applications of the vehicle routing problem with trailers and transshipments. *European Journal of Operational Research*, 227, 275-283.

Dumas, Y., Desrosiers, J., and Soumis, F. (1989) Large scale multi-vehicle dial-a-ride problems. Les Cahiers du GERAD, G-89-30, HEC Montréal.

Dumas, Y., Desrosiers, J., and Soumis, F. (1991) The Pickup and Delivery Problem with Time Windows. *European Journal of Operations Research*, 54, 7-22.

Dumitrescu, I., Ropke, S., Cordeau, J.-F., and Laporte, G. (2010) The traveling salesman problem with pickup and delivery: polyhedral results and a branch-and-cut algorithm, *Mathematical Programming*, 121, 269-305.

Edie, L.C. and Foote, R.S. (1960) Effect of shock waves on tunnel traffic flow. *Proceedings of the Highway Research Board*, 39, 492-505.

Electric Vehicle Operation Program, 1999. Performance comparison of Southern California Edison electric fleet vehicle. Technical Report, US Department of Energy.

Erdogan, G., Laporte, G., and Wolfler, Calvo R. (2014) The static bicycle relocation problem with demand intervals, *European Journal of Operational Research*, 238, 451-457.

Ettema, D. and Timmermans, H. (2007) Space-time accessibility under conditions of uncertain travel times: Theory and numerical simulations. *Geographical Analysis*, 39, 1538–4632.

Fisher, M. L., Greenfield, A., Jaikumar, R., and Lester, J. (1982) A computerized vehicle routing application. *Interfaces*, 12, 42-52.

Fisher, M. L., Jörnsten, K.O., and Madsen, O. B. G. (1997) Vehicle routing with time windows: Two optimization algorithms. *Operations Research*, 45, 488-492.

Fisher, M. L. and Rosenwein, M. B. (1989) An interactive optimization system for bulk-cargo ship scheduling. *Naval Research Logistic Quarterly*, 35, 27-42.

Frade, I., Ribeiro, A., Goncalves, G.A., and Antunes, A. P. (2011) Optimal location of charging stations for electric vehicles in a neighborhood in Lisbon, Portugal. *Transportation Research Record*: Journal of the Transportation Research Board 2252, 91–98.

Frey, H. C. and Liu, B. (2013) Development and Evaluation of a Simplified Version of MOVES for Coupling with a Traffic Simulation Model. Proceedings, 91st Annual Meeting of the *Transportation Research Board*, Washington D.C.

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

Gajpal, Y. and Abad, P. (2009) An ant colony system (ACS) for vehicle routing problem with simultaneous delivery and pickup, Computers & Operations Research, 36(12), 3215-3223.

Gazis D. C. and Potts R. B. (1963) The oversaturated intersection. *Proceedings of the 2nd International Symposium on the Theory of Road Traffic Flow*, 221-237.

Gendreau, M., Guertin, F., Potvin, J.-Y., and Séguin, R. (1998) Neighborhood search heuristics for a dynamic vehicle dispatching problem with pick-ups and deliveries. Technical Report CRT-98-10, Centre de recherche sur les transports, Université de Montréal, Canada.

Gendreau, M., Guertin, F., Potvin, J. Y., and Séguin, R. (2006) Neighborhood search heuristics for a dynamic vehicle dispatching problem with pick-ups and deliveries, *Transportation Research Part C: Emerging Technologies*, 14(3), 157-174.

Gendreau, M., Nossack, J., Pesch, E. (2015) Mathematical formulations for a 1-full-truckload pickup-and-delivery problem, *European Journal of Operational Research*, 242 (3), 1008-1016.

Ghilas, V., Demir, E., and Van Woensel, T. (2016) An adaptive large neighborhood search heuristic for the pickup and delivery problem with time windows and scheduled lines. *Computers & Operations Research*, 72, 12-30.

Goksal, F. P., Karaoglan, I., and Altiparmak, F. (2013) A hybrid discrete particle swarm optimization for vehicle routing problem with simultaneous pickup and delivery, *Computers & Industrial Engineering*, 65, 39-53.

Grangier, P., Gendreau, M., Lehuédé, F., and Rousseau, L. M. (2016) An adaptive large neighborhood search for the two-echelon multiple-trip vehicle routing problem with satellite synchronization. *European Journal of Operational Research*, 254, 80-91.

Greene, D. and Schafer, A. (2003) Reducing greenhouse gas emissions from U.S. transportation. center for climate and energy solutions: http://www.c2es.org/docUploads/ustransp.pdf.

Gromicho J., van Hoorn J. J., Kok A. L., and Schutten J. M. J. (2012) Restricted dynamic programming: a flexible framework for solving realistic VRPs. *Computers & Operations Research*, 39(5), 902-909.

Gschwind, T., Irnich, S., Rothenbächer, A.-K., and Tilk, C. (2017) Bidirectional labeling in column-generation algorithms for pickup-and-delivery problems, *European Journal of Operational Research*, In Press, https://doi.org/10.1016/j.ejor.2017.09.035.

Gutierrez-Jarpa, G., Desaulniers, G., Laporte, G., and Marianov, V. (2010) A branch-and-price algorithm for the vehicle routing problem with deliveries, selective pickups and time windows, *European Journal of Operational Research*, 206(2), 341-349.

Hall, R.W. (1991) Queueing Methods for Services and Manufacturing, Prentice Hall, Engelwood Cliffs, New Jersey.

Handy, S. L. (2005) Planning for accessibility: In theory and in practice. In D. M. Levinson & K. J. Krizek (Eds.), Access to Destinations (pp. 131–147). Amsterdam: Elsevier.

Hägerstrand, T. (1970) What about people in regional science? *Regional Science Association Papers*, 24, 7–21.

Hao, J., Hatzopoulou, M., and Miller, E. (2010) Integrating an Activity-Based Travel Demand Model with Dynamic Traffic Assignment and Emission Models: Implementation in the Greater Toronto, Canada, Area. *Transportation Research Record: Journal of the Transportation Research Board*, (2176), 1-13.

Hägerstrand, T. (1970) What about people in regional science? Regional Science Association Papers, 24, 7-21.

Häme, L. (2011) An adaptive insertion algorithm for the single-vehicle dial-a-ride problem with narrow time windows. *European Journal of Operational Research*, 209(1), 11-22.

Häme, L. and Hakula H. (2015) A maximum cluster algorithm for checking the feasibility of dial-d-ride instances. *Transportation Science*, 49(2), 295-310.

Held, M. and Karp, R. M. (1962) A dynamic-programming approach to sequencing problems. *Journal of the Society for Industrial and Applied Mathematics*, 10(1), 196-210.

Hernández-Pérez, H. and Salazar-González, J.J. (2004) A branch-and-cut algorithm for a traveling salesman problem with pickup and delivery, *Discrete Applied Mathematics*, 145, 126-139.

Hernández-Pérez H. and Salazar-González J-J (2009) The multi-commodity one-to-one pickup-and-delivery traveling salesman problem. *European Journal of Operational Research*, 196(3), 987-995.

Horner, M. W. and Wood, B. S. (2014) Capturing individuals' food environments using flexible space–time accessibility measures. *Applied Geography*, *51*, 99–107.

Hosni, H., Naoum-Sawaya, J., and Artail, H., (2014) The shared-taxi problem: formulation and solution methods. *Transportation Research Part B: Methodological*, 70, 303-318.

Hu, X., Chiu, Y. C., Delgado, S., Zhu, L., Luo, R., Hoffer, P., and Byeon, S. (2014) Behavior insights for an incentive-based active demand management platform. In 93rd Annual Meeting of the *Transportation Research Board*, Washington, DC.

Huang, Y. H. and Ting, C. K. (2010) Ant colony optimization for the single vehicle pickup and delivery problem with time windows, The 2010 International Conference on Technologies and Applications of Artificial Intelligence, 537-543.

Humphrey, N. (1996) Expanding metropolitan highways: implications for air quality and energy use. *Transportation Research Record*, 183 (No. HS-042 177).

Ioachim, I., Desrosiers, J., Dumas, Y., Solomon, M. M., and Villeneuve, D. (1995) A request clustering algorithm for door-to-door handicapped transportation. *Transportation Science*, 29, 63-78.

Irnich, S. (2000) A multi-depot pickup and delivery problem with a single hub and heterogeneous vehicles. *European Journal of Operational Research*, 122(2), 310-328.

Irnich, S., and Desaulniers, G. (2005) Shortest path problems with resource constraints. Column generation, 33–65.

Javid, R. J., Nejat, A., and Hayhoe, K. (2014) Selection of CO2 mitigation strategies for road transportation in the United States using a multi–criteria approach. *Renewable and Sustainable Energy Reviews, 38*, 960–972.

