Carpooling Service for Large-Scale Taxicab Networks

DESHENG ZHANG and TIAN HE, University of Minnesota
FAN ZHANG and MINGMING LU, Shenzhen Institutes of Advanced Technology, China
YUNHUAI LIU, Third Research Institute of Ministry of Public Security, China
HAENGJU LEE and SANG H. SON, Daegu Gyeongbuk Institute of Science and Technology, Republic of Korea

Carpooling has long held the promise of reducing gas consumption by decreasing mileage to deliver coriders. Although ad hoc carpools already exist in the real world through private arrangements, little research on the topic has been done. In this article, we present the first systematic work to design, implement, and evaluate a carpool service, called *coRide*, in a large-scale taxicab network intended to reduce total mileage for less gas consumption. Our *coRide* system consists of three components, a dispatching cloud server, passenger clients, and an onboard customized device, called *TaxiBox*. In the *coRide* design, in response to the delivery requests of passengers, dispatching cloud servers calculate cost-efficient carpool routes for taxicab drivers and thus lower fares for the individual passengers.

To improve *coRide*'s efficiency in mileage reduction, we formulate an NP-hard route calculation problem under different practical constraints. We then provide (1) an optimal algorithm using Linear Programming, (2) a 2-approximation algorithm with a polynomial complexity, and (3) its corresponding online version with a linear complexity. To encourage *coRide*'s adoption, we present a win-win fare model as the incentive mechanism for passengers and drivers to participate. We test the performance of *coRide* by a comprehensive evaluation with a real-world trial implementation and a data-driven simulation with 14,000 taxi data from the Chinese city Shenzhen. The results show that compared with the ground truth, our service can reduce 33% of total mileage; with our win-win fare model, we can lower passenger fares by 49% and simultaneously increase driver profit by 76%.

Categories and Subject Descriptors: C.2.3 [**Network Operations**]; C.2.4 [**Distributed Systems**]; J.7 [**Computers in Other Systems**]

General Terms: Algorithms, Design, Experimentation, Verification

Additional Key Words and Phrases: Ridesharing, taxi network, fare model, application

This work was supported in part by the US NSF Grants CNS-1544887 and CNS-1446640; Global Research Laboratory Program (2013K1A1A2A02078326) through NRF; China 973 Program (2015CB352400); and Research Program of Shenzhen under Grants JSGG20150512145714248, KQCX2015040111035011, and CYZZ20150403111012661.

A preliminary work has been presented in ACM SenSys 2013 [Zhang et al. 2013].

Authors' addresses: D. Zhang, Department of Computer Science, Rutgers University, 110 Frelinghuysen Road, Piscataway, NJ 08854; email: zhang@cs.umn.edu; T. He, Department of Computer Science and Engineering, University of Minnesota, 200 Union Street SE, Minneapolis, MN 55455; email: tianhe@cs.umn.edu; F. Zhang and M. Lu, Shenzhen Institutes of Advanced Technology, China, 1068 Xueyuan Avenue, Shenzhen University Town, Shenzhen, P.R.China; emails: {zhangfan, lumm}@siat.ac.cn; Y. Liu, Third Research Institute of Ministry of Public Security, China, 110 Zhangheng Lu, 430 Nong, Shanghai, China; email: yunhuai@trimps.ac.cn; H. Lee and S. H. Son, Department of Information and Communication Engineering, Daegu Gyeongbuk Inst. Of Science and Technology (DGIST), 50-1 Sang-Ri, Hyeonpung-Myeon, Dalseong-Gun, Daegu, Korea; emails: haengjulee@gmail.com, son@dgist.ac.kr.

DOI: http://dx.doi.org/10.1145/2897517

ACM Reference Format:
Desheng Zhang, Tian He, Fan Zhang, Mingming Lu, Yunhuai Liu, Haengju Lee, and Sang H. Son. 2016. Carpooling service for large-scale taxicab networks. ACM Trans. Sen. Netw. 12, 3, Article 18 (August 2016), 35 pages.
DOI: http://dx.doi.org/10.1145/2897517

1. INTRODUCTION

Among all transportation modes, taxicabs play a particularly prominent role in residents' daily commutes in many metropolitan areas [Schaller Consulting 2006]. Based on a recent survey in New York City [Taxi and Commission 2011], over 100 taxicab companies operate more than 13,000 taxicabs, with stable demand of 660,000 passengers per day, and transport more than 25% of all transit passengers, accounting for 45% of all transit fares paid. To fulfill such delivery requests, these taxicabs travel a total of roughly 800 million miles per year [Schaller Consulting 2006]. Unfortunately, at 25MPG, these taxicabs consume about 32 million gallons of gas every year, more than the total annual gas consumption in some middle-sized countries (e.g., Central African Republic [Nationmaster 2011]), therefore leading to severely harmful tailpipe emissions and energy consumption. On the other hand, in carbon emissions trading under the Kyoto Protocol [Wikipedia 2016], governments will provide economic incentives for achieving reductions in the emissions of carbon pollutants. Thus, for both environmental and economic purposes, it is imperative to find a practical initiative to support the same delivery requests for taxicab transportation from passengers with lower total mileage and less carbon emissions.

In this article, we argue that a *taxicab carpool service* is a promising solution. The key advantage of a carpool service is that it can pool groups of several passengers heading in a similar direction into *one* rather than *several* taxicabs. In other words, a carpool service provides a valid solution for delivering the same number of passengers with lower total mileage and thus less gas. The economic incentive for drivers is that groups of passengers can pay a higher aggregated fare, whereas the incentive for passengers is that every passenger will pay less than in a noncarpool situation. With an effective fare model, we can achieve a win-win situation. Furthermore, carpools can also improve the availability of taxicab service during rush hours and after major events.

Admittedly, taxi carpools are not a new concept and have been around for years. But they are mostly negotiated by individual drivers and passengers in an ad hoc manner without a facilitating infrastructure. Until now, we have lacked a systematic study of carpooling in large-scale taxicab networks. We note that many studies have focused on taxicab scheduling [Balan et al. 2011; Ge et al. 2010; Huang and Powell 2012; Liu et al. 2011; Wu et al. 2012; Yuan et al. 2011b; Zhang et al. 2011] or novel systems taking advantage of taxicab mobile traces [Aslam et al. 2012; Biagioni and Eriksson 2012; Liu et al. 2012; Wei et al. 2012; Yuan et al. 2010, 2011a, 2012; Zhang et al. 2012; Zheng et al. 2011], but little research has been done on taxicab carpool services with a software and hardware codesign. In this article, we present the first systematic study of how to design, evaluate, and implement taxicab carpool services in real-world scenarios. Specifically, the key contributions of this article are as follows:

—To the best of our knowledge, we conduct the first carpool service that considers the mutual benefits for passengers and drivers in large-scale taxicab networks and provide a comprehensive study of how to fill the same passenger delivery requests with less total mileage and thus less gas consumption. To achieve our goal, we develop customized hardware, TaxiBox, with multiple sensors and onboard devices (such as CDMA communication module, MIC, camera, and carpool fare meters). Further, we develop a passenger client app, which is used to locate on-duty taxis and to send

carpool requests. Using these front-end devices, we design a taxicab carpool system, *coRide*, with a back-end server to gather requests from passengers, to inform drivers of carpool requests, to calculate carpool routes for drivers, and to estimate carpool fares for passengers.

—In *coRide*, we introduce a mathematical concept *delivery graph* to represent a carpool route schedule for delivering passengers. Given requests provided by passengers, we seek an optimal delivery graph to achieve the minimum total mileage. We show that this optimization is NP-hard by linking it to the classic traveling salesman problem and provide (1) an optimal solution with integer programming, (2) a 2-factor approximation solution with a polynomial complexity, and (3) an online algorithm to accommodate online streaming requests with a linear complexity. In addition, we consider different real-world constraints, for example, passenger travel periods, number of available taxicabs, and taxicab capacities.

—Based on the carpool route, we propose a win-win carpool fare model to encourage both drivers and passengers to participate in carpooling. In this model, given a carpool benefit due to mileage reduction, passengers and a driver will share this benefit based on a ratio dynamically adjusted by the supply and demand relationships in a taxicab network.

—To test the performance of *coRide* in a real-world setting, we implement *coRide* with a small-scale real-world trial in Shenzhen with three taxicabs and 12 passengers for 31 days. The results show that we reduce mileage by 49% compared to the ground truth without carpool services.

—To test the performance of *coRide* on large-scale systems, we use a real-world dataset consisting of GPS traces from more than 14,000 taxicabs in a Chinese city, Shenzhen, with a population of 13 million. The evaluation results show that compared with the ground truth, our carpool service reduces the total mileage by as much as 33%, and our win-win fare model lowers passenger fares by 49% and increases driver profit by 76% at the same time.

The rest of the article is organized as follows. Section 2 introduces the related work. Section 3 proposes the motivation for the design. Section 4 presents the *coRide* system overview. Section 5 depicts the passenger clients. Section 6 describes our customized device, TaxiBox. Section 7 explains the algorithms for carpool route calculation. Section 8 presents a win-win carpool fare model. Section 9 shows real-world small-scale implementation about our *coRide* system. Section 10 validates our services with a big dataset. Finally, we conclude this article in Section 11.

2. RELATED WORK

The premise of taxicab carpooling is not new, but in the real world it is normally negotiated privately by drivers and passengers in an ad hoc manner. We lack a systematic design to balance benefits of all the involved parties (e.g., drivers, passengers, and taxicab operators). Two types of previous work are directly related to our work: research on taxicab systems and on carpool services.

2.1. Taxicab Systems

The increasing availability of GPS devices has encouraged a surge of research intended to improve the efficiency of large-scale taxicab networks. First, several systems are proposed for the benefit of passengers or drivers, for example, allowing passengers to query the expected duration and fare of a planed taxicab trip based on the history of previous trips [Balan et al. 2011] and query real-time taxicab availability to make informed transportation choices [Wu et al. 2012], as well as recommending optimal pickup locations or routes [Ge et al. 2010, 2011; Yuan et al. 2011b]. Second, taxicab

traces can also help taxicab network operators better oversee taxicabs and provide efficient service to passengers, for example, discovering spatial and temporal causal interactions to provide timely and efficient service in certain areas with disequilibrium [Huang and Powell 2012; Liu et al. 2011], and detecting anomalous taxicab trips to discover driver fraud or road network changes [Zhang et al. 2011]. Third, traces from experienced taxicab drivers can help other drivers improve their driving performance, for example, navigating newer drivers to smart routes based on those of experienced taxicab drivers [Wei et al. 2012], and assisting other drivers to improve their driving performance with GPS records from experienced taxicab drivers [Yuan et al. 2011a]. Fourth, large-scale taxicab traces enable us to better understand traffic conditions of cities, for example, semantics of origin-destination flows [Zhang et al. 2012], traffic congestion and volumes [Aslam et al. 2012], and traffic patterns between regions with different functions [Yuan et al. 2012].

Yet existing research on taxicab systems focuses on scheduling individual taxicabs, assuming that one taxicab can accommodate only a single delivery request at a time. In contrast, our system allows shared delivery. Technically, we focus on carpool route calculation and a win-win fare model, neither of which has been investigated before.

2.2. Carpool Services

Limited ad hoc taxicab carpools exist in both developed and developing countries. For example, in New York City, up to four passengers can carpool together in a single taxicab ride from 6 AM to 10 AM on a weekday, along three preset routes in Manhattan at a flat fare of $3 or $4 per passenger, significantly less than the regular metered rates ["New York Times" 2010]. In Beijing, ad hoc taxicab carpooling is also allowed with the consent of both passengers and drivers, and every passenger pays 60% of the regular fare. Further, some door-to-door shuttle services are also available in major airports and can enable shared rides to or from airports [International 2015]. However, in the aforementioned carpool services, both time and locations are preset and the services are arranged on the spot by passengers or drivers in a small-scale ad hoc manner, and no infrastructure is provided to improve the efficiency of carpooling.

There are several pieces of theoretical work about carpools for private cars. The carpools for private cars are with regular passengers and fixed routes to or from work, which leads to a private classic carpool problem where one has to assign drivers to subsets of carpool participants to reduce the detour of drivers [Fagin and Williams 1983]. It is different from our carpool, where the drivers are not considered. Some work has been proposed to find the opportunities for carpool by finding the shared common routes by tracking personal mobile devices [Ananthanarayanan et al. 2009], but the whole system architecture and how to calculate the carpool route is not given. Fagin and Williams [1983] first present a simple carpool scheduling algorithm in which no penalty is assessed to a carpool member who does not ride on any given day. The proposed algorithm, the FW share, is shown to be fair, in a certain reasonable sense. Naor [2005] provides an axiomatic characterization of the fair share and indicates that the FW share is the unique one satisfying these requirements. Different from the classic carpool problem, our service focuses on how to pool passengers in different taxicabs to optimize the total mileage. In our service, passengers do not have a regular and fixed route, and drivers are not like the passengers who need to be considered. Several papers have been proposed after our conference paper [Zhang et al. 2013] to explore carpool services, for example, real-time carpooling [Ma et al. 2013], slugging form of carpooling [Ma and Wolfson 2013], and social effects of carpooling [Meurer et al. 2014; Han et al. 2014, 2016]. In contrast, our work is from system aspects to explore the carpool design issues with a hardware-software codesign with real-world implementation.