Jaw, J., Odoni, A., Psaraftis, H. N, and Wilson, N. (1986) A heuristic algorithm for the multi-vehicle advance-request dial-a-ride problem with time windows. *Transportation Research Part B: Methodological*, 20, 243-257.

Jorgensen, R. M., Larsen, J., and Bergvinsdottir, K.B. (2007) Solving the dial-a-ride problem using genetic algorithms. *Journal of the operational research society*, 58, 1321-1331.

Jung, S., Haghani, A. (2000) A genetic algorithm for pick-up and delivery problem with time windows. *Transportation Research Record* 1733, Transportation Research Board, 1-7.

Kikuchi, S. and Donnelly, R. A. (1992) Scheduling demand-responsive transportation vehicles using fuzzy-set theory. *Journal of Transportation Engineering*, 118, 391-409.

Kim, M., Schonfeld, P. (2014) Integration of conventional and flexible bus services with timed transfers. *Transportation Research Part B: Methodological*, 68, 76-97.

Kirchler, D. and Calvo, R. W. (2013) A granular tabu search algorithm for the dial-a-ride problem, *Transportation Research Part B: Methodological*, 56, 120-135.

Kitamura, R. (1988) An evaluation of activity-based travel analysis. *Transportation*, 15(1-2), 9-34.

Kok, A. L., Hans, E. W., and Schutten, J. M. J. (2012) Vehicle routing under time-dependent travel times: the impact of congestion avoidance. *Computers & Operations Research*, 39(5), 910-918.

Kuijpers, B., Miller, H. J., Neutens, T., and Othman, W. (2010) Anchor uncertainty and space–time prisms on road networks. *International Journal of Geographical Information Science*, *24*(8), 1223–1248.

Kuijpers, B. and Othman, W. (2009) Modeling uncertainty of moving objects on road networks via space-time prisms. *International Journal of Geographical Information Science*, 23:1095-1117.

Kwan, M.–P. (1998) Space–time and integral measures of individual accessibility: A comparative analysis using a point–based framework. *Geographical Analysis*, 30, 191–216.

Kwan, M.–P. (1999) Gender and individual access to urban opportunities: A study using space–time measures. *The Professional Geographer*, 51, 211–227.

Li, H. and Lim, A. (2001) A metaheuristic for the pickup and delivery problem with time windows. ICTAI-2001, 13th IEEE Conf. Tools Artificial Intelligence, Dallas, TX, 160-170.

Li, P., Mirchandani, P., Zhou, X. (2015) Solving simultaneous route guidance and traffic signal optimization problem using space-phase-time hyper network, *Transportation Research Part B: Methodological*, 81, 103-130.

Lin, S. (1965) Computer solutions of the traveling salesman problem. *Bell System Technical Journal*, 44: 2245-2269.

Lin, C., Choy, K. L., Ho, G. T. S., and Ng, T. W. (2014) A genetic algorithm-based optimization model for supporting green transportation operations, *Expert Systems with Applications*, 41(7), 3284-3296.

Lin, D.Y., Eluru, N., Waller, S., and Bhat, C. (2008) Integration of Activity-Based Modeling and Dynamic Traffic Assignment. *Transportation Research Record: Journal of the Transportation Research Board*, (2076), 52-61.

Lin, S. and Kernighan, B. W. (1973) An effective heuristic algorithm for the traveling salesman problem. *Operations Research*, 21: 498-516.

Liu, Y., Seah, H. S., and Shou, G. (2017) Constrained energy–efficient routing in time–aware road networks, *Geoinformatica*, 21, 89–117.

Lu, Q. and Dessouky, M. (2004) An exact algorithm for the multiple vehicle pickup and delivery problem. *Transportation Science*, 38(4) 503-514.

Lu, Q. and Dessouky, M. M. (2006) A new insertion-based construction heuristic for solving the pickup and delivery problem with hard time windows, *European Journal of Operational Research*, 175 (2), 672-687.

Lu, E. H.-Ch., Yang, Y.-W., and Su, Z. L.-T. (2016) Ant Colony Optimization Solutions for Logistic Route Planning with Pick-up and Delivery, *IEEE International Conference on Systems, Man, and Cybernetics*, Budapest, Hungary.

Luo, Y. and Schonfeld. P. (2007) A rejected-reinsertion heuristic for static Dial-A-Ride Problem. *Transportation Research Part B*, 41, 7, 736-755.

Luo, Y. and Schonfeld. P. (2011) Online rejected-reinsertion heuristic for the dynamic multi-vehicle dial-a-ride problem. *TRB 90th Annual Meeting at Washington D.C*.

Madsen, O. B. G., Ravn, H. F., and Rygaard, J. M. (1995) A heuristic algorithm for dial-a-ride problem with time windows, multiple capacities, and multiple objectives. *Annals of Operations Research*, 60, 193-208.

Mahmassani, H.S. Beyond the data: Extracting knowledge, deriving insight, delivering Intelligence, Available at: http://www.transportation.northwestern.edu/docs/2012/2012.10.30_HSM_Workshop.pdf, 2012, Retrieved 15 July, 2016.

Mahmoudi, M. and Zhou, X. (2016) Finding optimal solutions for vehicle routing problem with pickup and delivery services with time windows: a dynamic programming approach based on state-space-time network representations, *Transportation Research Part B, Methodological*, 89, 19-42.

Makigami, Y., Newell, G.F., and Rother, R. (1971) Three-dimensional representations of traffic flow. *Transportation Science*, 5, 302-313.

Masson, R., Lehuédé, F., and Péton, O. (2013) An adaptive large neighborhood search for the pickup and delivery problem with transfers. *Transportation Science*, 47(3): 344-355.

Masson, R., Lehuédé, F., and Péton, O. (2014) The dial-a-ride problem with transfers, *Computers & Operations Research*, 41(1), 12-23.

Männel, D. and Bortfeldt, A. (2016) A hybrid algorithm for the vehicle routing problem with pickup and delivery and three-dimensional loading constraints, *European Journal of Operational Research*, 254 (3), 840-858.

Melachrinoudis, E., Ilhan, A. B., and Min, H. (2007) A dial-a-ride problem for client transportation in a healthcare organization. *Computers & Operations Research*, 34, 742-759.

Meng, L. and Zhou, X. (2014) Simultaneous train rerouting and rescheduling on an N-track network: A model reformulation with network-based cumulative flow variables. *Transportation Research Part B: Methodological, 67*, 208-234.

Miller, H. J. (1991) Modelling accessibility using space-time prism concepts within geographical information systems. *International Journal of Geographical Systems*, 5 (3), 287-301.

Miller, H. J. (1999) Measuring space–time accessibility benefits within transportation networks: Basic theory and computational methods. *Geographical Analysis*, 31, 187–212.

Miller, H. J. (2005) Place–based versus people–based accessibility. In *Access to destinations* (pp. 63–89). Emerald Group Publishing Limited.

Miller-Hooks, E.D. and Mahmassani, H.S. (1998) Least possible time paths in stochastic, time-varying networks. *Computers & Operations Research*, 25, 1107-1125.

Miller-Hooks, E.D. and Mahmassani, H.S. (2000) Least expected time paths in stochastic, time-varying transportation networks. *Transportation Science*, 34 (2), 198-215.

Mitrovic-Minic′, S., Krishnamurti, R., and Laporte G. (2004) Double-horizon based heuristics for the dynamic pickup and delivery problem with time windows. *Transportation Research Part B: Methodological*, 38, 669-685.

Mitrovic´-Minic´, S., Laporte, G. (2006) The pickup and delivery problem with time windows and transshipment. *INFOR: Information Systems and Operational Research*. 44, 217-227.

Montane, F. A. T. and Galvao, R. D. (2006) A tabu search algorithm for the vehicle routing problem with simultaneous pickup and delivery service. *Computers & Operations Research*, 33, 595-619.

Morning Consult, 2015. National Tracking Poll #150505 - June 29-31, 2015. Morning Consult, Washington, D.C. [ONLINE] Available at: https://morningconsult.com/wp-content/uploads/2015/06/150505_crosstabs_mc_v2_AD.pdf, Retrieved 15 July, 2016.

Mues, C. and Pickl, S. (2005) Transshipment and time windows in vehicle routing. In *Proceedings of the 8th international symposium on parallel architectures. Algorithms and networks* (pp. 113-119). Piscataway, NJ: IEEE.

Murrel, D. (1980) Passenger Car Fuel Economy: EPA and Road: A Report to the Congress in Response to the National Energy Conservation Policy Act of 1978, Public Law 95-619, Title IV, Part 1, Section 404.

Nanry, W. P. and Barnes, J. W. (2000) Solving the pickup and delivery problem with time windows using reactive tabu search, *Transportation Research Part B: Methodological*, 34(2), 107-121.

Neutens, T., Schwanen, T., Witlox, F., and De Maeyer, P. (2010) Equity of urban service delivery: A comparison of different accessibility measures. *Environment and Planning A*, 42, 1613–1635.