2.3. Vehicle Routing Problem

Our carpool route calculation problem is a special application of the vehicle routing problem (VRP) with time windows where the service can begin within the time window. Toth and Vigo [2001] provide an excellent work to explain VRP. The VRP with time windows can also be applied to school bus routing and scheduling. The review on the school bus routing problem can be found in Park and Kim [2010]. Because VRP is known to be NP-hard, many heuristics have been developed to solve large-scale VRP effectively. By relaxing different combinations of the constraints, we create several different types of subproblems that we can exploit. They include the assignment problem, knapsack problem, K-traveling salesman problem, and minimum spanning tree problem [Toth and Vigo 2001; Ahuja et al. 2014]. We rely on the minimum spanning problem to develop the approximation algorithm by relaxing taxicab capacity, the number of available taxicabs, and travel time period constraints.

Instead of a heuristic method to solve VRP, Desrochers et al. [1992] developed a new optimization algorithm to optimally solve a VRP with time windows. They solve the LP relaxation of the set partitioning formulation of the VRP with time windows, which provides an good lower bound for a branch-and-bound algorithm to solve the integer set partitioning formulation. However, relatively small-sized problems are solved by exact approaches, and heuristic approaches are preferred as problem sizes are bigger.

2.4. Real-World Services

Uber recently proposed a new service called Uberpool [Uber 2015], which uses real-time passenger requests to group several passengers together with similar origin and destination with a heuristic solution. Lyft is an another company that provides a ride-sharing service called Lyft Line [Lyft 2015] that matches passengers with other riders going in the same direction. There are many other startups (e.g., Ola Cabs, Didi Kuaidi, Haxi) competing in this lower-cost real-time ride-sharing market. To receive their carpooling service, passengers and drivers need to download their apps to their smartphones. When a passenger requests a ride, the system finds a match and the passenger splits the fare with another person. Trips are up to 50% less than their regular service without carpooling.

Despite all the benefits that the companies bring to the transportation market, there are many ongoing oppositions. First, there is an issue in safety. In order to accept the request, drivers have about 15 seconds to tap the phone to accept it, without regard to road conditions or safety [Richtel 2014]. It presents a significant distraction to drivers while driving. Unlike taxi drivers, drivers in Uber and Lyft are unlicensed and untrained, and they are not under taxi regulations. The second issue lies in an economic reason. Uber and other ride-sharing companies have brought a new challenges to the taxi industry. Taxi drivers claim that Uber presents unfair competition because it does not pay taxes or licensing fees [The Guardian 2014].

In contrast to their smartphone application facilitating peer-to-peer ride sharing, our work is from a system aspect to explore the carpool design issues with a hardware-software codesign with real-world implementation. To successfully address the aforementioned issues (i.e., safety, opposition by taxi drivers) with Uber and other companies, coRide is developed in close collaboration with the existing taxi industry in Shenzhen. Taxi drivers communicate with the dispatching center through a communication module, which is attached to TaxiBox. This is an extended communication module from the old one of GPRS (taxi drivers are already familiar with this usage) and is much safer than tapping the smartphone while driving. In addition, it employs a channel for voice communication to further enhance safety. In addition, coRide also benefits taxicab drivers because the aggregated carpool fare is higher than the regular service fare under our carefully designed win-win fare model. Therefore, we

Taxicab Network Summary	
Collection Period	6 Months
Collection Date	01/01/12-06/30/12
Numbe of Taxicabs	14,453
Number of Passengers	98,472,628
Total Travel Distance	594,031,428 (KM)
Total Fare	2,255,052,932 (CNY)
Average Travel Distance	6.032 (KM)
Average Fare	22.9 (CNY)

Fig. 1. Statistics.

believe that this will change the way taxi drivers (including taxi companies) regard ride-sharing programs. However, we emphasize that the carpool route calculation algorithms, integer linear programming, approximation algorithm, and online algorithm developed in this article do not have an application limit only for the taxi industry. They are general enough to be applied in any ride-sharing environments.

3. MOTIVATION

In this section, based on two datasets about traces and fares of 14,000 taxicabs in a Chinese city, Shenzhen, with a population of 10 million, we first introduce the basic properties of large-scale taxicab networks in dense urban areas. Then, we present evidence about the inefficiencies of current taxicab networks, demonstrate opportunities for carpools to address these inefficiencies, and identify challenges to facilitate carpooling in current taxicab networks. Details of the datasets appear in Section 10.

3.1. Properties of Taxicab Networks

In developed countries such as the United States, taxicabs are usually used to take passengers to airports, and personal vehicles are used for other activities, excepting extremely large cities such as New York City. But in developing countries, due to the high costs of owning personal vehicles, taxicabs and other public transportation are popular for daily activities. In dense urban areas such as Beijing, taxicabs are affordable for local traveling with an initial fare about $2 USD for a 3km trip, and are more comfortable than other public transportation with cheaper fares (such as buses or the subway). Due to the popularity and the affordability of taxicab services, the number of taxicabs in a taxicab network of a large city is typically more than 10,000. Thus, these taxicabs can be easily found on streets at the most of times and locations (except in rush hours or in hot pickup spots) and are commonly used for shopping, traveling to and from the work or schools, and other daily activities.

3.2. Inefficiencies of Taxicab Networks

We discuss the inefficiencies from three perspectives, that is, society, drivers, and passengers, based on a taxicab dataset shown in Figure 1.

3.2.1. Society. For society, the key inefficiency of taxicab networks is the large gas consumption of *a long travel distance*. We observe that these taxicabs travel a total of 1.2 billion kilometers per year, consuming about 100 million liters of gas to deliver 200 million passengers and causing harmful tailpipe emissions. Figure 2 gives the total travel distance of all taxicabs in the network on an hourly basis.

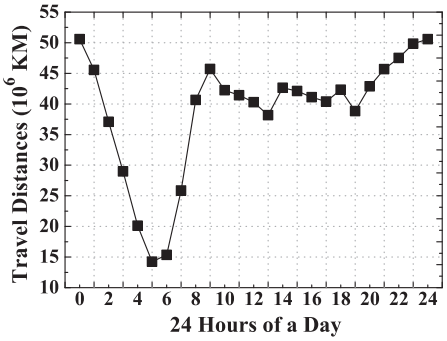

Fig. 2. Travel distance.

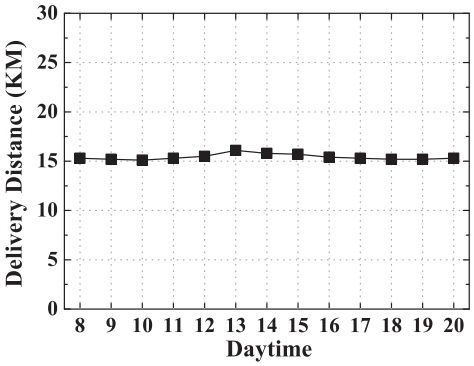

Fig. 3. Delivery distance.

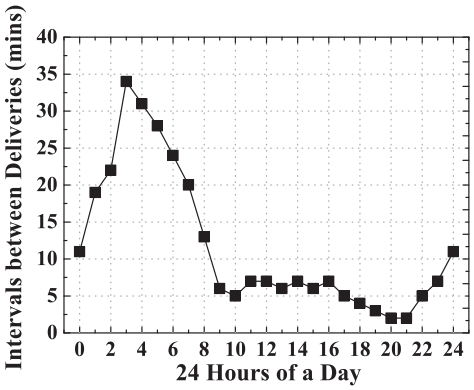

Fig. 4. Delivery intervals.

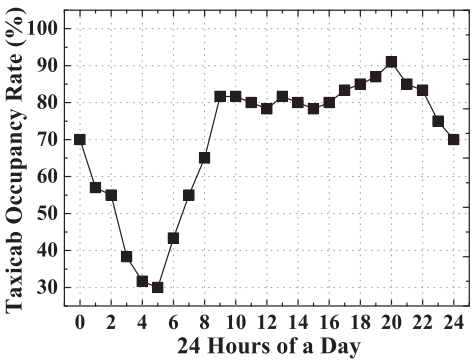

Fig. 5. Occupancy rate.

3.2.2. Drivers. For drivers, the key inefficiency is *low profits*, which are decided mainly by delivery distances (i.e., the mileage with paying passengers). Intuitively, drivers should earn more profit in rush hours, but this is not the case in large cities with severe congestion. In regular hours without congestion, the total distance (i.e., also including total mileage without paying passengers) is high, but the percentage of delivery distance is low, since it is not easy to find a passenger. In rush hours with congestion, in contrast, the percentage of delivery distance is high (i.e., easy to find a passenger), but the total distance is low due to the slow pace of traffic. Figure 3 shows the average delivery distance. It shows that delivery distances at different times of day (i.e., rush and non-rush hours) are not significantly different.

3.2.3. Passengers. For passengers, key inefficiencies are *high fares* and *low availability*. According to statistics about New York City [Schaller Consulting 2006; Taxi and Commission 2011], the average fare is $11.44 USD for a 2.8-mile trip with an 11-minute travel time, which is 5.8 times higher than the average public transit fare (bus or subway) on average. In our statistics, the average fare of 22.9 Chinese Yuan (CNY) for taxicabs is 11 times of a bus fare of 2 CNY on average. These two datasets also provide some evidence about the low availability of taxicab services. First, in the time intervals between deliveries presented in Figure 4, a small interval indicates that a taxicab will pick up a new passenger right after it drops off an old passenger, that is, low availability. In Figure 4, the average time interval in rush hours is small, less than

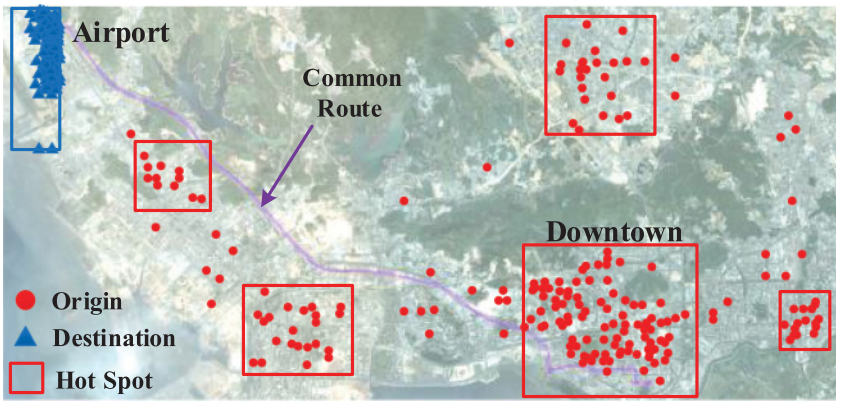

Fig. 6. Trips to an airport.

3 minutes, indicating low availability of taxicab services. Second, low availability can also be seen in the taxicab occupancy ratios in Figure 5, where a high occupancy ratio indicates fewer empty taxicabs. The ratios in Figure 5 indicate that more than 80% of taxicabs are occupied on average during rush hour. Thus, Figures 4 and 5 indicate the low availability of taxicabs during rush hour.

3.3. Opportunities for Taxicab Carpooling

We show the opportunities that carpools provide to address the aforementioned inefficiencies. Specifically, we discuss three factors to show how likely it is that carpooling services can be achieved in reality: (1) the distance between passengers' origins as well as the distance between passengers' destinations, (2) the travel distances of shared routes between passengers, and (3) the passenger preference to the carpooling services. The benefit of carpooling services can be further unleashed if we have more passengers who (1) start from close origins or end at close destinations, (2) share the long-distance common routes, and (3) are willing to accept carpooling services.

3.3.1. Close Origins and Close Destinations. Based on the dataset, we show 200 consecutive trips to an airport during a 32-minute time window in Figure 6 where most passengers came to the airport from downtown and several hot spots.

Similarly, we show 200 consecutive trips from an airport during a 29-minute time window in Figure 7. We observe the similar phenomenon that the most passengers went the downtown and several hot spots from the airport.

In Figure 8, we show the CDF of distances between the origins of 1,000 trips to the airport. Similarly, almost 50% of the trips have an origin closer than 1km to another origin, and almost 90% of trips have an origin closer than 5km to another origin. In Figure 9, we show the CDF of distances between destinations of 1,000 trips from the airport. Almost 60% of trips have a destination closer than 1km to another destination, and almost 80% of trips have a destination closer than 5km to another destination.

3.3.2. Shared Routes. Based on the dataset, Figure 10 shows the CDF of distances of shared routes of 1,000 trips to the airport. More than 90% of trips share at least 7.5km with another trip, and more than 50% of trips share at least 20km with another trip. Figure 11 shows the CDF of distances of shared routes of 1,000 trips from the airport. Similarly, more than 90% of trips share at least 5km with another trip, and more than 50% of trips share at least 20km with another trip.