Newell, G.F. (1982) *Applications of Queueing Theory,* Chapman and Hall, London.

Newell, G.F. (1993) A simplified theory of kinematic waves in highway traffic, part I: General theory. *Transportation Research Part B*, 27(4), 281-287.

Nie, Y. M. and Ghamami, M. (2013) A corridor-centric approach to planning electric vehicle charging infrastructure. *Transportation Research Part B*, 57, 172–190.

Osenga, K. (2005) Entrance Ramps, Tolls, and Express Lanes–Proposals for Decreasing Traffic Congestion in the Patent Office. Fla. St. UL Rev., 33, 119, 130.

Pankratz, G. (2005) A grouping genetic algorithm for the pickup and delivery problem with time windows, *OR Spectrum*, 27, 21-41.

Paquette, J., Cordeau, J.-F., Laporte, G., and Pascoal, M. M. B. (2013) Combining multicriteria analysis and tabu search for dial-a-ride problems. *Transportation Research Part B: Methodological*, 52, 1-16.

Parragh, S.N. (2009) Ambulance Routing Problems with Rich Constraints and Multiple Objective, Ph. D. Thesis, University of Vienna, Faculty of Business, Economics and Statistics.

Parragh, S., and Schmid, V. (2013) Hybrid column generation and large neighborhood search for the Dial-A-Ride Problem, *Computers & Operations Research*, 40(1), 490-497.

Peeta, S. and Ziliaskopoulos, A. K. (2001) Foundations of dynamic traffic assignment: The past, the present and the future. *Networks and Spatial Economics*, 2001, *1*(3-4), pp. 233-265.

Pendyala, R., Konduri, K., Chiu, Y.C., Hickman, M., Noh, H., Waddell, P., Wang, L., You, D., and Gardner, B. (2012) Integrated Land Use-Transport Model System with Dynamic Time-Dependent Activity-Travel Microsimulation. *Transportation Research Record: Journal of the Transportation Research Board*, (2303), 19-27.

Psaraftis, H. N. (1980) A dynamic programming approach to the single-vehicle, many-to-many immediate request dial-a-ride problem. *Transportation Science*, 14(2) 130-154.

Psaraftis, H. N. (1983) An exact algorithm for the single-vehicle many-to-many dial-a-ride problem with time windows. *Transportation Science*, 17(3) 351-357.

Psaraftis, H. N., Orlin, J. B., Bienstock, D., and Thompson, P. M. (1985) Analysis and solution algorithms of sealift routing and scheduling problems: Final report. Technical Report 1700-85, MIT, Sloan School of Management, Cambridge, MA.

Qu, Y. and Bard, J. F. (2012) A GRASP with adaptive large neighborhood search for pickup and delivery problems with transshipment, *Computers & Operations Research*, 39(10), 2439-2456.

Quadrifoglio, L., Dessouky, M. M., and Palmer, K. (2007) An insertion heuristic for scheduling Mobility Allowance Shuttle Transit (MAST) services, *Journal of Scheduling*, 10, 25-40.

Rais, A., Alvelos, F., and Carvalho, M.S. (2014) New mixed integer programming model for the pickup-and-delivery problem with transshipment. *European Journal of Operational Research*, 235(3):530-539.

Rappoport, H. K., Levy, L. S., Golden, B. L., and Toussaint, K. (1992) A planning heuristic for military airlift. *Interfaces*, 22:73-87, 1992.

Rappoport, H. K., Levy, L. S., Toussaint, K., and Golden, B. L. (1994) A transportation problem formulation for the MAC airlift planning problem. *Annals of Operations Research*, 50, 505-523.

Rasouli, S., and H. Timmermans, (2014) Activity-based models of travel demand: promises, progress and prospects. *International Journal of Urban Sciences*, 18(1), pp. 31-60.

Raubal, M., Miller, H. J., and Bridwell, S. (2004) User-centred time geography for location-based services. *Geografiska Annaler: Series B, Human Geography*, *86*(4), 245–265.

Raviv, T., Tzur, M., and Forma, I.A. (2013) Static repositioning in a bike-sharing system: models and solution approaches. *EURO Journal on Transportation and Logistics*, 2: 187-229.

Rekiek, B., Delchambre, A., and Saleh, H. A. (2006) Handicapped person transportation: an application of the grouping genetic algorithm. *Engineering Application of Artificial Intelligence*, 19, 511-520.

Ribeiro, G. M. and Laporte, G. (2012) An adaptive large neighborhood search heuristic for the cumulative capacitated vehicle routing problem, *Computers & Operations Research*, 39, 728-735.

Ritzinger, U., Puchinger, J., and Hartl, R.F. (2016) Dynamic programming based metaheuristics for the dial-a-ride problem. *Annals of Operations Research*, 236(2), 341-358.

Rodrigues, V.P., Morabito, R., Yamashita, D., Silva, B.J., and Ribas, P. (2016) Ship routing with pickup and delivery for maritime oil transportation system: MIP model and heuristics. *Systems*, 4(3), 31.

Ropke, S. and Cordeau, J.-F. (2009) Branch and cut and price for the pickup and delivery problem with time windows. *Transportation Science*, 43(3) 267-286.

Ropke, S., Cordeau, J.-F., and Laporte, G. (2007) Models and branch-and-cut algorithms for pickup and delivery problems with time windows. *Networks*, 49(4) 258-272.

Ropke, S. and Pisinger, D. (2006) An adaptive large neighborhood search heuristic for the pickup and delivery problem with time windows. *Transportation Science*, 40(4) 455-472.

Roy, S., Rousseau, J. M., Lapalme, G., and Ferland, J. A. (1984a) Routing and scheduling for the transportation of disabled persons—the algorithm. Technical Report TP 5596E, Centre de Recherche sur les Transports, Montréal, Canada.

Roy, S., Rousseau, J. M., Lapalme, G., and Ferland, J. A. (1984b) Routing and scheduling for the transportation of disabled persons—the tests. Technical Report TP 5598E, Centre de Recherche sur les Transports, Montréal, Canada.

Ruland, K. S. (1995) Polyhedral solution to the pickup and delivery problem. PhD thesis, Sever Institute of Technology, Washington University, St. Louis, MO.

Ruland, K.S. and Rodin, E.Y. (1997) The pickup and delivery problem: Faces and branch-and-cut algorithm, *Computers & Mathematics with Applications*, 33 (12), 1-13.

Salazar-González, J.J., Santos-Hernández, B. (2015) The split-demand one-commodity pickup-and-delivery travelling salesman problem, *Transportation Research Part B: Methodological*, 75, 58-73.

Savelsbergh, M. and Sol, M. (1998) Drive: Dynamic routing of independent vehicles. *Operations Research*, 46(4) 474-490.

Schneider, M., Stenger, A., and Goeke, D. (2014) The electric vehicle–routing problem with time windows and recharging stations. *Transportation Science*, 48(4), 500–520.

Schwanen, T., Banister, D., and Anable, J. (2012) Rethinking habits and their role in behaviour change: the case of low–carbon mobility. *Journal of Transport Geography*, 24, 522–532.

Sexton, T. R. and Bodin, L. D. (1985a) Optimizing single vehicle many-to-many operation with desired delivery times: I. Scheduling. *Transportation Science*, 19(4) 378-410.

Sexton, T. R. and Bodin, L. D. (1985b) Optimizing single vehicle many-to-many operation with desired delivery times: II. Routing. *Transportation Science*, 19(4) 411-435.

Sexton T. and Choi, Y. (1986) Pickup and delivery of partial loads with time windows*, American Journal of Mathematical and Management Sciences,* 6, 369-398.

Shen, Y., Potvin, J.-Y., Rousseau, J.-M., and Roy, S. (1995) A computer assistant for vehicle dispatching with learning capabilities. *Annals of Operations Research*, 61, 189-211.

Sherali, H. D. and Smith, J. C. (2001) Improving Discrete Model Representations via Symmetry Considerations. *Management Science*, 47, 1396-1407.

Sherman, D. (2016) *Semi-Autonomous Cars Compared! Tesla Model S vs. BMW 750i, Infiniti Q50S, and Mercedes-Benz S65 AMG.* Available at: http://www.caranddriver.com/features/semi-autonomous-cars-compared-tesla-vs-bmw-mercedes-and-infiniti-feature, Retrieved 15 July, 2016.

Shi, T. and Zhou, X. (2015) A mixed integer programming model for optimizing multi-level operations process in railroad yards. *Transportation Research Part B: Methodological*, *80*, 19-39.

Shiri, S. and Huynh, N. (2016) Optimization of drayage operations with time-window constraints*, International Journal of Production Economics*, 176, 7-20.

Shiri, S. and Huynh, N. (2017) Assessment of U.S. chassis supply models on drayage productivity and air emissions. *Transportation Research Part D: Transport and Environment*, http://dx.doi.org/10.1016/j.trd.2017.04.024.

Sol, M. (1994) Column generation techniques for pickup and delivery problems. Ph.D. thesis, Eindhoven University of Technology, Eindhoven, The Netherlands.

Solanki, R. S., and Southworth, F. (1991) An execution planning algorithm for military airlift. *Interfaces*, 21, 121-131.

Solomon, M. M., Chalifour, A., Desrosiers, J., and Boisvert, J. (1992) An application of vehicle routing methodology to large-scale larvicide control programs. *Interfaces*, 22, 88-99.

Sombuntham, P. and Kachitvichayanukul, V. (2010) A Particle Swarm Optimization Algorithm for Multi-depot Vehicle Routing problem with Pickup and Delivery Requests, *Proceeding of International Conference of Engineers and Computer Scientists*, Hong Kong, 2010, Vol III, IMECS.

Spieser, K., Treleaven, K., Zhang, R., Frazzoli, E., Morton, D., and Pavone, M. (2014) Toward a systematic approach to the design and evaluation of automated mobility–on–demand systems: A case study in Singapore. In *Road Vehicle Automation* (pp. 229–245). Springer International Publishing.

Stein, W. R. and Walker, D. (2003) Link-based calculation of motor vehicle air toxin emissions using MOBILE 6.2, Proceedings of the Ninth TRB Conference on the Application of Transportation Planning Methods, April 6-10, 2003, Baton Rouge, Louisiana.

Subramanian, A., Drummond, L., Bentes, C., Ochi, L., and Farias, R. (2010) A parallel heuristic for the vehicle routing problem with simultaneous pickup and delivery, *Computers & Operations Research*, 37(11), 1899-1911.

Subramanian, A., Uchoa, E., Pessoa, A. A., and Ochi, L. S. (2013) Branch-cut-and-price for the vehicle routing problem with simultaneous pickup and delivery, *Optimization Letters*, 7, 1569-1581.

Sullivan, J. L., Novak, D. C., Aultman–Hall, L., and Scott, D. M. (2010) Identifying critical road segments and measuring system–wide robustness in transportation networks with isolating links: A link-based capacity-reduction approach. *Transportation Research Part A: Policy and Practice*, *44*(5), 323–336.

Sun, Y. and Schonfeld, P. (2016) Holding decisions for correlated vehicle arrivals at intermodal freight transfer terminals. *Transportation Research Part B: Methodological*, 90, 218-240.

https://tianchi.shuju.aliyun.com/competition/information.htm?spm=5176.100067.5678.2.LRPZpR&raceId=231581

Sungur, I., Ordóñez, F., and Dessouky, M. M. (2008) A robust optimization approach for the capacitated vehicle routing problem with demand uncertainty, *IIE Transactions*, 40, 509-523.

Swersey, A. and Ballard, W. (1983) Scheduling school buses. *Management Science*, 30, 844-853.

Sweda, T. and Klabjan, D. (2011) An agent-based decision support system for electric vehicle charging infrastructure deployment. In: 7th IEEE Vehicle Power and Propulsion Conference, Chicago, Illinois.

Tasan A. S. and Gen, M. (2012) A genetic algorithm based approach to vehicle routing problem with simultaneous pick-up and deliveries, *Computers & Industrial Engineering*, 62, 755-761.

Teodorovic, D. and Radivojevic, G. (2000) A fuzzy logic approach to dynamic dial-a-ride problem. *Fuzzy Sets and Systems*, 116, 23-33.

Tesla Motors Team. *Your Autopilot Has Arrived.* Available at: https://www.tesla.com/blog/your-autopilot-has-arrived, 2015, Retrieved 15 July, 2016.

Thomas, C. S. (2009) Transportation options in a carbon–constrained world: Hybrids, plug–in hybrids, biofuels, fuel cell electric vehicles, and battery electric vehicles. *International Journal of Hydrogen Energy*, *34*(23), 9279–9296.

Tong, L., Zhou, X., and Miller, H. J. (2015) Transportation network design for maximizing space–time accessibility. *Transportation Research Part B: Methodological*, *81*, 555–576.

Toth, P. and Vigo, D. (1997) Heuristic algorithms for the handicapped persons transportation problem. *Transportation Science*, 31, 60-71.

Toth, P. and Vigo, D. (2014) Vehicle routing: problems, methods, and applications (Vol. 18), *SIAM*.

USEPA, 2012. Motor Vehicle Emission Simulator (MOVES) 2010 User Guide. EPA–420–B–12–001b

Van der Bruggen, L. J. J., Lenstra, J. K., and Schuur P. C. (1993) Variable-depth search for the single-vehicle pickup and delivery problem with time windows. *Transportation Science*, 27, 298-311.

Veenstra, M., Cherkesly, M., Desaulniers, G., and Laporte, G. (2017) The pickup and delivery problem with time windows and handling operations. *Computers & Operations Research*, 77, 127-140.

Venkateshan, P. and Mathur, K. (2011) An efficient column-generation-based algorithm for solving a pickup-and-delivery problem, *Computers & Operations Research*, 38(12), 1647-1655.

Walsh, T. (2012) Symmetry breaking constraints: Recent results. In *Proceedings of the Twenty-Sixth AAAI Conference on Artificial Intelligence*, 2192-2198.

Wang, H.-F. and Chen, Y.-Y. (2012) A genetic algorithm for the simultaneous delivery and pickup problems with time window, *Computers & Industrial Engineering*, 62, 84-95.

Wang, X., Dessouky, M., and Ordonez, F. (2015) A Pickup and delivery problem for ridesharing considering congestion. *Transportation Letters The International Journal of Transportation Research*, doi: 10.1179/1942787515Y.0000000023.

Wang, X., Dessouky, M., and Ordonez, F. (2016) A Pickup and delivery problem for ridesharing considering congestion. *Transportation Letters,* 8:5, 259-269.

Wang, C., Mu, D., Zhao, F., Sutherland, J.W. (2015) A parallel simulated annealing method for the vehicle routing problem with simultaneous pickup-delivery and time windows, *Computers & Industrial Engineering*, 83, 111-122.

Wang, X. and Regan, A. C. (2002) Local truckload pickup and delivery with hard time window constraints. *Transportation Research Part B: Methodological*, 36, 97-112.

Wang, X. and Regan, A.C. (2009) On the convergence of a new time window discretization method for the traveling salesman problem with time window constraints. *Computers & Industrial Engineering*, 56(1): 161-164.

Widener, M. J., Farber, S., Neutens, T., and Horner, M. (2015) Spatiotemporal accessibility to supermarkets using public transit: an interaction potential approach in Cincinnati, Ohio. *Journal of Transport Geography*, *42*, 72–83.

Wilson H. and Colvin, N. (1977) Computer control of the Rochester dial-a-ride system. Technical Report R-77-31, Department of Civil Engineering, MIT, Cambridge, MA.

Wilson, H., Sussman, J. Wang, H., and Higonnet, B. (1971) Scheduling algorithms for dial-a-ride systems. Technical Report USL TR-70-13, Urban Systems Laboratory, MIT, Cambridge, MA.

Wilson H. and Weissberg, H. (1976) Advanced dial-a-ride algorithms research project: Final report. Technical Report R76-20, Department of Civil Engineering, MIT, Cambridge, MA.

Xu, H., Chen, Z.L., Rajagopal, S., and Arunapuram, S. (2003) Solving a practical pickup and delivery problem, *Transportation Science*, 37(3), 347-364.

Yang, L. and Zhou X. (2014) Constraint reformulation and a Lagrangian relaxation-based solution algorithm for a least expected time path problem. *Transportation Research Part B: Methodological*. 59, 22-44.

Yu, V. F. and Lin, S.-W. (2014) Multi-start simulated annealing heuristic for the location routing problem with simultaneous pickup and delivery. *Applied Soft Computing*, 24, 284-290.

Zachariadis, E., Tarantilis, C., and Kiranoudis, C. (2015) The load-dependent vehicle routing problem and its pick-up and delivery extension. *Transportation Research Part B: Methodological*, 71, 158-181.

Zhou, X. and Taylor, J. (2014) DTALite: a queue-based mesoscopic traffic simulator for fast model evaluation and calibration. *Cogent Engineering*, 1 (1), 961345.

Zhou, X., Tanvir, S., Lei, H., Taylor, J., Liu, B., Rouphail, N. M., and Frey, H. C. (2015) Integrating a simplified emission estimation model and mesoscopic dynamic traffic simulator to efficiently evaluate emission impacts of traffic management strategies. *Transportation Research Part D*, 37, 123–136.

Zhao, F., Li, S. Sun, J., and Mei, D. (2009) Genetic algorithm for the one-commodity pickup-and-delivery travelling salesman problem, *Computers & Industrial Engineering*, 56, 1642-1648.