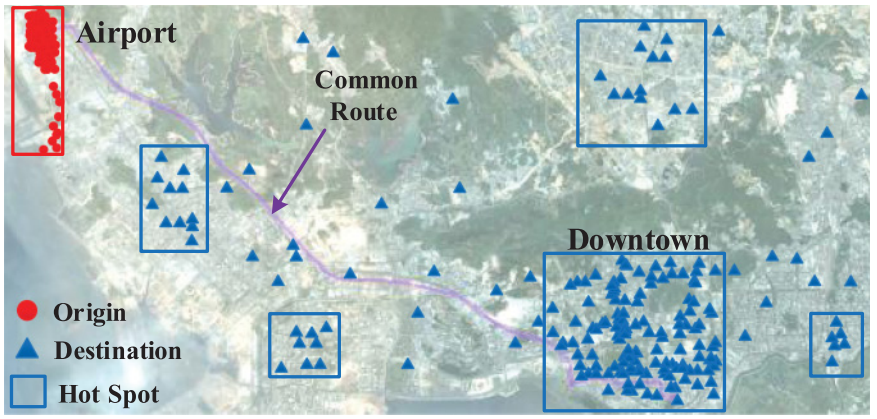

Fig. 7. Trips from an airport.

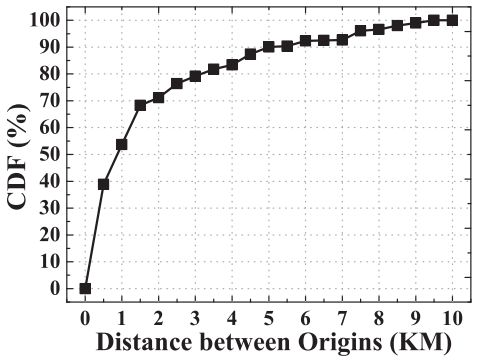

Fig. 8. Close origins.

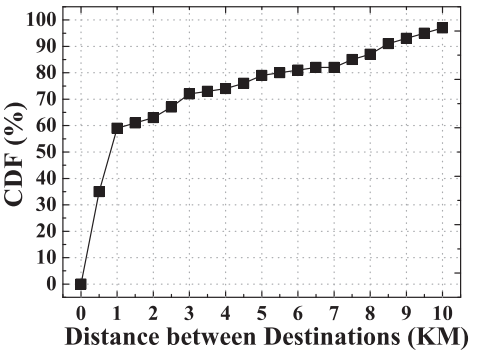

Fig. 9. Close destinations.

3.4. Benefits for Taxicab Carpool

From these figures, we observe a good opportunity for carpools to benefit multiple parties. For passengers, a taxicab carpool can increase the availability of taxicab services in extreme weather, peak hours, or hot pickup locations, reducing the waiting time for passengers; in addition, multiple passengers in a carpool can share the fare, reducing the fare paid by individual passengers. For taxicab drivers, a taxicab carpool can increase profits, since the aggregated carpool fare is higher than the regular service fare with the same travel distance. For operators, a taxicab carpool can provide more transportation capacity and enable more efficient gas consumption. Note that taxicab carpools do not aim to completely replace the traditional taxicab services, but serve as a key supplement for situations where regular taxicab services are insufficient in peak hours or extreme weather, or situations where some passengers would like to take transportation that is cheaper than traditional taxicab services yet more convenient than the bus and subway.

3.5. Passenger Sensitivity for Taxicab Carpool

Although taxi carpooling is not well supported at regulatory levels in many countries, according to a taxi pooling survey taken in Beijing [Data100 Company 2012], 75% of interviewees accept carpooling services, 57% of interviewees have carpooled with others

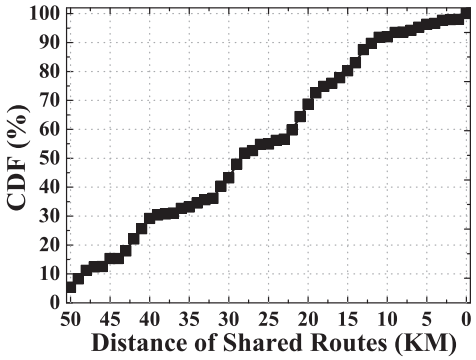

Fig. 10. Shared distance to airport.

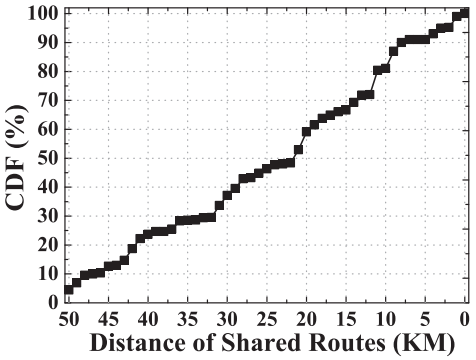

Fig. 11. Shared distance from airport.

at least once, and 73% of interviewees accept a simple carpooling price mechanism where every passenger pays 60% of the regular service fare for the shared distance, leading to extra profits for drivers; several key concerns about carpooling pointed out by more than half of interviewees are as follows: (1) prolonged travel time (64%), (2) hard to find passengers to carpool or hard to find carpoolable taxicab (50%), and (3) unable to print duplicated receipts for all passengers (50%). Based on this survey, we find that most passengers are willing to accept carpools and to share the benefits of carpools with coriders and the driver, but we still face several challenges to enable a practical carpool service in large-scale taxicab networks, which we will introduce next.

3.6. Challenges for Taxicab Carpool

We present three challenges and some possible solutions for implementing carpool services in the current taxicab networks.

Acquisition of Detailed Status About Taxicabs: To find the most suitable taxicab, a dispatching center should acquire detailed information about taxicabs, for example, how many seats are left, which is difficult to obtain in current taxicab networks, where only the general taxicab status (e.g., locations, speeds, with passenger or not, etc.) can be obtained by dispatching centers through real-time GPS record uploading. Thus, an onboard device should be installed in taxicabs to let dispatching centers monitor the detailed status of taxicabs and find the most suitable taxicab.

Carpool Route Calculation: After finding the most suitable taxicab, a dispatching center should calculate the optimal carpool route based on multiple received requests in a centralized way and send the calculated route to a driver. This route should be efficient in terms of the total distance to deliver all assigned passengers. In addition, the calculation of routes should be fast enough to enable a responsive taxicab carpool service.

Fare Estimation and Calculation: With a carpool route schedule, dispatching centers should notify passengers with fare details of several carpool options for their approval. A win-win carpool fare model that estimates fares is missing in the taxicab business. Further, current fare meters can calculate only a single fare, and a more advanced fare meter that can record multiple concurrent trips is desirable for carpooling.

To address these challenges, we aim to develop a carpool system, *coRide*, as a hardware and software codesign with a passenger client app and a front-end onboard device called *TaxiBox*, along with a back-end cloud server to upgrade current taxicab networks. *coRide* employs multiple sensors and external devices attached to *TaxiBox* to effectively manage taxicab networks. We will provide an overview of *coRide*, passenger clients, and TaxiBox design in Sections 4, 5, and 6, respectively.

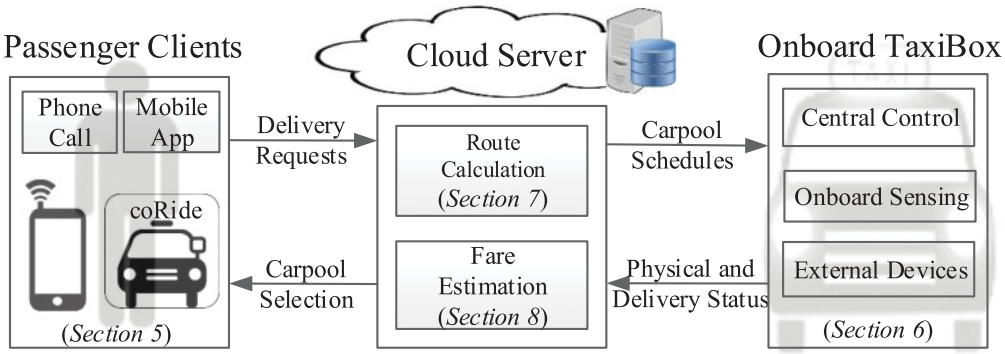

Fig. 12. *coRide* system overview.

4. *CORIDE* SYSTEM OVERVIEW

In this section, we present an overview of *coRide*, which consists of three key parts: a cloud server, passenger clients, and the onboard TaxiBox as shown in Figure 12.

4.1. Passenger Clients

Passenger participation is required by our design, which can be encouraged by our win-win fare model discussed later. Assuming that passengers will be willing to participate, they will provide delivery requests to the dispatching center. The most common way to provide delivery requests is to call the dispatching center by phone to provide the number of passengers, pickup time, origin, destination, and possible delivery deadline. Further, mobile apps can be used to provide delivery requests without calling the dispatching center. Based on the delivery requests provided by passengers, the dispatching center will return a carpooling option with a reduced fare for their approval, along with a noncarpool option with a regular fare for comparison. An example of such passenger clients is given in Section 5.

4.2. Onboard TaxiBox

When a carpool is approved by passengers, a dispatching center will locate a suitable taxicab for the carpool based on the current status of taxicabs and then send a carpool route schedule to this taxicab's TaxiBox. The driver will respond to this carpool request by changing the status of the taxicab and then performing the carpool route schedule. These functions are performed by three key components of TaxiBox, which will be introduced in detail in Section 6.

4.3. Dispatching Cloud Server

In this article, we will focus on function designs for a cloud server at dispatching centers with an emphasis on taxicab carpool services rather than regular services. In our carpool service design, a cloud server is mainly in charge of

(i) receiving delivery requests from passengers,
(ii) calculating carpool routes based on delivery requests,
(iii) estimating carpool fares for passengers to approve,
(iv) sending carpool routes to suitable taxicabs, and
(v) obtaining the physical and delivery status of taxicabs.

Fig. 13. App screenshot.

In the rest of the article, we present the detailed passenger client and TaxiBox design in Sections 5 and 6; for the dispatching cloud server, we describe two key functions, the carpool route calculation and fare model in Sections 7 and 8, respectively.

5. PASSENGER CLIENTS

We embed *coRide* as a carpooling service into one of our taxicab-booking apps for taxicab passengers using the taxicab network in Shenzhen. In Figure 13, we show the app screenshots about on-duty taxicab checking, carpooling request submitting, and phone calls for services.

For the on-duty taxicab checking function, based on the access of real-time data of taxicabs in Shenzhen, we analyze both locations and status of all taxicabs and show them with different icons based on their status on a Shenzhen city map. For the carpooling function, based on the location and time entered by users, the app sends a carpooling request to the cloud server, which returns a carpooling schedule based on the status of taxis. The current version of our app is in testing based on the regulation of Shenzhen city. Although our app is specifically designed for Shenzhen, the key functions (e.g., on-duty taxi checking and carpooling request submitting) can be generalized to other cities as well.

6. ONBOARD TAXIBOX

In this section, we present our hardware design, then show the deployment of TaxiBox, and finally propose the capability of taxicabs with TaxiBox.

6.1. TaxiBox Design

As shown by Figure 14, our onboard device, TaxiBox, consists of three main parts: central control system, onboard sensing system, and external devices.

The central control system has two key parts, the power module and the CPU module. For the power module, we employ TPS54160 from Texas Instruments, which is a 60V, 1.5A, step-down SWIFT DC/DC converter with an integrated high-side MOSFET. For the CPU module, we use a 32-bit 72MHz processor STM32F103 from ARM Cortex-M3 processors with A/D Convertors of 12-bit accuracy.

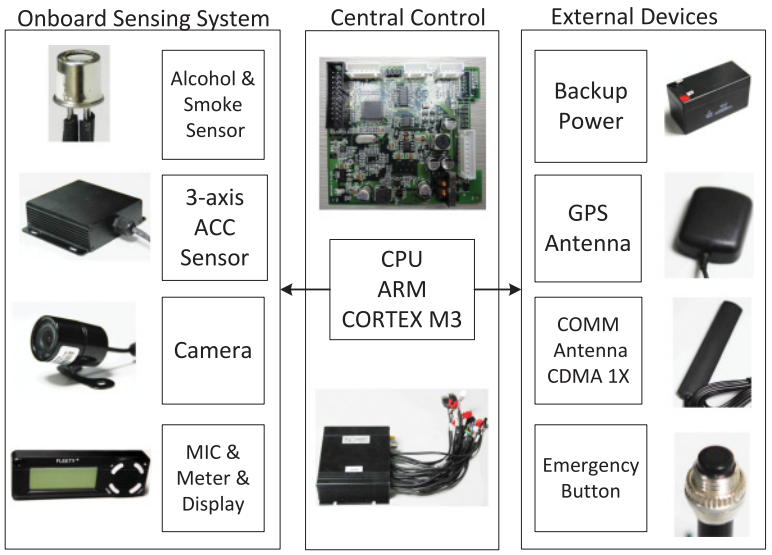

Fig. 14. TaxiBox hardware design.

The onboard sensing system has open interfaces to multiple sensors, and the current hardware has (1) alcohol and smoke sensors, (2) a $\pm 2g$ triaxial acceleration sensor, and (3) a camera and a microphone. Based on these sensors, a dispatching center is capable of monitoring the comprehensive physical status of a taxicab on streets.

Various external devices can be integrated into our TaxiBox. Some external devices in the current TaxiBox design include (1) a display and a speaker integrated to the display, (2) a traditional fare meter for fare calculation and receipt printing, (3) backup power for a situation in which the main power is not available, (4) an emergency button, (5) a GPS module with a separate GPS antenna, and (6) a CDMA 1X communication module with a separate antenna.