Ziliaskopoulos, A.K. and Mahmassani, H. S. (1993) Time-dependent, shortest-path algorithm for real-time intelligent vehicle highway system application. *Transportation research record*, 1408, 94-100.

Zockaie, A., Saberi, M., Mahmassani, H.S., Jiang, L., Frei, A., and Hou, T. (2015) Towards Integrating an Activity-Based Model with Dynamic Traffic Assignment Considering Heterogeneous User Preferences and Reliability Valuation: Application to Toll Revenue Forecasting in Chicago. In *Transportation Research Board 94th Annual Meeting, Washington, D.C.*

APPENDIX A

DESCRIPTION OF THE PDPTW IN THE OD NETWORK

Cordeau (2006) formulated the PDPTW on a network that is built based on demand request nodes and the links are defined as direct connections between pickup and delivery nodes (without explicitly considering transportation links or paths). For a systematic comparison, the following notation is adapted from Cordeau (2006).

Table A.1
Sets, indices and parameters used in Cordeau (2006) for the PDPTW.

Symbol	Definition
$\boldsymbol{n}$	Number of passengers
$\boldsymbol{P}$	Set of passengers' pickup nodes. $P = \{1, \dots, n\}$
$\boldsymbol{D}$	Set of passengers' delivery nodes. $D = \{n+1, \dots, 2n\}$
0	Node representative of origin depot
$\boldsymbol{2n+1}$	Node representative of destination depot
$\boldsymbol{N}$	Set of passengers' pickup and drop-off nodes and vehicles' depots. $N = \{P, D, \{0, 2n+1\}\}$
$\boldsymbol{A}$	Set of arcs
$\boldsymbol{G}$	Directed graph $G = (N, A)$
$\boldsymbol{i}$	Passenger i's pickup node
$\boldsymbol{n+i}$	Passenger i's delivery node
$\boldsymbol{q_i}$	Load at node i, $(i \in N)$
$\boldsymbol{d_i}$	Service duration at node i, $(i \in N)$
$\boldsymbol{e_i}$	Earliest time at which service is allowed to start at node i, $(i \in N)$
$\boldsymbol{l_i}$	Latest time at which service is allowed to start at node i, $(i \in N)$
$(\boldsymbol{i,j})$	Index of arc between adjacent nodes i and j
$\boldsymbol{c_{ij}}$	Routing cost of arc (i,j)
$\boldsymbol{t_{ij}}$	Travel time of arc (i,j)
$\boldsymbol{V}$	Set of vehicles
$\boldsymbol{v}$	Vehicle index
$\boldsymbol{Q_v}$	Capacity of vehicle v
$\boldsymbol{T_v}$	Maximal duration of vehicle v's route
$\boldsymbol{L}$	Maximum ride time of a passenger

Note that $q_0 = q_{2n+1} = 0$, $q_i \geq 0$ for $(i = 1, \dots, n)$, and $q_i = -q_{i-n}$ $(i = n + 1, \dots, 2n)$, and service duration $d_i \geq 0$ and $d_0 = d_{2n+1} = 0$. Time window $[e_i, l_i]$ is also specified either for the pickup node or for the drop-off node of a request, but not for both. The arc set is also defined as $A = \{(i,j): (i = 0, j \in P)\ or\ (i \in P \cup D, j \in P \cup D, i \neq j, i \neq n + j)\ or\ (i \in D, j = 2n + 1)\}$. The model uses three-index variables x_{ij}^{v} being equal to 1 if and only if vehicle v travels from node i to node j. Let B_i^v be the time at which vehicle v begins servicing node i and Q_i^v be the load of vehicle v upon departing from node i. Finally, for each passenger i, let L_i^v be the ride time of passenger i on vehicle v. The PDPTW can be formulated as follows:

$$Min \sum_{v \in V} \sum_{i \in N} \sum_{j \in N} c_{ij}^v x_{ij}^v \qquad (A.1)$$

s.t.

$$\sum_{v \in V} \sum_{j \in N} x_{ij}^v = 1 \qquad \forall i \in P \qquad (A.2)$$

$$\sum_{j \in N} x_{ij}^v - \sum_{j \in N} x_{n+i,j}^v = 0 \qquad \forall i \in P, v \in V \qquad (A.3)$$

$$\sum_{j \in N} x_{0j}^v = 1 \qquad \forall v \in V \qquad (A.4)$$

$$\sum_{j \in N} x_{ji}^v - \sum_{j \in N} x_{ij}^v = 0 \qquad \forall i \in P \cup D, v \in V \qquad (A.5)$$

$$\sum_{i \in N} x_{i,2n+1}^v = 1 \qquad \forall v \in V \qquad (A.6)$$

$$x_{ij}^v\left(B_i^v + d_i + t_{ij}\right) \leq B_j^v \qquad \forall i \in N, j \in N, v \in V \qquad (A.7)$$

$$x_{ij}^v\left(Q_i^v + q_j\right) \leq Q_j^v \qquad \forall i \in N, j \in N, v \in V \qquad (A.8)$$

$$L_i^v = B_{n+i}^v - (B_i^v + d_i) \qquad \forall i \in P, v \in V \qquad (A.9)$$

$$B_{2n+1}^v - B_0^v \leq T_v \qquad \forall v \in V \qquad (A.10)$$

$$e_i \leq B_i^v \leq l_i \qquad \forall i \in N, v \in V \qquad (A.11)$$

$$t_{i,n+i} \leq L_i^v \leq L \qquad \forall i \in P, v \in V \qquad (A.12)$$

$$max\{0, q_i\} \leq Q_i^v \leq min\{Q_v, Q_v + q_i\} \qquad \forall i \in N, v \in V \qquad (A.13)$$

$$x_{ij}^v \in \{0,1\} \qquad \forall i \in N, j \in N, v \in V \qquad (A.14)$$

The objective function (A.1) minimizes the total routing cost. (A.2) guarantees that each passenger is definitely picked up. (A.2) and (A.3) ensure that each passenger's origin and destination are visited exactly once by the same vehicle. (A.4) expresses that each vehicle v starts its route from the origin depot. (A.5) ensures the flow balance on each node. (A.6) expresses that each vehicle v ends its route at the destination depot. (A.7) and (A.8) ensure the validity of the time and load variables. (A.9) defines each passenger's ride time. (A.10) to (A.13) impose maximal duration of each route, time windows, the ride time of each passenger, and capacity constraints, respectively. Since the non-negativity of the ride time of each passenger guarantees that node i is visited before node $n + i$, (A.12) also functions as precedence constraints.

APPENDIX B

MIXED INTEGER-PROGRAMMING MODEL FOR THE PDPT PROPOSED BY RAIS ET AL. (2014)

Rais et al. (2014) formulated the PDPT on a network that is built based on demand request nodes. The links are defined as direct connections between pickup and delivery nodes (without explicitly considering transportation links or paths). For a systematic comparison, the following notation is adapted from Rais et al. (2014). Moreover, this is the modified version of Rais et al. (2014) in which constraints (B.16) and (B.17) have been modified.

Table B.1
Sets, indices, parameters, and variables used by Rais et al. (2014) for the PDPT.

Symbol	Definition
G	Directed graph $G = (N, A)$ having node-set N and arc-set A
N	Set of nodes
A	Set of arcs
ij	Arc from node i to node j; $i, j \in N$ and $ij \in A$
K	Set of vehicles
k	Vehicle index; $k = 1, \dots, \lvert K \rvert$
u_k	Vehicle k's load-carrying capacity
$o(k), o'(k)$	Vehicle k's initial depot and final depot, respectively; $o(k), o'(k) \in N$
R	Set of passenger pickup-and-delivery requests
r	Request index; $r = 1, \dots, \lvert R \rvert$
q_r	Quantity of request r
$p(r)$	Pickup node associated with passenger request r; $p(r) \in N$
$d(r)$	Delivery node associated with passenger request r; $d(r) \in N$
T	Set of transshipment nodes; $T \subseteq N$
c_{ij}^{k}	Unit cost of transportation from node i to j using vehicle k
τ_{ij}^{k}	The time required by vehicle k to go from node i to node j
$[a_{p(r)}, b_{p(r)}]$	Request r's pickup time window
$[a_{d(r)}, b_{d(r)}]$	Request r's delivery time window
x_{ij}^{k}	$= 1$ if vehicle k uses arc ij, and $= 0$ otherwise
y_{ij}^{kr}	$= 1$ if vehicle k carries request r on arc ij, and $= 0$ otherwise
z_{ij}^{k}	$= 1$ if node i precedes node j (not necessarily immediately) in the route of vehicle k, and $= 0$ otherwise
s_{jr}^{kl}	$= 1$ if request r is transferred from vehicle k to vehicle l ($l \neq k$) at node j, and $= 0$ otherwise
t_j^k, $\bar{t}_j^k$	The arrival and departure times of vehicle k at node j, respectively. For each arc $ij \in A$ such that $x_{ij}^{k} = 1$, we must have $t_j^k \geq \bar{t}_i^k + \tau_{ij}^{k}$, as well as $\bar{t}_j^k \geq t_j^k$

Then, the PDPT with time windows is formulated as follows:

$$Min \ \sum_{k \in K} \sum_{ij \in A} c_{ij}^{k} x_{ij}^{k} \qquad \text{(B.1)}$$

subject to:

$\sum_{j:o(k)j\in A} x^k_{o(k)j} \leq 1 \qquad \forall k \in K \qquad$ (B.2)

$\sum_{j:o(k)j\in A} x^k_{o(k)j} = \sum_{j:jo'(k)\in A} x^k_{jo'(k)} \qquad \forall k \in K \qquad$ (B.3)

$\sum_{j:ij\in A} x^k_{ij} - \sum_{j:ji\in A} x^k_{ji} = 0 \qquad \forall k \in K, \forall i \in N\backslash\{o(k), o'(k)\} \qquad$ (B.4)

$\sum_{k\in K} \sum_{j:p(r)j\in A} y^{kr}_{p(r)j} = 1 \qquad \forall r \in R \qquad$ (B.5)

$\sum_{k\in K} \sum_{j:jd(r)\in A} y^{kr}_{jd(r)} = 1 \qquad \forall r \in R \qquad$ (B.6)

$\sum_{k\in K} \sum_{j:ij\in A} y^{kr}_{ij} - \sum_{k\in K} \sum_{j:ji\in A} y^{kr}_{ji} = 0 \qquad \forall r \in R, \forall i \in T \qquad$ (B.7)

$\sum_{j:ij\in A} y^{kr}_{ij} - \sum_{j:ji\in A} y^{kr}_{ji} = 0 \qquad \forall k \in K, \forall r \in R, \forall i \in N\backslash T \qquad$ (B.8)

$y^{kr}_{ij} \leq x^k_{ij} \qquad \forall ij \in A, \forall k \in K, \forall r \in R \qquad$ (B.9)

$\sum_{r\in R} q_r y^{kr}_{ij} \leq u_k x^k_{ij} \qquad \forall ij \in A, \forall k \in K \qquad$ (B.10)

$x^k_{ij} \leq z^k_{ij} \qquad \forall i,j \in N, \forall k \in K, i \neq o(k), j \neq o'(k) \qquad$ (B.11)

$z^k_{ij} + z^k_{ji} = 1 \qquad \forall i,j \in N, \forall k \in K, i \neq o(k), j \neq o'(k) \qquad$ (B.12)

$z^k_{ij} + z^k_{jl} + z^k_{li} \leq 2 \quad \forall i,j,l \in N, \forall k \in K, i,j \neq o(k), l \neq o'(k) \qquad$ (B.13)

$\bar{t}^k_i + \tau^k_{ij} - t^k_j \leq M(1 - x^k_{ij}) \qquad \forall ij \in A, \forall k \in K \qquad$ (B.14)

$t^k_j \leq \bar{t}^k_j \qquad \forall j \in N, \forall k \in K \qquad$ (B.15)

$\sum_{i:ij\in A} a_{p(r)} y^{kr}_{ip(r)} \leq t^k_{p(r)}, \bar{t}^k_{p(r)} \leq \sum_{j:ij\in A} b_{p(r)} y^{kr}_{p(r)j} \qquad \forall k \in K, \forall r \in R \qquad$ (B.16)

$\sum_{i:ij\in A} a_{a_{d(r)}} y^{kr}_{ia_{d(r)}} \leq t^k_{d(r)}, \bar{t}^k_{d(r)} \leq \sum_{j:ij\in A} b_{d(r)} y^{kr}_{d(r)j} \qquad \forall k \in K, \forall r \in R \qquad$ (B.17)

$\sum_{j:ji\in A} y^{kr}_{ji} + \sum_{j:ij\in A} y^{lr}_{ij} \leq s^{kl}_{jr} + 1 \qquad \forall r \in R, \forall i \in T, \forall k,l \in K, k \neq l \qquad$ (B.18)

$t^k_j - \bar{t}^l_j \leq M(1 - s^{kl}_{jr}) \qquad \forall r \in R, \forall j \in T, \forall k,l \in K, k \neq l \qquad$ (B.19)

$x^k_{ij} \in \{0,1\} \qquad \forall ij \in A, \forall k \in K \qquad$ (B.20)

$y_{ij}^{kr} \in \{0,1\} \qquad \forall ij \in A, \forall k \in K, \forall r \in R \qquad$ (B.21)

The objective of the problem is to find a set of minimum-cost vehicle routes for meeting all passenger requests. Constraint (B.2) ensures that each vehicle starts at most one route from its origin depot. Not all of the available vehicles may have to be used for meeting the passenger requests. Constraint (B.3) guarantees the same vehicle ends the route at its final depot. Flow conservation of the vehicles through the nodes in the network must be maintained (Constraint B.4). All pickups and deliveries must be enforced (constraints (B.5) and (B.6)). Constraint (B.7) states that the request flow conservation must be maintained at the transshipment nodes where the requests are allowed to switch from one vehicle to another. The request flow conservation is maintained at the non-transshipment nodes by constraint (B.8). Therefore, any vehicle that picks up a request must also drop off the same request. Constraint (B.9) highlights that a vehicle flow on an arc must exist if there is some request flow in the same vehicle on the same arc. The capacity of each vehicle on each arc of the network is assured by constraint (B.10). The sub-tours in the solution are eliminated by constraints (B.11), (B.12), and (B.13). Constraint (B.14) states that if vehicle flow k on arc ij exists, we must have $t_j^k \geq \bar{t}_i^k + \tau_{ij}^k$, as well as $\bar{t}_i^k \geq t_j^k$. Constraints (B.16) and (B.17) ensure that all pickup and delivery time windows are imposed, respectively. The precedence and synchronization of the transport load transfers between vehicles at the transshipment nodes are managed by constraints (B.18) and (B.19). Constraint (B.20) states that variables x_{ij}^k and y_{ij}^{kr} are binary. We provide the model proposed by Rais et al. (2014) to show our motives for defining our model on hyper-networks.

(1) The Rais et al. (2014) mixed integer programming model solves the problem on passengers’ origin/destination-based network. Their model does not work with transportation networks directly in which travel time may vary over the time of the day or with the load of the vehicle (e.g. HOV or HOT lanes).

(2) By the serial structure of our proposed hyper-network, we can tackle the symmetry issue, which has been comprehensively explained at the end of Section 4.6.

(3) We do not have non-linear constraints related to the validity of the time and load variables such as the ones presented in the Rais et al. (2014) mixed integer programming model.

(4) We convert our problem to a time-dependent state-dependent least cost path problem which can be solved by computationally efficient algorithms.

(5) Apart from the first phase where the algorithm clusters OD pairs and uses a heuristic for that, the other phases provide a near- optimal solution: the second phase of our algorithm provides a pseudo-optimal solution (pseudo-optimal due to the time discretization) and the last phase provides an optimal chain of work pieces that can be performed by each real vehicle.

(6) Our model is relatively flexible to add any further constraints related to the passengers’ preferences (e.g. vehicle type, particular departure/arrival time windows, ride time, etc.) and also vehicles’ restrictions (e.g. drivers’ work shift, vehicles capacity, etc.).

APPENDIX C

ARAMETERS TUNNG FOR CAINIAO NETWORK

We have taken our group 31 as an example of a group of tasks to explain how we have adjusted these parameters for a group of tasks by the aid of our evaluation function. In these experiments, we have defined 5 different scenarios as follows:

I. Set $\beta_A^{P\&D} = 2$, $\beta_B^{P\&D} = 1$ and define some sub-scenarios to adjust the values of β_A^P and β_B^P;

II. Set $\beta_A^{P\&D} = 2$, $\beta_B^{P\&D} = 0.5$ and define some sub-scenarios to adjust the values of β_A^P and β_B^P;

III. Set $\beta_A^{P\&D} = 2$, $\beta_B^{P\&D} = 1.5$ and define some sub-scenarios to adjust the values of β_A^P and β_B^P;

IV. Set $\beta_A^{P\&D} = 2$, $\beta_B^{P\&D} = 2$ and define some sub-scenarios to adjust the values of β_A^P and β_B^P;

V. Set $\beta_A^{P\&D} = 2$, $\beta_B^{P\&D} = 2.5$ and define some sub-scenarios to adjust the values of β_A^P and β_B^P;

Note that in Table C1, the demand completion time is the time that the last vehicle finishes its task. Fig. C1 illustrates the values of the objective function for different scenario-sub scenarios to help us choose the best parameter settings. Scenario I. sub-scenario III has the smallest objective function among other scenarios. Therefore, we set the parameters of group 31 based on the parameters used in scenario I. sub-scenario III.