In some existing taxicab networks, the communication modules usually use GPRS (e.g., for GPS coordinates uploading) between taxicabs and dispatching centers. But in our design, taxicabs typically have a larger dataset to upload to or download from a dispatching center. Thus, a CDMA 1X, instead of GPRS, communication module is attached to TaxiBox, since CDMA employs different channels for voice and data communications, which clearly has advantages in terms of communication speed and stability, compared to GPRS that employs the same channel for data communications.

6.2. TaxiBox Deployment

We have deployed our TaxiBox in 98 taxicabs as shown by Figure 15. The alcohol and smoke sensors are installed in the ceiling of taxicabs for better sensor functions. The camera is in front of passengers so as to take pictures from a better angle. The main part of TaxiBox is hidden above the glove box. The display is installed above the air conditioner control panel for easier access by drivers. The three-axis acceleration sensor is hidden under the glove box.

6.3. TaxiBox Capability

In this section, based on the hardware we deployed in taxicabs, we introduce the capabilities of TaxiBox.

Fig. 15. TaxiBox deployment.

Taxicab Physical Status Sensing: Dispatching centers should be fully aware of the status (e.g., location, speed, etc.) of taxicabs to provide better carpool service. With GPS and CDMA 1X modules onboard, a taxicab can periodically upload its real-time physical status to a dispatching center. The onboard traditional fare meter and TaxiBox with a display can function together as a smart meter that can record the status of several trips (i.e., the delivery distance and fare for different passengers onboard), whereas the traditional fare meter can only record a single delivery distance. Further, a speaker is integrated into the display so dispatching centers can issue a voice schedule or voice navigation.

Taxicab Delivery Status Sensing: In addition to a taxi's physical status, a dispatching center is also interested in the real-time status of its deliveries. The status of deliveries includes delivery distances, with passengers or not, fare, duration, start time, end time, and pickup and dropoff location, which all can be obtained by TaxiBox and uploaded to dispatching centers.

7. CLOUD SERVER

In this section, we focus on the cloud server design in terms of carpool route calculation. We first propose preliminaries about our carpool work, then define a carpool route calculation problem, and finally propose its solution.

Carpools can be classified into four categories:

(i) One origin to one destination ($1 \rightarrow 1$)
(ii) One origin to many destinations ($1 \rightarrow N$)
(iii) Many origins to one destination ($N \rightarrow 1$)
(iv) Many origins to many destinations ($N \rightarrow N$)

For the sake of presentation, we will focus on $1 \rightarrow N$ because $1 \rightarrow 1$ is a special case of $1 \rightarrow N$, and $N \rightarrow 1$ can be solved with $1 \rightarrow N$ by reversing origins and destinations. As for the $N \rightarrow N$ case, we can treat it as a special $1 \rightarrow N$ case where we add several constraints to make sure all the origins are visited before their corresponding destinations. For example, we can have a virtual origin and treat all other origins

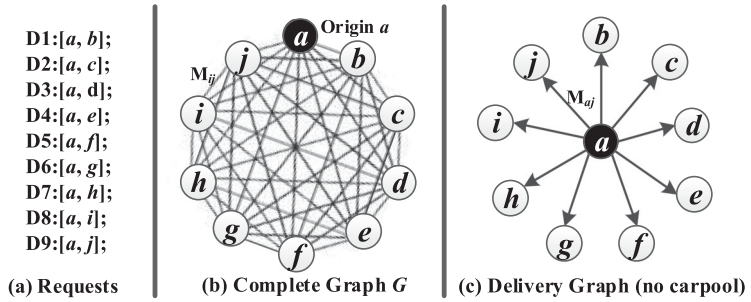

Fig. 16. Complete graph.

and designations as virtual designations, but the visiting order will be constrained by the original origin-destination pairs where a destination cannot be visited unless its corresponding origin has been visited. Without loss of generality, we use the $1 \rightarrow N$ model (e.g., carpool passengers from an airport) as an example for the design.

7.1. Preliminaries

For a carpool, a passenger will provide a *delivery request* with an origin, a destination, a start time, and an optional end time (a possible end time serves as a deadline for delivery, but our model works with an unknown end time). Thus, given several requests for carpooling from the same origin as in Figure 16(a), we shall analyze distances between their destinations, which can be shown as a complete graph. We construct this complete graph as shown in Figure 16(b) by (1) treating both origin and destinations as vertices and (2) linking all vertices to each other with directed edges, associated edge weights with pairwise mileage costs.

Figures 16(a) and 16(b) give an example of how to create a complete graph based on nine delivery requests from the origin a. A weight on an edge (e.g., M_{ij}) indicates the real-world mileage between two locations. Given the complete graph, we can obtain a carpool route based on a *delivery graph*, which is defined as follows.

Definition 1 (*Delivery Graph*). With a complete graph G given by delivery requests, a *delivery graph* is a subgraph of G where (1) the origin vertex can reach all destination vertices and (2) no branches exist at any vertex but the origin vertex (i.e., spoke topology).

With this definition, we can see that a delivery graph uniquely indicates a carpool route where the total carpool mileage is equal to the sum of all its edges' weights. In Definition 1, the condition (1) is to make sure that with a carpool route, all passengers can be delivered from the origin to their destinations; the condition (2) is to make sure that every passenger will take only one taxicab during the carpool, that is, without relay. Figure 16(c) gives an example of a delivery graph without carpool, that is, all passengers are delivered by separate drivers with separate mileages, for example, M_{aj}.

The examples of a delivery graph DG with carpool are given in Figure 17(a). In DG, the origin vertex a can reach all destination vertices, and no branches exist at any destination vertex. A delivery graph (e.g., DG) indicates a real-world carpool by specifying (1) a *passenger assignment* for taxicabs and (2) a *delivery order* for a taxicab's passenger assignment. For example, DG shows that three taxicabs fulfill passenger requests with destinations on three paths (the total weight on edges of a path indicates the real-world mileage): Taxi 1 delivers passengers to b, with a mileage M_{ab}; Taxi 2 delivers passengers to c, d, e, and f, with a mileage $M_{ac} + M_{cd} + M_{de} + M_{ef}$; Taxi 3 delivers passengers to g, h, i, and j, with a mileage $M_{ag} + M_{gh} + M_{hi} + M_{ij}$.

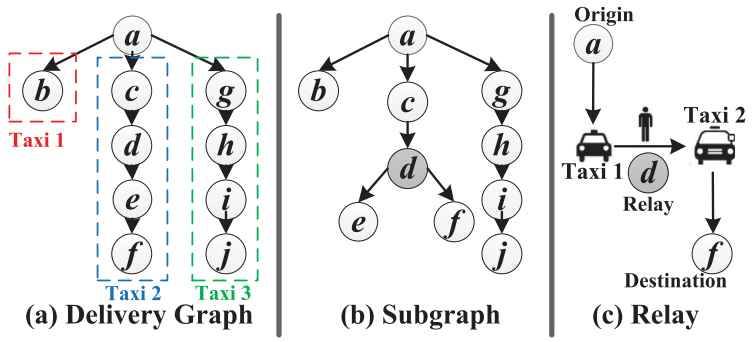

Fig. 17. Delivery graph with carpool.

Figure 17(b) gives an example of carpools prohibited by *coRide*. To carpool by this subgraph, we have to at least use two taxicabs to deliver passengers to c, d, e, and f: the first taxicab delivers passengers with destination c, d, and e, but carries only the passenger with destination f from origin a to an intermediate vertex d; the second taxicab has to pick up this passenger at vertex d (a relay) and then deliver him or her to destination f as shown in Figure 17(c).

We argue that the carpool service with a relay is not practical in the real world, because (1) the relayed passenger has to pay multiple times to different drivers, and (2) the coordination between taxicabs would lead to a large layover delay. Therefore, *coRide* supports only the delivery graphs with spoke topology to indicate a practical real-world carpool route schedule for drivers to deliver passengers without a relay.

7.2. Carpool Route Calculation Problem

Based on the delivery graph proposed in the last subsection, we propose our carpool route calculation problem: **given a complete graph based on delivery requests, find the minimum-weight delivery graph.**

The complete graph can be easily constructed based on delivery requests provided by passengers; as a subgraph, a delivery graph specifies passenger assignments and delivery orders to fulfill all delivery requests; the minimum total weight of a delivery graph indicates that it fulfills all requests with a carpool spending the minimum total mileage. To perform a practical carpool, we also consider three constraints as follows:

(i) ***Taxicab Capacity c***: it shows how many passengers can be pooled into one taxi.
(ii) ***Number of Available Taxicabs n***: it shows how many taxicabs can be used for carpool at the origin.
(iii) ***Travel Period*** $[t_i^s, t_i^e]$: it shows the earliest pickup time t_i^s and the latest dropoff time t_i^e for a delivery request i.

Based on this discussion, our carpool route calculation problem is related to a *multiple traveling salesmen problem* (called mTSP, where multiple salesmen start from a depot to visit different cities with the minimum total distance [Oberlin et al. 2009]) with the special carpool constraints. An mTSP is generally solved with integer programming to the optimal solution. But for our large-scale setting, the optimal solution results in a long running time, since it is NP-hard. Thus, an approximation algorithm should be used to obtain a delivery graph within a reasonable time.

Another key feature of our carpool route calculation problem is that instead of booking a carpool trip a day or two in advance, some passengers may provide *online* delivery requests just tens of minutes before the starting time of their deliveries. So an *online*

algorithm is also necessary. Therefore, the design agenda about our solution to the carpool route calculation problem is given as follows:

(i) We use integer linear programming to formulate our carpool route calculation to obtain the optimal solution in Section 7.3.
(ii) For a practical (faster) solution, we propose a 2-factor approximation algorithm to obtain a suboptimal solution in Section 7.4.
(iii) To consider online requests, we present our online algorithm in Section 7.5.

7.3. Optimal Solution

In the literature, the optimal solution for mTSP is given by integer linear programming. Thus, we formulate our Carpool Route Calculation with following definitions:

(1) $G = (V, A)$: a weighted complete graph where vertex a is the origin vertex where a carpool starts and $V' = V - \{a\}$ is the set for destinations, and a weight on A indicated as c_{ij} is the real-world mileage cost from vertex i to vertex j
(2) $x_{ij} = 1$ if edge $(i, j) \in A$ is used; $x_{ij} = 0$ otherwise
(3) $[\boldsymbol{t_i^s}, \boldsymbol{t_i^e}]$: a travel period for a passenger to vertex i
(4) $\boldsymbol{n}$: the number of available taxicabs
(5) $\boldsymbol{c}$: the taxicab capacity
(6) $\boldsymbol{y_i}$: total number of dropped passengers before vertex i
(7) $\boldsymbol{q_i}$: total number of dropped passengers at vertex i
(8) $\boldsymbol{p_i}$: time arriving at vertex i
(9) $\boldsymbol{w_i}$: latest start time of dropped passengers before vertex i
(10) $\boldsymbol{T(i, j)}$: travel time between vertex i and vertex j

$$\min \quad \sum_{(i,j)\in A} c_{ij}x_{ij}$$

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

$$\sum_{j\in V} x_{ij} \in \{0, 1\} \qquad \forall i \in V' \tag{2}$$

$$\sum_{j\in V'} x_{aj} \le n \qquad \sum_{i\in V'} x_{ia} = 0 \tag{3}$$

$$y_i + q_i \le c \qquad \forall i \in V' \tag{4}$$

$$\text{If } x_{ij} = 1 \Rightarrow y_i + q_i \le y_j \qquad \forall i, j \in V' \tag{5}$$

$$p_i \le t_i^e \qquad \forall i \in V' \tag{6}$$

$$\text{If } x_{ij} = 1 \Rightarrow p_i + T(i, j) \le p_j \qquad \forall i, j \in V' \tag{7}$$

$$\max\{w_i, t_i^s\} + T(a, i) \le t_i^e \qquad \forall i \in V' \tag{8}$$

$$\text{If } x_{ij} = 1 \Rightarrow w_i \le w_j \qquad \forall i, j \in V' \tag{9}$$

$$\sum_{i\notin S}\sum_{j\in S} x_{ij} \ge 1 \qquad \forall S \subseteq V' \tag{10}$$

$$x_{ij} \in \{0, 1\} \qquad \forall i, j \in V, \tag{11}$$

where Equation (1) ensures that exactly one taxicab visits a destination; Equation (2) ensures that exactly one taxicab leaves one destination for the next delivery, or the carpool is over and no delivery needs to be made; Equation (3) is about the constraint

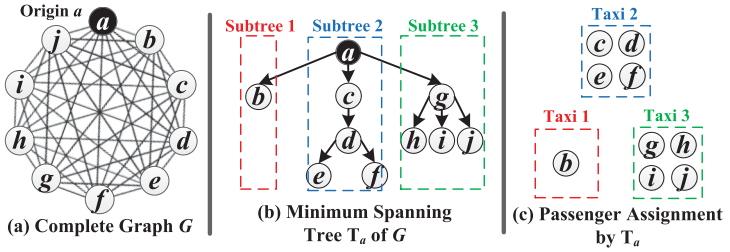

Fig. 18. Passenger assignment.

on the number of available taxicabs; Equations (4) and (5) are about the taxicab capacity constraint; Equations (6), (7), (8), and (9) are about the travel period constraint; and Equation (10) is to prevent the formation of subtours not including origin vertex a. Note that though a taxicab has disjoint vertices in a delivery graph, they can share the same route in the real world when performing a carpool.

Since the traditional traveling salesmen problem is NP-hard, as a generalized version, our problem is also NP-hard (due to space constraint we omit the formal proof). Therefore, when the number of destinations increases, the running time to solve the previous integer programming increases exponentially. Although integer linear programming is sufficient for a small number of destinations, we need to accommodate the case where the number of destinations is large with an efficient algorithm.

7.4. Approximation Algorithm

We first propose an approximation algorithm and discuss impacts of three constraints.

7.4.1. Approximation Algorithm Without Constraints. An approximation algorithm without constraints is under a scenario where all passengers' travel periods are not considered; the origin has unlimited taxicabs with unlimited capacities.

We first present some rationales. Any carpool from the same origin can be performed with two key steps: (1) we shall *assign passengers* to different empty taxicabs and (2) we shall calculate a *delivery order* for a given passenger assignment for a particular taxicab. In the following, we will describe how to make use of this rationale for our approximation algorithm in two steps and provide its approximation ratio.

(i) **Passenger Assignment**: Based on a complete graph G created with delivery requests, we will show how to assign passengers in Figure 18.

To assign passengers to different taxicabs, we shall take into account the distances between destinations given by G in Figure 18(a). The objective is to find a *minimum-weight* subgraph of G to assign destination vertices to different paths (every path is used by a unique taxicab). Since the minimum spanning tree (MST) is the minimum weight subgraph of G, in this article we try to employ an MST to obtain a passenger assignment. Figure 18(b) gives a G's MST T_a with three subtrees rooted at origin vertex a. Based on T_a, we assign the passengers, who have destination vertices in the same subtree, into the same taxicab. For example, passengers with destinations c, d, e, and f are assigned into Taxicab 2, as in Figure 18(c). Note that T_a is not a delivery graph we try to obtain, because T_a has branches at destination vertices. Thus, we have shown how to conduct passenger assignment based on a given complete graph.

(ii) **Delivery Order Calculation**: Based on the assignment in step (i), in Figure 19 we show how to calculate a delivery order for passengers in the same taxicab.

As in step (i), a passenger assignment for a particular taxicab is given by a subtree rooted at the origin vertex a. Thus, we employ Subtree 2 (named ST) in the MST T_a in Figure 19(a) as an example to show how to obtain a delivery order. With an observation

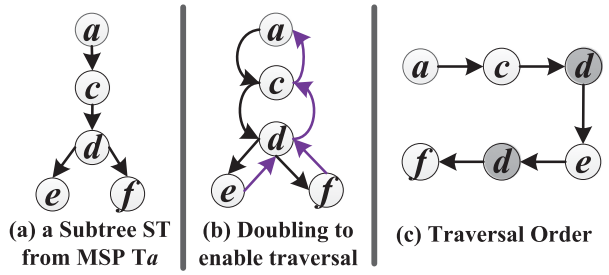

Fig. 19. Delivery order calculation.

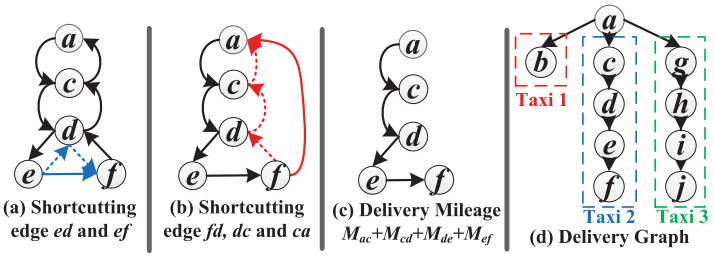

Fig. 20. Shortcutting about duplicated vertices.

on ST, we found that ST only gives a passenger assignment, but not a fixed delivery order since ST has a branch that requires passenger relay, which is prohibited by *coRide*. Thus, a new subgraph transformed from ST should be created to calculate an order without relay. In this article, we use a depth-first traversal from root vertex to decide a delivery order. But as we can see in Figure 19(a), ST is a directed graph and cannot be traversed based on current edges. Thus, as in Figure 19(b), we double the edges in ST to create loops to enable a traversal $a \rightarrow c \rightarrow d \rightarrow e \rightarrow d \rightarrow f$. This order is not a delivery order since it involves duplicated vertices (i.e., d), and thus a longer total mileage $M_{ac} + M_{cd} + M_{de} + M_{ed} + M_{df}$.

To obtain a delivery order, we use a shortcut strategy to eliminate duplicated vertices in a traversal. In Figure 20, we show how to shortcut some edges about duplicated vertices, thus further reducing delivery mileage.

As in Figures 20(a) and 20(b), we shortcut edges ed and df with a new edge ef, and then shortcut edges fd, dc, and ca with another new edge fa. Note that based on triangle inequality, the length of an added edge (e.g., M_{ef}) is always shorter than the sum of edges it is shortcutting (*e.g.*, $M_{ed} + M_{df}$). Further, we delete the edge fa to obtain the delivery order $a \rightarrow c \rightarrow d \rightarrow e \rightarrow f$ and the total mileage cost of four edges ($M_{ac} + M_{cd} + M_{de} + M_{ef}$) as in Figure 20(c). Therefore, with a traversal, we have shown how to calculate a delivery order based on a given passenger assignment. With these two steps, we finish our approximation algorithm to obtain our delivery graph in Figure 20(d).

Proof of Approximation Ratio: We have proved that our traversal algorithm has a constant performance ratio of 2; that is, the total mileage obtained by our carpool schedule is at most 2 times the optimal mileage we obtained by the optimal solution using integer programming. This is because (1) with shortcutting, the weight of our delivery graph $W(S)$ is smaller than a weight of a Traversal $W(T')$, that is, $W(S) < W(T')$; (2) a traversal is exactly 2 times that of an MST, $W(T') = 2W(T)$; and (3) the MST is smaller than or equal to the optimal solution since the optimal solution

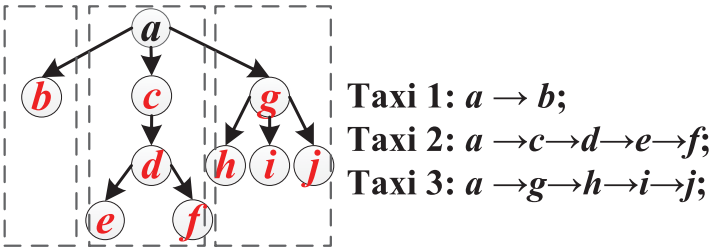

Fig. 21. Minimum number of taxicabs for deliveries.

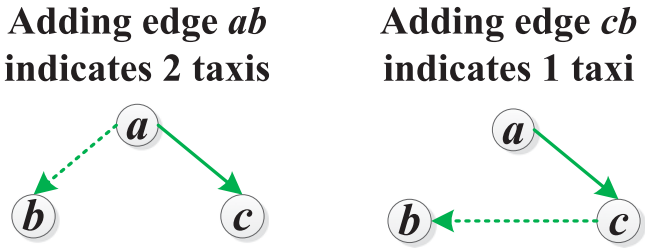

Fig. 22. Constraints on number of available taxicabs n.

is a spanning tree and MST is the smallest spanning tree, $2W(T) \leq 2W(O)$. Thus, $W(S) < W(T') = 2W(T) \leq 2W(O)$; therefore, $\frac{W(S)}{W(O)} < 2$.

During the construction of the minimum spanning tree, three constraints (i.e., Taxicab Capacity, Number of Available Taxicabs, and Travel Period) have special impacts, which will be introduced in the following three subsections.

7.4.2. Impact of Number of Available Taxicabs n. In this section, we show how to solve a carpool problem with constraints on the number of available taxicabs n.

We can reduce the total mileage by delivering passengers separately, if they are heading in significantly different directions. In a delivery graph G, every subtree rooted at the origin is associated with a separate taxicab, which satisfies all delivery requests in this subtree. For example, in Figure 21, the spanning tree has three subtrees (boxed), and therefore we need three taxicabs to satisfy the deliveries.

When constructing a spanning tree, we have to find one spanning tree whose number of subtrees rooted at the origin is not bigger than n. Figure 22 shows how to impose such a constraint during a spanning tree construction.

In Figure 22, given that $n = 1$ (i.e., there is only one taxicab available for origin a), suppose that after adding edge ac, currently the minimum edge that should be added to the tree is edge ab according to Prim's algorithm. But adding edge ab indicates that we need two taxicabs to fulfill the deliveries, which is against to $n = 1$. Alternatively, we can add edge cb and it will still fulfill the deliveries yet with one taxicab.

Note that if the only constraint is the number of taxicabs, there is always a spanning tree (e.g., a "path" graph where one taxicab takes all passengers) under the constraint of the number of taxicabs.

7.4.3. Impact of Taxicab Capacity c. In this section, we consider how to solve a carpool problem with constraints on the taxicab capacity c.

Since the taxicab capacity is limited (e.g., four for a sedan and six for a van), a delivery graph should not have infinite depth for any delivery branch. It is clear that given a fixed spanning tree, the minimum taxicab capacity is equal to the size of its

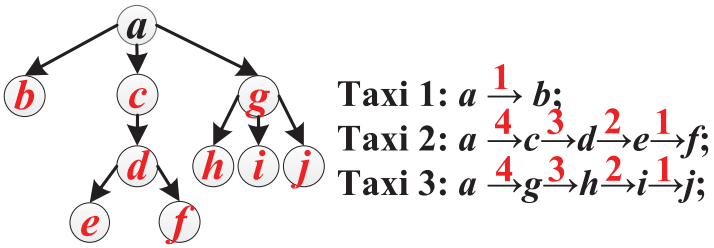

Fig. 23. Minimum taxicab capacity for deliveries.

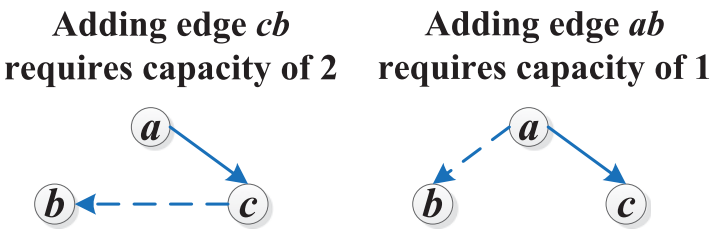

Fig. 24. Constraints on taxicab capacity c.

biggest subtree rooted at the origin, because a taxicab has to deliver all passengers in this subtree from the origin. Figure 23 gives an example, where the biggest subtree has four vertexes, and therefore the minimum taxicab capacity is four.

When constructing a spanning tree, we have to control the sizes of subtrees to make sure the size of the largest subtree is less than given capacity constraint c. Figure 24 shows how to consider it during the construction of a spanning tree.

In Figure 24, suppose $c = 1$ for simplicity, and suppose that after adding edge ac, currently the minimum edge that should be added to the tree is edge cb according to Prim's algorithm. But adding edge cb indicates that we need taxicabs with a capacity of two to fulfill the deliveries, which is against $c = 1$. Alternatively, we can add edge ab and it will still fulfill the deliveries yet with a capacity of one.

Note that if the only constraint is capacity, there is always a spanning tree (e.g., a star graph where every taxicab only takes one passenger) under the taxicab capacity constraint.

7.4.4. Impact of Travel Period. In this subsection, we analyze the carpool problem with constraints on the travel periods of deliveries.

A travel period of a delivery i is specified by $[t_i^s, t_i^e]$, where t_i^s is the earliest time that a delivery i can start, and t_i^e is the latest time that delivery i must finish. The reason to consider travel periods is that in practice, two deliveries with nonoverlapping periods cannot be carpooled together, even though they have the same origin and destination. Thus, a carpool schedule is valid only if its minimum spanning tree fits the travel periods of all deliveries on this tree, which needs to be validated when constructing the minimum spanning tree.

We first provide some rationale behind the validations. Due to a carpool, a taxicab has to leave the origin vertex after the *carpool start time*, which is given by *the start time of the last passenger* in this carpool. Given this carpool start time, the validation is based on the expected travel times of all deliveries. According to a schedule based on the minimum spanning tree, if the carpool start time plus a travel time of a delivery i is smaller than or equal to delivery i's end time, then this spanning tree can accommodate delivery i.

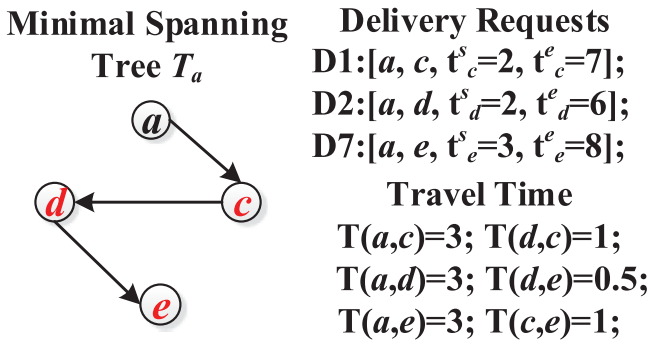

Fig. 25. Validation on travel period.

To impose this constraint, for a minimum spanning tree T_a and a delivery i with travel period $[t_i^s, t_i^e]$ from origin vertex a to destination vertex i, we need to ensure that a spanning tree T_a can accommodate delivery i by satisfying

$$\max_{k \in p} t_k^s + T(a, i) \leq t_i^e,$$

where p is a path on T_a from a to i, and hence $\max_{k \in p} t_k^s$ is the start time of the last passenger, and $T(a, i)$ is the travel time from a to i in p of T_a. The left-hand side is the expected arrival time of delivery i by this carpool, and the right-hand side is the latest end time of delivery i, given by the passenger. Thus, if the left-hand side is not bigger than the right-hand side, it indicates that T_a can accommodate i. Figure 25 gives an example of how to validate whether the MST can accommodate a delivery or not.