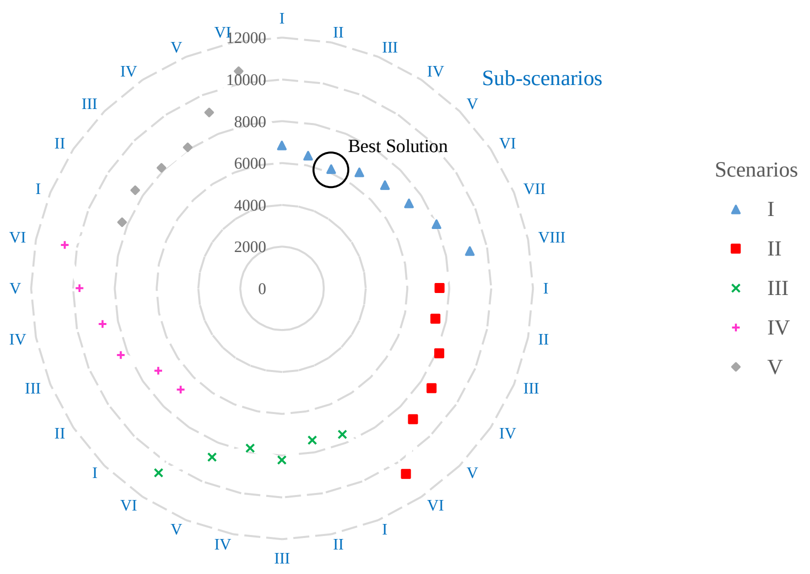


Fig. C.1. A comparison between the values of the objective function for different scenarios-sub scenarios for group 31.

In Fig. C.2, we intend to show the cumulative arrival/departure and on board diagrams for tasks in case scenario I, sub-scenarios I to VIII. In this figure, the graphs indicated by black color are the cumulative arrival tasks to the system, the graphs indicated by red color are the cumulative on board tasks (pickup only), and the graphs indicated by blue color are the cumulative departure tasks from the system. Fig. C.2(a) to C.2(d) are the cumulative flow graphs for scenario I, sub-scenario I to IV, while C.2(e) to C.2(h) are the cumulative flow graphs for scenario I, sub-scenario V to VIII. There is no incomplete task in sub-scenarios I to IV, while a number of tasks have remained incomplete in sub-scenarios V to VIII.

Table C.1
Parameter setting for group 31.

Scenario No.	Sub-scenario No.					Performed tasks				Total travel time (min)	Demand completion time (min)	Total # of vehicles needed	objective function
						Type “A”		Type “B”					
		$\beta_A^{P\&}$	$\beta_B^{P\&D}$	β_A^{P}	β_B^{P}	Picked up only	Picked up & delivered	Picked up only	Picked up & delivered				
I	I	2	1	0.5	2	0	44	0	18	3769	671	8	6840
	II	2	1	1	2	0	44	0	18	3443	617	8	6460
	III	**2**	**1**	**1.5**	**2**	**0**	**44**	**0**	**18**	**3153**	**612**	**8**	**6165**
	IV	2	1	2	2	0	44	0	18	3586	685	8	6671
	V	2	1	2.5	2	0	42	3	15	2968	659	7	6977
	VI	2	1	3	2	3	41	5	12	2924	659	7	7303
	VII	2	1	3.5	2	5	39	6	12	3061	698	8	8009
	VIII	2	1	4	2	7	37	7	11	3505	571	9	9156
II	I	2	0.5	0.5	2	0	44	0	16	3769	775	8	7544
	II	2	0.5	1	2	0	44	0	16	3706	786	8	7492
	III	2	0.5	1.5	2	4	40	2	16	3797	896	8	8153
	IV	2	0.5	2	2	4	40	2	15	3896	964	8	8620
	V	2	0.5	2.5	2	2	42	6	11	3909	978	8	8867
	VI	2	0.5	3	2	6	38	10	8	4126	1236	9	10702
III	I	2	1.5	0.5	2	1	42	2	16	3669	617	8	7576
	II	2	1.5	1	2	1	43	2	16	3756	764	8	7410
	III	2	1.5	1.5	2	4	40	2	16	3797	964	8	8221
	IV	2	1.5	2	2	4	40	2	16	3443	906	8	7809
	V	2	1.5	2.5	2	6	38	2	16	3606	999	9	8745
	VI	2	1.5	3	2	9	35	4	14	4236	1369	9	10615
IV	I	2	2	0.5	2	1	43	0	18	3569	689	8	6848
	II	2	2	1	2	0	44	1	17	3797	764	8	7111
	III	2	2	1.5	2	4	40	2	16	3909	964	8	8333
	IV	2	2	2	2	4	40	4	14	3896	1096	8	8752
	V	2	2	2.5	2	6	38	6	12	3960	978	9	9678
	VI	2	2	3	2	9	35	6	12	4038	1236	9	10584
V	I	2	2.5	0.5	2	4	38	0	18	3614	697	8	8271
	II	2	2.5	1	2	4	38	0	18	3709	764	8	8433
	III	2	2.5	1.5	2	6	38	0	18	3666	938	8	8144
	IV	2	2.5	2	2	6	38	0	18	3705	869	8	8114
	V	2	2.5	2.5	2	6	38	2	16	3976	986	9	9102
	VI	2	2.5	3	2	9	35	4	14	4236	1345	9	10591

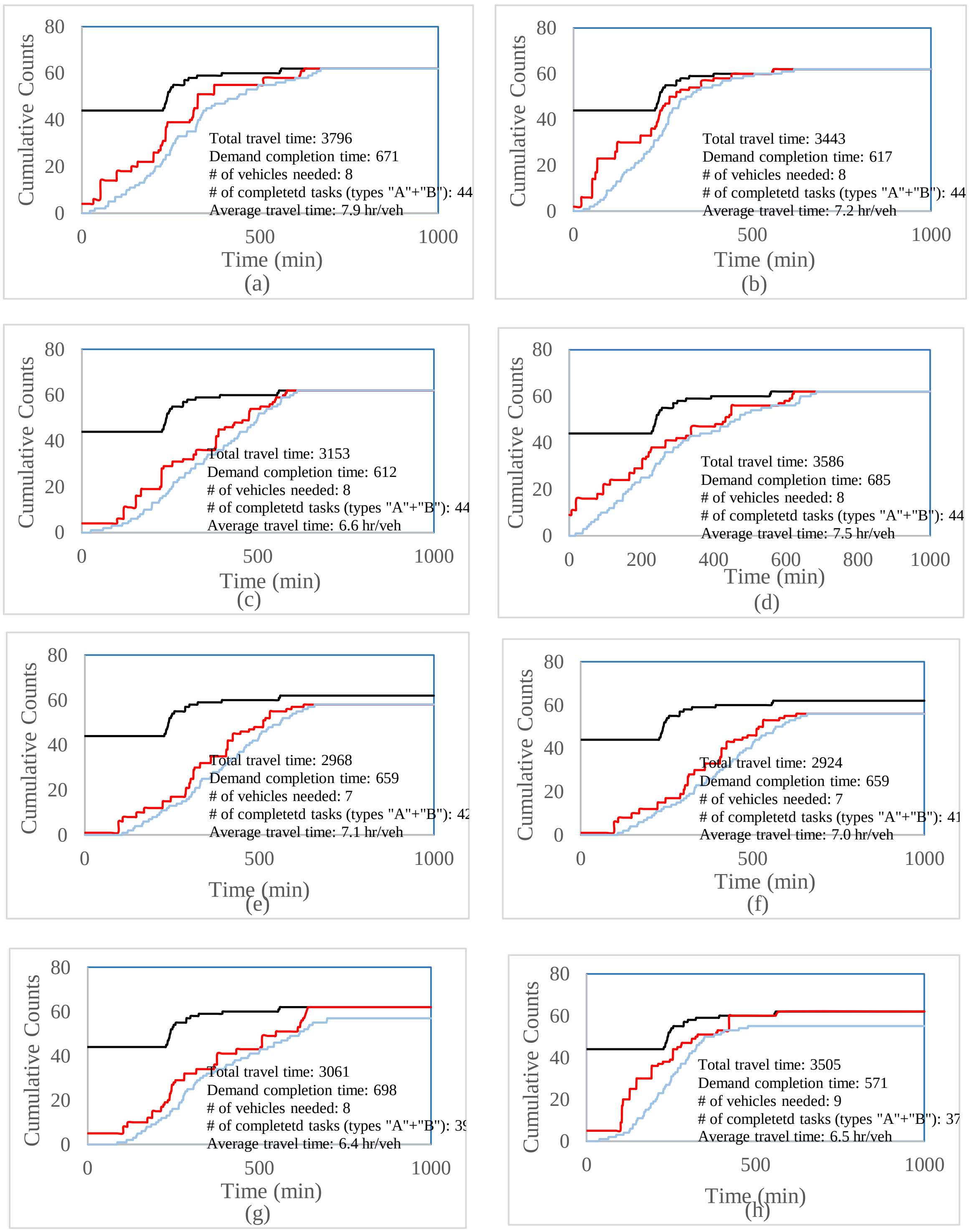


Fig. C.2. Cumulative arrival/departure as well as on board diagrams for scenario I (a) sub-scenario I, (b) sub-scenario II, (c) sub-scenario III, (d) sub-scenario IV, (e) sub-scenario V, (f) sub-scenario VI, (g) sub-scenario VII, and (h) sub-scenario VIII.