In Figure 25, suppose that during the construction of a spanning tree, the next minimum edge that should be added to the spanning tree according to Prim's algorithm is an edge de. Based on delivery requests and travel time in Figure 25, $\max_{k \in p} t_k^s = \max\{2, 2, 3\} = 3$; $T(a, e) = 3 + 1 + 0.5 = 4.5$; thus, $\max_{k \in p = \{a \to c \to d \to e\}} t_k^s + T(a, e) = 7.5 \leq t_e^e = 8$. Therefore, the edge de is a safe edge and can be added to the spanning tree.

Note that if the only constraint is the travel period, there is always a spanning tree (a star graph where every taxicab only takes one passenger) under this constraint.

7.4.5. Other Possible Constraints. The other possible constraints in performing a practical carpool can be easily incorporated in the MST. For example, we consider the trunk space, which is an important factor for a certain travel (e.g., to and from airport). We assume that a taxi has a trunk space, denoted by W. A delivery request i now includes passengers' luggage size w_i. Without loss of generality, we can assume that $w_i \leq W$. If a request has luggage bigger than the trunk space, it is not satisfied by any car and can be safely dropped from our consideration. A spanning tree T_a can accommodate delivery request i if

$$\sum_{k \in p} w_k + w_i \leq W,$$

where p is a path on T_a. This equation implies that T_a can accept i without violating the trunk space constraint. In Figure 24, suppose $W = 5$ and request c and request b have luggage size of four. After adding edge ac, the minimum edge according to Prim's algorithm is cb. Then adding cb breaks the trunk space constraint. Rather, we can add edge ab to fulfill the delivery without violating the constraint.

7.4.6. Put All Constraints Together. A practical approximation algorithm shall construct a minimum spanning tree that (1) accommodates all travel periods of its deliveries,

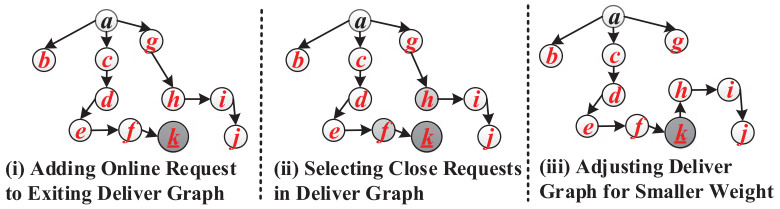

Fig. 26. Online algorithm.

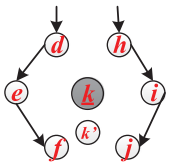

Fig. 27. Close request.

(2) has the biggest size of subtrees not bigger than c, and (3) has a number of subtrees not bigger than n. The details about how to impose the three constraints have been given in the previous section. We note that the order of imposing constraints should not be changed, since it is easiest to find more taxicabs to fulfill requests, relatively easier to find bigger taxicabs to fulfill requests, and harder to require passengers to change their schedules. If the conditions conflict with each other, we can always find a feasible solution by using more taxicabs. Note that with highly diverse travel periods or a small taxicab capacity, lots of taxicabs will be used to satisfy deliveries individually, which is a delivery schedule based on "fat" spanning trees with small yet many subtrees. In contrast, with a small number of available taxicabs, lots of deliveries will be pooled into one taxicab and then be fulfilled one by one, which is a delivery schedule based on "thin" spanning trees with big yet fewer subtrees.

7.5. Algorithm for Online Requests

Instead of providing requests a day or two earlier, some passengers may provide online requests an hour, or even several minutes, before the delivery start time. In *coRide*, we respond to online requests by adding them to an existing carpool schedule. Given an online request k and a delivery graph of existing carpool schedules, our online algorithm has three key steps as shown by the example in Figure 26:

- **(i) Adding New Online Request to the Existing Delivery Graph**: Based on the location of request k, we add k to the closest request f already in the delivery graph. This is the optimal solution for adding this online request. This is because if the optimal solution adds k to another request instead of f, we can always add k to f to obtain a smaller delivery graph, which is better than the optimal solution. So k must be added to f in the optimal solution.
- **(ii) Selecting a Set of Close Requests Regarding the New Online Request**: Based on the location of request k, we select a set of requests that are closer to the new online request k than any other requests already in the delivery graph. In this example, we select h and f as the close requests to k.
- **(iii) For Every Close Request, Adjust the Structure of the Delivery Graph for Small Weight**: For every close request, we first find a route from this close request to the new online request. Then, we compare if the longest link between two requests on this route is longer than the link from the new online request to this close request. If so, we add this link to the delivery graph and delete the longest link. For example, in our example, for the close request h, we first find a route from h to k, which is from h to g to a to c to d to e to f to k. We select the longest link in this route, that is, g to h, and then we compare it to the link from k to h. In our case, the distance from g to h is longer than that from k to h, so we add a new edge from k to h and delete the edge from g to h for smaller total weight, as shown in the figure.

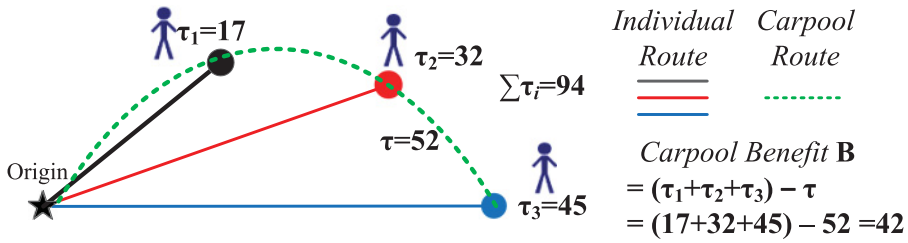

Fig. 28. Carpool benefit.

The most important part of this online algorithm is its time complexity. Assuming we have $|C|$ close requests for an online request in step (ii), and for every close request, we at most go through all existing $|N|$ requests for the adjustment in step (iii). Thus, the time complexity is $O(|C| \times |N|)$. But we can prove that $|C|$ is smaller than seven, so we have a linear time complexity $O(|N|)$. Due to space limitations, we give some intuition in Figure 27. For a new online request k, we at most have six close requests, which form a regular hexagon. If we have one more close request (e.g., k'), it will make one of the existing requests (i.e., f), closer to k' than k. Thus, f is not a close request anymore, which makes the total count of close requests smaller than or equal to six.

8. WIN-WIN FARE MODEL

Generally, a taxicab fare consists of three main parts: an initial charge for every service; surcharge for luggage, waiting time, and so forth; and main charge based on traveled distance. In our model, we focus on how to consider a carpool benefit into calculations of the main charge. Such a carpool benefit shall be shared between the passengers (as a group) and the driver, as well as among the passengers themselves. The rationale behind sharing the carpool benefits with drivers is that we have to encourage drivers to participate in the nonmandatory carpool application. We believe that negotiating privately by passengers alone and sharing the benefits only between passengers will severely hurt the interests of drivers, since the total profit for all drivers will decrease significantly.

8.1. Carpool Benefit

A carpool benefit $\mathbf{B}$ between the total noncarpool fare and a fare paid for a carpool distance is given as follows:

$$\mathbf{B} = \sum_{i=1}^{c} \tau_i - \tau,$$

where c is the total number of passengers in this carpool, τ_i is the separate noncarpool fare for passenger i, and τ is the regular fare for a distance equal to the carpool distance (not the carpool fare). Thus, the total noncarpool fare of all passengers is given by $\sum \tau_i$, and the regular fare for the carpool distance is given by τ, and their difference is a carpool benefit $\mathbf{B}$. Given a carpool schedule, all three parameters are obtainable, and thus $\mathbf{B}$ is also obtainable.

For example, Figure 28 shows three passengers (with noncarpool fare $\tau_1 = 17$, $\tau_2 = 32$, $\tau_3 = 45$) carpooled together with a distance of a regular fare $\tau = 52$, leading to $\mathbf{B} = 42$. Note that $\tau = 52$ is a regular fare for a distance equal to the carpool distance and is not the actual carpool fare all passengers will pay together under our model.

To build a win-win fare model, we need to (1) share a carpool benefit between the driver and all passengers as a group and (2) share the benefit within the passenger group.

8.2. Sharing % Between Driver & Passenger

We use ρ to indicate the sharing percentage of the passengers (all passengers as a group) for a given carpool benefit $\mathbf{B}$, and hence $1-\rho$ is the sharing percentage of the driver.

(i) For a carpool benefit $\mathbf{B}$, all passengers as a group pay the following:

$$\text{Total Fare Paid by Passengers} = \sum_{i=1}^{c} \tau_i - \rho \times \mathbf{B},$$

where $\sum \tau_i$ is the sum of regular fares by all passengers in a noncarpool situation; $\rho \times \mathbf{B}$ is the benefit to passenger group.

(ii) For a carpool benefit $\mathbf{B}$, a driver collects the following:

$$\text{Total Fare collected by Drivers} = \tau + (1-\rho) \times \mathbf{B},$$

where τ is the fare a driver collects for the carpool distance; $(1-\rho)\times\mathbf{B}$ is the benefit for a driver to carpool. Note it is easy to check that the total carpool fare paid by passengers equals the amount collected by the driver in our model.

In real-world scenarios, ρ can be dynamically decided based on various factors about the supply-and-request relationship in a taxicab network. In this article, we give an example to define $\rho = \frac{\text{\# of occupied taxicabs}}{\text{\# of total taxicabs}}$ in a certain area during a time window to balance the carpool incentives between the driver and the passenger. Thus, for a large ρ, that is, more occupied taxicabs, the more benefit will be given to the passengers to encourage passengers to carpool; for a small ρ, that is, more empty taxicabs, the more benefit will be given to the driver to discourage passengers to carpool, balancing deliveries among other empty taxicabs. In Figure 28, given $\rho = \frac{1}{2}$, the total carpool fare collected by drivers is $52 + \frac{1}{2} \times 42 = 73$, which is equal to the total carpool fare for all passengers, that is, $94 - \frac{1}{2} \times 42 = 73$.

8.3. Sharing % Among Passengers

Among the total carpool benefits for all passengers, that is, $\rho \times \mathbf{B}$, we shall decide a sharing percentage to show a carpool benefit for a particular passenger i, and thus model the carpool fare for a passenger i. It is given as follows:

$$\text{Carpool Fare Paid by a Passenger } i = \tau_i - \rho \times \mathbf{B} \times \frac{\tau_i}{\sum \tau_i},$$

where τ_i is the noncarpool fare a passenger i has to pay at a noncarpool situation; $\rho \times \mathbf{B} \times \frac{\tau_i}{\sum \tau_i}$ is the carpool benefit for a particular passenger i. In Figure 28, given $\rho = \frac{1}{2}$, the carpool fare paid by a passenger 3 is $45 - \frac{1}{2} \times 42 \times \frac{45}{94} \approx 34$.

Currently, we use $\frac{\tau_i}{\sum \tau_i}$ to share the carpool benefit among passengers based on their noncarpool fare. In other words, we differentiate passengers by their destinations to the common origin, not the delivery order. But the last dropped-off passenger typically will have a farther destination than other passengers (since our carpool graph is based on the minimum spanning tree), so he or she will share more carpool benefit than other earlier dropped-off passengers in our fare model, which implicitly compensates the passengers with a longer traveling time. In more advanced designs, the sharing percentages among passengers can also be directly decided by the priority of services; for example, based on the delivery order μ_i of passenger i in a carpool, the sharing percentage can be defined as $\frac{\mu_i}{\Sigma \mu_j}$.

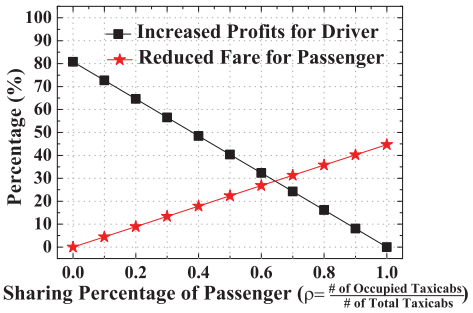

Fig. 29. Incentive balancing.

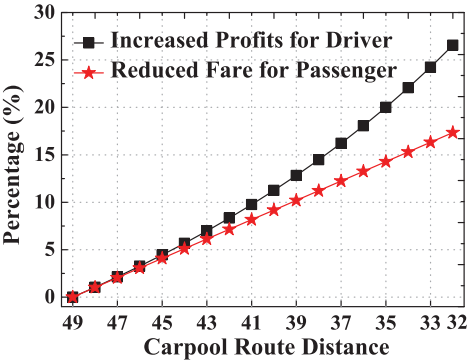

Fig. 30. Win-win model.

8.4. Carpool Fare Model Evaluation

In this subsection, we numerically evaluate our fare model. Based on three delivery requests in Figure 28, Figure 29 shows the impact of different sharing percentages ρ on the fare that every passenger pays and the fare the driver collects. It shows that when ρ increases from 0 to 1 (indicating a trend of undersupplied taxicab services in the real world), the carpool incentive for the passenger increases from 0% fare savings to 44% fare savings, whereas the carpool incentive for the driver decreases from 80% more profit to 0% more profit. By adjusting sharing percentage ρ according to taxicab supply, our model can dynamically balance the carpool incentives for drivers and passengers.

Given requests with fixed noncarpool fares, a short carpool distance will increase the carpool benefit (the same $\sum \tau_i$, but a smaller τ), which results in a win-win situation (i.e., more profits for drivers and lower fares for passengers). Taking the passengers with $\tau_1 = 17$ and $\tau_2 = 32$ in Figure 28 as examples, physically, the lower bound of a fare paid for the carpool distances should be $\tau_{min} = \max\{\tau_1, \tau_2\} = 32$. In addition, passengers may not select a carpool delivery where they pay a fare together more than the sum of their regular noncarpool fares. So, logically, the upper bound for τ is $\tau_{max} = \tau_1 + \tau_2 = 49$. Figure 30 shows impacts of different carpool route distances (by different τ from τ_{max} to τ_{min}) on the percentages of passengers' savings and percentages of drivers' profits. First, it shows a win-win situation as long as $\tau < \tau_{max}$. Second, the smaller τ, the higher the profit for drivers, and the lower the fare for passengers.

9. *CORIDE* IMPLEMENTATION

We have installed the customized TaxiBox in a small portion (98 taxicabs) of the taxicab network of a Chinese city, Shenzhen, with a population of 10 million to test the functionality of TaxiBox. We quickly learned that it takes time to install hardware in current taxicabs, and that it is much more difficult than we had anticipated. Although the taxicab operators requested that their drivers cooperate with the deployment, drivers still were not enthusiastic about installing devices in taxicabs with no immediate benefits to them. During the deployment, it was usual for drivers to not appear or to arrive late and leave early due to business or personal matters. It was also hard to persuade drivers to be more involved in system testing, for example, logging passenger numbers for every delivery. How to provide an incentive for them to be involved in system deployment and testing is a key question we need to address.

For a large-scale carpool deployment, through the operators from which we obtained datasets, the dispatching center to collect delivery requests via phone apps we introduced has been established. But the detailed regulation laws are still under the process

Fig. 31. Experiment in Tanglang Station.

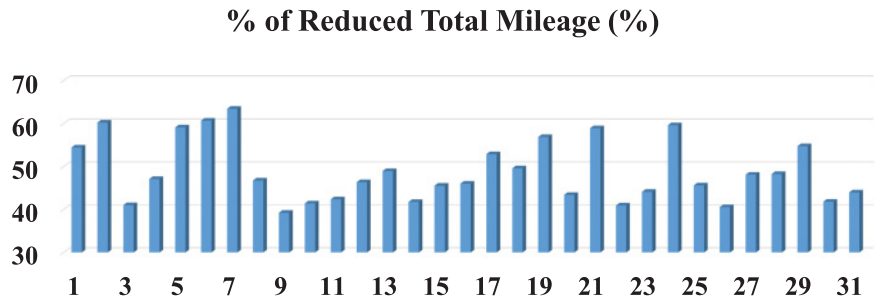

Fig. 32. Reduced mileage.

to being passed, and hopefully will be completed within this year. Thus, a large-scale carpool service evaluation is hard to conduct for the current situation.

In this section, we describe our trial implementation of our *coRide* system. We rent three taxicabs to drive 12 volunteers from a subway station to their workplaces as in Figure 31. Their workplaces are near our research lab in Shenzhen, and within a 1km radius. They arrived at the subway station based on their own schedule, but usually from 8:20 AM to 8:50 AM. All the volunteers went to work during these 31 days. Based on their final destinations, we formulate a request graph, and then obtain a delivery graph for them based on our approximation algorithm under several constraints, for example, a vehicle count of four, a vehicle capacity of four, and a tolerated waiting time of 5 minutes. The taxicabs will go back to the subway station after all volunteers are delivered. We plot the data of the 12 involved volunteers for a 31-day period evaluation in Figure 32. We use a metric called the percentage of reduced total mileage, which is obtained by the total mileage used to deliver all passengers with carpool, and the total mileage used to deliver all passengers without carpool. Due to different combinations of volunteers based on their starting time, the percentage of reduced mileage is different at different days even though their origins and destinations are the same. On average, we reduce mileage by 49% for all passengers.

10. *CORIDE* EVALUATION

In this section, we perform a large-scale trace-driven evaluation of a real-world dataset about GPS records of 14,453 taxicabs belonging to different taxicab companies in Shenzhen. The first dataset contains daily GPS trace data of the taxicab network, and the second dataset is about deliveries. The GPS dataset was collected by letting each taxicab upload its records 30 seconds on average to a centralized base station, and the

Description of Datasets			
GPS Dataset		Delivery Dataset	
Collection Period	01/01/12-06/30/12	Collection Period	01/01/12-06/30/12
Numbe of Taxis	14,453	Numbe of Taxis	14,453
Data Size	450GB	Data Size	18GB
Record Number	3,888,000,000	Record Number	95,000,000
Format		Format	
Plate Mumber	Date and Time	Plate Mumber	Begin & End Time
Status	Speed	Delivery Distance	Delivery Duration
Direction	GPS Coordinates	Delivery Fare	Unload Distance

Fig. 33. Details of datasets.

delivery dataset was obtained by an offline method. Figure 33 gives details about these datasets.

In the GPS dataset, key attributes are taxicab status and GPS coordinates, which can indicate whether a taxicab at a certain location is empty or not. In the delivery dataset, the key attributes are delivery distance, duration, and fare, which can describe a taxicab delivery event. Further, the unload distance indicates the distance between the end location of the last delivery and the begin location of this delivery. By combining these two datasets, we can fully understand the daily operational situation of the entire taxicab network and conduct a valid evaluation. Due to the large size of the datasets, we mainly found two kinds of errors: (1) Location Error: GPS coordinates show that a taxicab is off the road; (2) Missing Records: a fair amount of GPS records are missing. The errors may result from different reasons (e.g., GPS device malfunctions, software issues, etc). We perform a preprocessing to clean datasets to rule out taxicabs with more than 10% of missing or errant records.

10.1. Evaluation Overview

To show the effectiveness of carpool services, we compare two carpool route calculation algorithms, the ***optimal carpool*** and the approximation algorithm, indicated as ***coRide***, with the ***ground truth***, which is the original GPS traces from the dataset. To show the performance of *coRide* to address online requests, we also plot the performance of ***coRide online***.

The previous algorithms are evaluated based on three different real-world constraints. (1) **Taxicab Capacity** $\boldsymbol{c}$ to show how many deliveries can be pooled together in a single taxicab, (2) **Number of Available Taxicabs** $\boldsymbol{n}$ to show how many taxicabs can be used at an origin to fulfill all delivery requests, and (3) **Travel Period** $[\boldsymbol{t_i^s}, \boldsymbol{t_i^e}]$ to show the delivery start time and a tolerated end time. For travel period constraints, since we can obtain the actual travel period about every delivery in the dataset, we use a *tolerated detour time* t (minutes) plus the actual end time of a trip to show this constraint. For example, for an actual travel period $[t_i^s, t_i^e]$ in the dataset about delivery i, with a tolerated detour time t, the travel period we used to test a spanning tree is $[t_i^s, t_i^e + t]$, instead of the actual travel period $[t_i^s, t_i^e]$.

From the three perspectives of society, passengers, and drivers, we evaluate the performance of these algorithms by several metrics. From society's perspective, with the **Percentage of Reduced Total Mileage**, we investigate how much mileage we can reduce by carpooling, given the aforementioned constraints and different time lengths between the time to provide delivery requests and time to start deliveries for online requests. We also investigate the impacts of both the hours of the day and days of the week on the percentage of reduced total mileage. From passengers' perspective, with the **Percentage of Reduced Fare** paid by passengers, we show the minimum fare they can pay, given tolerated detour times. From drivers' perspective, with the

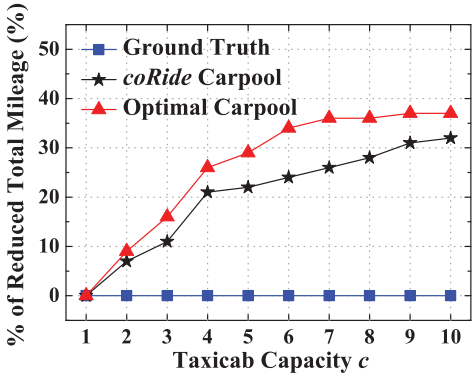

Fig. 34. Total mileage versus c.

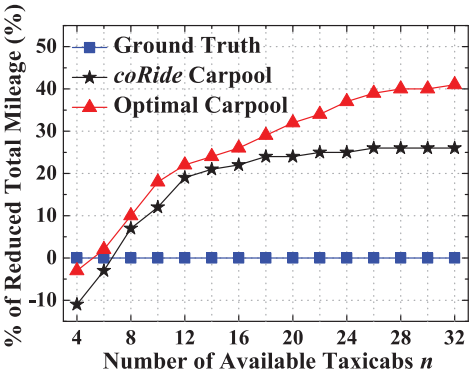

Fig. 35. Total mileage versus n.

Percentage of Increased Profit earned by drivers, we present the maximum fare they can collect, given tolerated detour times. Further, we investigate our operating model by different carpooling locations and carpooling times. In addition, we also investigate two practical metrics: (1) the running time of the optimal algorithm to show why this optimal algorithm is not feasible in terms of running time, and (2) the increased individual mileage due to carpooling to show a possible negative effect of carpooling, that is, increasing the travel time for passengers.

In the evaluation, for datasets about individual days of the week, we first process datasets to obtain delivery requests, and then based on the delivery requests we calculate the carpool route by different algorithms. By processing these requests on a daily basis, we show the performance when passengers provide delivery requests 24 hours earlier than the delivery start time, and for requests starting on one day and ending the day after, we classify them into the day they start. For *coRide* online, we show its performance when passengers provide requests at 0.25, 0.5, 1, and 12 hours earlier than the delivery start time. The results are average outcomes of 7 days of evaluations.

10.2. Reduced Total Mileage

In this subsection, we evaluate *coRide* via the percentage of reduced total mileage at different parameters.

10.2.1. Taxicab Capacity c. Figure 34 plots the effect of taxicab capacity c on the percentage of reduced total mileage with tolerated detour time $t = 5$ and number of available taxicabs $n = 16$. With the increase of taxicab capacity c, the percentage of reduced total mileage for *coRide* carpool and the optimal carpool also increases. For example, in *coRide* carpool, the percentage of reduced total mileage increases from 0% to 22%, when taxicab capacity c increases from one to four. This is because when taxicab capacity c increases, a delivery of a taxicab can be pooled with more other deliveries, and thus it can reduce the total mileage. It implies that a carpool functions more effectively when taxicabs can carry more passengers. We observe that the optimal carpool always outperforms the *coRide* carpool. But during the increase of taxicab capacity c from four to 10, the performance gain increases between the optimal carpool and *coRide* carpool. This indicates the optimal carpool functions better when c is larger.

10.2.2. Number of Available Taxicabs n. Figure 35 plots the effect of the different number of available taxicabs n on the percentage of reduced total mileage with tolerated detour time $t = 5$ and taxicab capacity $c = 4$. We observe that with the increase of number of available taxicabs n, the percentages of reduced total mileage in *coRide* carpool

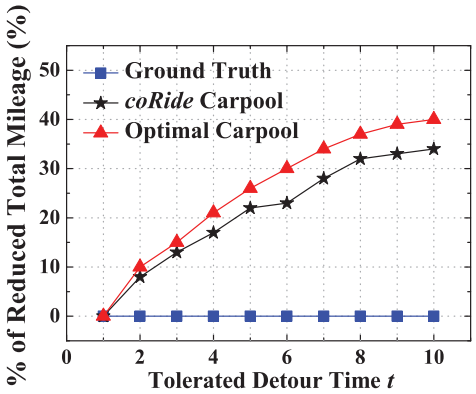

Fig. 36. Total mileage versus t.

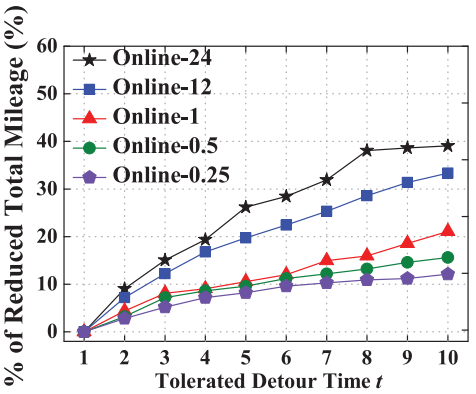

Fig. 37. Mileage (online).

increase from -11% to 27%. These are some negative percentages of reduced total mileage when the number of available taxicabs n is small, and a similar situation is also shown in the performance of the optimal carpool. This is because with fewer taxicabs at an origin, we have to pool more unrelated deliveries in this origin into the same taxicab, and drop them off one by one, and it will increase the total mileage. But when the number of available taxicabs n is larger than eight, we can reduce the total mileage by carpools. We also find that when the number of available taxicabs n is larger than 20, the performance gain between the optimal carpool and *coRide* increased. It may result from the fact that a spanning tree with more subtrees will not necessarily help *coRide* to achieve the global minimum mileage.