APPENDIX D

SMALL-SCALE SIX-NODE AND MIDIUM-SCALE SIOUX FALLS TRANSPORTATION NETWORK

In our experiments for the six-node transportation network and Sioux Fall network, the free-flow speed for all transportation links is assumed to be 60 mph. According to Fig. 5.2, the emission rate at 60 mph speed is roughly 0.3 per hour or 0.005 per minute. That is why 0.005 is assumed as the unit of resource in these experiments. Consider a physical transportation network consisting of six nodes presented in Fig. D.1. Each link in this network is associated with time-dependent travel time $TT(i,j,t)$. Without loss of generality, the number written on each link denotes the time-invariant travel time $TT(i,j)$ in terms of minutes. The length of time horizon is assumed to be 3 minutes, and nodes 0 and 3 are an individual's origin and destination, respectively.

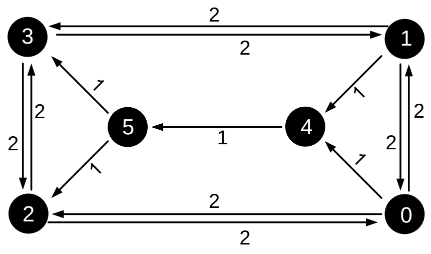


Fig. D.1. A six–node transportation network

In this network, the node sequence of the least cost path from the origin node (node 0) to the sink node (we assumed node 6 as the sink node) is 0, 4, 5, 3, 6; the time sequence is 0, 1, 2, 3, 4; the sequence of emission level index is 0.000, 0.005, 0.010, 0.015, 1.00; and the label sequence is 0.00, 0.005, 0.010, 0.015, and 0.015. The resource hyper-prism corresponding this example is presented in Table D.1.

Table D.1
Resource hyper–prism for an individual whose origin and destination are nodes 0 and 3, respectively.

Vertex No.	Node No.	Time stamp	Resource level	Summation of forward and backward labels
100	0	0	0.015	0.015
300	4	1	0.015	0.015
399	5	2	0.015	0.015
195	3	3	0.015	0.015
404	6	4	1.000	0.015

Sioux Falls network includes 24 nodes and 76 links illustrated in Fig. D.2. Similar to the six-node transportation network, the free-flow speed for all transportation links is 60 mph, the length of time horizon is 15 minutes, and nodes 9 and 14 are an individual’s origin and destination, respectively. We also assume that maximum emission production index is 0.065.

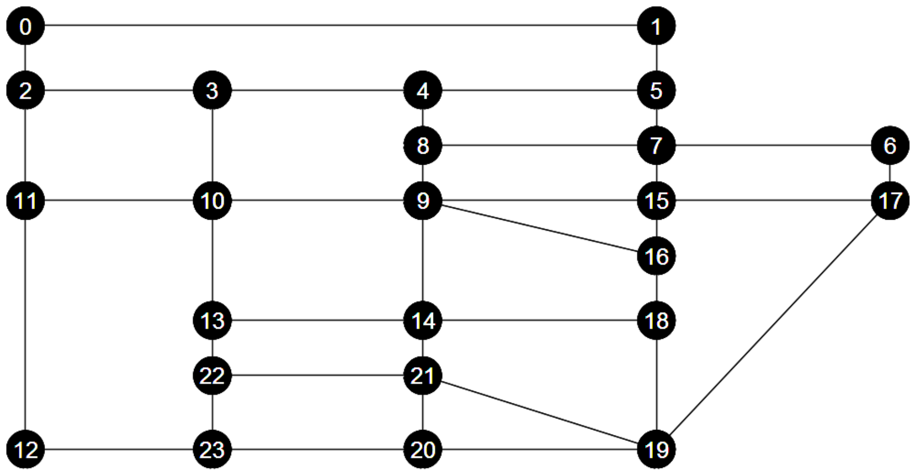


Fig. D.2. Sioux Falls network with 24 nodes and 76 links (all links are bi-directional).

Since we are not able to illustrate the four-dimensional RHP, we map it to the three-dimensional ST prism. The space-time prism considering the individual’s time and resource restrictions has been illustrated by Fig. D.3. According to Fig. D.3, some nodes in the ST prism are more accessible than others, or in other words, they are reachable in longer period of time.

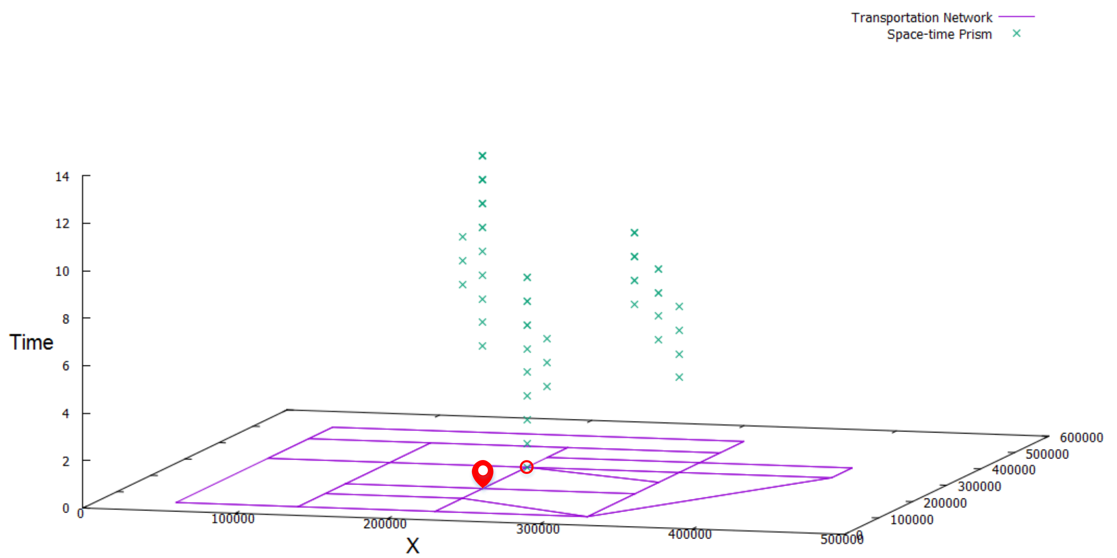


Fig. D.3. Space-time prism for Sioux Falls network considering the individual's time and resource constraints.

The CPU time for running our algorithm on this example is 1.4 seconds. In Fig. D.4, we specify the nodes that are accessible for both origin and destination respecting the individual's time and resource restrictions. Note that nodes 10 and 13 are accessible in terms of time, but not from the resource standpoint.

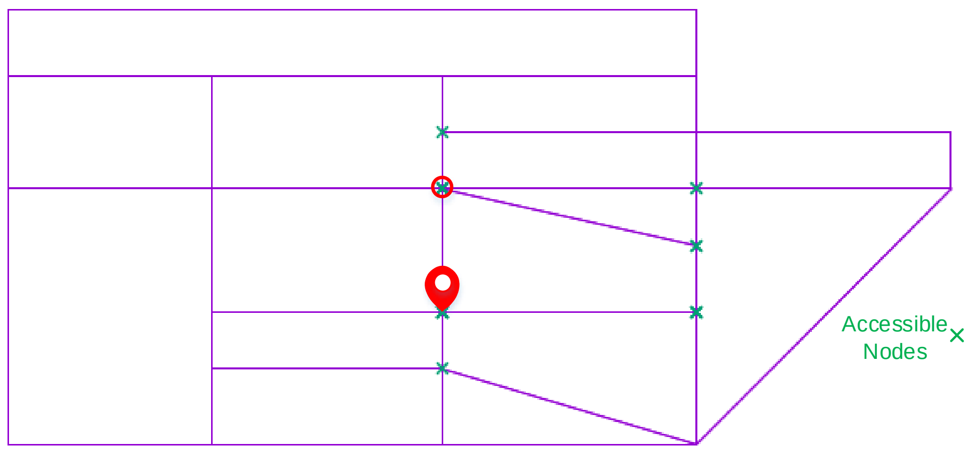


Fig. D.4. Accessible nodes for both origin and destination respecting the individual's time and resource constraints.