10.2.3. Travel Period $[t_i^s, t_i^e]$. Figure 36 plots the effect of different travel periods in terms of different tolerated detour times on the percentage of reduced total mileage with the number of available taxicabs $n = 16$ and taxicab capacity $c = 4$. In Figure 36, we observe that with the increase of tolerated detour time t in terms of minutes, the percentages of reduced total mileage in *coRide* carpool increase from 0% to 33%, while these of the optimal carpool increase from 0% to 40%, leading to a 7% performance gain. In a carpool, with more detour time, more mileage can be reduced by pooling more deliveries together. The increase of t enables a larger travel period, making more deliveries correlated with each other in time. With the increase of tolerated detour time t, the increases of performance gain slow down between the percentages of reduced total mileage of the optimal carpool and *coRide* carpool.

10.2.4. Online Requests. In the aforementioned *coRide* carpool, we process requests by days, so it means we pool the delivery requests that passengers provided 24 hours in advance (named *coRide* online-24). In Figure 37, we evaluate the performance of *coRide* for online request situations where (1) half of the passengers provide requests in advance of 24 hours, and based on them, we build carpool graphs, and (2) the other half of the passengers provide requests in advance of 0.25, 0.5, 1, and 12 hours (indicated as *coRide* online-0.25, *etc*), and we use our online algorithm to optimally add these online requests together to the existing carpool graphs every 0.25, 0.5, 1, or 12 hours, leading to new different carpool graphs. We observe that *coRide* online-24 outperforms all other versions, indicating that the earlier the passengers provide requests, the better the performance. This is because with more requests to begin with, we can build a more effective spanning tree.

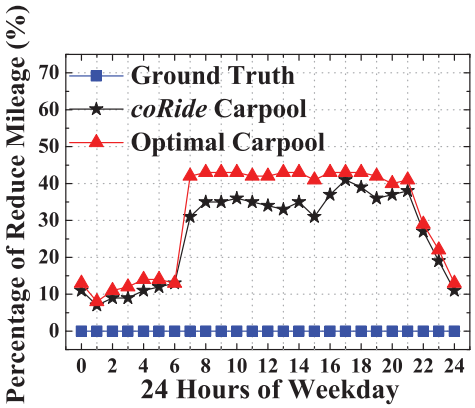

Fig. 38. Weekday mileage.

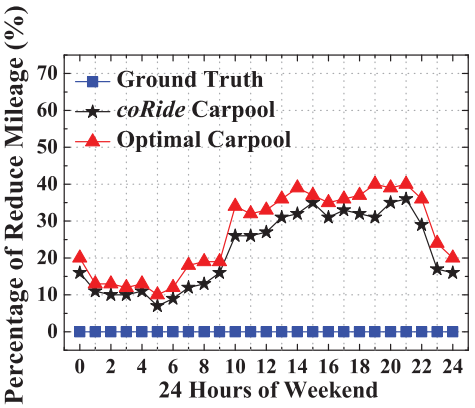

Fig. 39. Weekend mileage.

10.2.5. Hourly Windows on Weekdays and Weekends. We evaluate *coRide* carpool's performance via the percentage of reduced total mileage on weekdays and weekends, respectively. The other constraints are set as $t = 5$, $n = 16$, and $c = 4$. Figure 38 plots the average percentage of reduced total mileage in different 1-hour time windows for 5 weekdays. We observe that in weekday rush hours (e.g., $07:00-10:00$), the percentages of reduced total mileage for two carpool schemes, the optimal carpool and *coRide* carpool, are both higher than 30%. In contrast, in non-rush hours (e.g., $00:01-7:00$), the percentages of reduced total mileage for them are below 20%. Figure 39 shows the average percentage of reduced total mileage in a weekend. We observe that different from weekdays, the high percentages of reduced total mileage on the weekends are between daytime 10:00 and 21:00. There is no significantly high percentage of reduced total mileage in certain time windows among 10:00 to 21:00 than others. But in Figure 38, there are higher performances in the time windows 07:00 to 10:00 and 16:00 to 20:00 than others. It shows that performance of carpool on weekends is different from that on weekdays, since people would take taxicabs at different times on weekdays and weekends.

10.2.6. Running Time of Algorithms. Figure 40 shows the running time of the optimal carpool algorithm and *coRide* carpool algorithm at different carpool passenger numbers p at a single origin. We observe that as the passenger number p increases from two to 18, the running time for the *coRide* carpool algorithm is negligible compared to the running time for the optimal carpool algorithm. This is because our carpool route calculation problem is NP-hard, and the optimal carpool algorithm uses integer programming to obtain the solution, which leads to a longer running time and is not practical for real-world carpool route calculation with a large number of passengers.

10.2.7. Percentage of Increased Individual Mileage. We evaluate the performance of *coRide* carpool by the percentage of increased individual mileage due to carpools with different travel periods. This increased individual mileage also provides an indication of the detour time a passenger will tolerate for carpooling with others. Note that although the individual mileage increases, the fare for individual passengers is actually reduced, since more passengers will share the fare for common routes, leading to a large carpool benefit, as shown by our Fare Model in Section 8. Figure 41 plots the effect of different travel periods in terms of t on the percentage of increased individual mileage with $n = 16$ and $c = 4$. With the increase of t from one to 10, the percentage of increased individual mileage in *coRide* carpool increases from 0% to 30%, while that of the

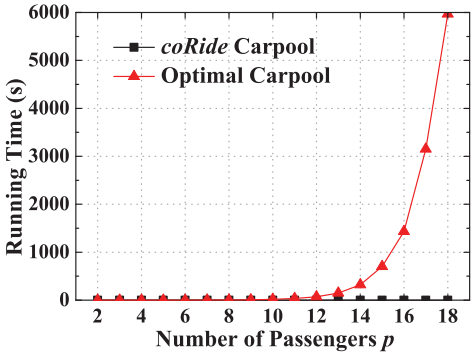

Fig. 40. Time versus p.

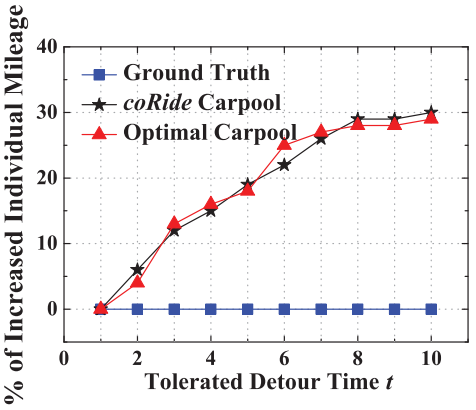

Fig. 41. Mileage versus t.

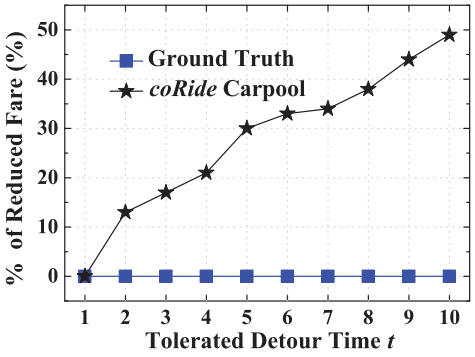

Fig. 42. Reduced fare.

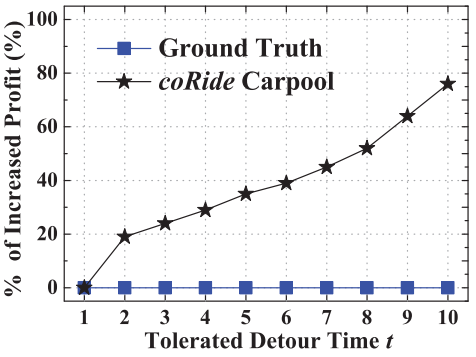

Fig. 43. Increased profit.

optimal carpool has a similar trend. In *coRide* and the optimal carpool, with more detour time, a high mileage is added to individual deliveries, since after carpool, most of the passengers will have a new yet longer route compared to the ground truth.

10.3. Reduced Fare for Passengers

We evaluate the performance of *coRide* carpool in terms of maximally reducing the fare for individual passengers, based on the win-win fare model we proposed in Section 8. Based on the datasets, we have the ground truth for regular fares of individual passengers, and based on the carpool route, we shall have the carpool fare. We let all the passengers and the driver evenly share the carpool benefit due to the mileage reduction of a carpool route. In Figure 42, we observe that with the increase of tolerated detour time t, the percentages of reduced fare for individual passengers in *coRide* carpool increase from 0% to as much as 49%. In a carpool, with more detour time, high mileage can be shared with other passengers, thus leading to a large carpool benefit for fare reductions. It will lead to an economic incentive for passengers to carpool.

10.4. Increased Profit for Drivers

In this subsection, we evaluate the performance of *coRide* carpool in terms of maximally increasing the profit for taxicab drivers based on our win-win fare model. With the method similar to that of the last subsection, we can produce an increased profit by

comparing the total carpool fare collected by the taxicab driver and the ground truth of the regular fare about the first passenger picked up in the carpool, which gives the fare the driver will collect in the case that no carpool is conducted. In Figure 43, we plot the effects of different travel periods in terms of t on the percentage of increased benefits. It shows that with the increase of tolerated detour time t, the percentage of increased benefits for the drivers in *coRide* carpool increases from 0% to as much as 76%, which leads to a considerable incentive for taxicab drivers to take carpool trips.

11. CONCLUSION

In this work, we analyze, design, implement, and evaluate a prototype taxicab carpool system *coRide* to reduce the total mileage to deliver passengers. Our effort provides a few valuable insights and guidelines, which are hoped to be useful for realizing carpooling services commercially in the near future. Specifically, (1) we found unprecedented evidence of inefficiencies of current systems, and opportunities for new systems based on our real-world datasets; (2) we implement a customized hardware supporting the essential functionalities for carpooling; (3) we affirmed that complicated route functions should be implemented in a centralized cloud and near optimality can be achieved; (4) it is important to establish incentives for all the parties involved (e.g., a win-win situation); and (5) finally, our work only addresses the technical frontier, and it is even more critical to establish the right policy that would make a large-scale deployment feasible.

REFERENCES

Ravindra Ahuja, Thomas Magnanti, and James Orlin. 2014. Network flows: Theory, algorithms, and applications. *Pearson New International Edition*.

Ganesh Ananthanarayanan, Maya Haridasan, Iqbal Mohomed, Doug Terry, and Chandramohan A. Thekkath. 2009. StarTrack: A framework for enabling track-based applications. In *Proceedings of the 7th International Conference on Mobile Systems, Applications, and Services (MobiSys'09)*.

Javed Aslam, Sejoon Lim, Xinghao Pan, and Daniela Rus. 2012. City-scale traffic estimation from a roving sensor network. In *Proceedings of 10th ACM Conference on Embedded Network Sensor Systems (SenSys'12)*.

Rajesh Krishna Balan, Khoa Xuan Nguyen, and Lingxiao Jiang. 2011. Real-time trip information service for a large taxi fleet. In *Proceedings of the International Conference on Mobile Systems, Applications, and Services (MobiSys'11)*.

James Biagioni and Jakob Eriksson. 2012. Map inference in the face of noise and disparity. In *Proceedings of the 20th International Conference on Advances in Geographic Information Systems (SIGSPATIAL'12)*.

Data100 Company. 2012. *Taixcab Carpooling Survey*. Data100 Company. http://wenku.baidu.com/view/2f0fea1f964bcf84b9d57b4c.html.

Martin Desrochers, Jacques Desrosiers, and Marius Solomon. 1992. A new optimization algorithm for the vehicle routing problem with time windows. *Operations Research* 40 (1992), 342–354.

Ronald Fagin and John H. Williams. 1983. A fair carpool scheduling algorithm. *IBM Journal of Research and Development* 27 (1983), 133–139.

Yong Ge, Chuanren Liu, Hui Xiong, and Jian Chen. 2011. A taxi business intelligence system. In *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'11)*.

Yong Ge, Hui Xiong, Alexander Tuzhilin, Keli Xiao, Marco Gruteser, and Michael Pazzani. 2010. An energy-efficient mobile recommender system. In *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'10)*.

The Guardian. 2014. Taxi Drivers in European Capitals Strike over Uber. http://www.theguardian.com/politics/2014/jun/11/taxi-drivers-strike-uber-london-live-updates.

Meng Han, Mingyuan Yan, Zhipeng Cai, and Yingshu Li. 2016. An exploration of broader influence maximization in timeliness networks with opportunistic selection. *Journal of Network and Computer Applications* 63 (2016), 39–49.

Meng Han, Mingyuan Yan, Jinbao Li, Shouling Ji, and Yingshu Li. 2014. Neighborhood-based uncertainty generation in social networks. *Journal of Combinatorial Optimization* 28, 3 (2014), 561–576.

Yan Huang and Jason W. Powell. 2012. Detecting regions of disequilibrium in taxi services under uncertainty. In *Proceedings of the 20th International Conference on Advances in Geographic Information Systems (SIGSPATIAL'12)*.

SuperShuttle International. 2015. SuperShuttle service. http://www.supershuttle.com/.

Wei Liu, Yu Zheng, Sanjay Chawla, Jing Yuan, and Xie Xing. 2011. Discovering spatio-temporal causal interactions in traffic data streams. In *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'11)*.

Xuemei Liu, James Biagioni, Jakob Eriksson, Yin Wang, George Forman, and Yanmin Zhu. 2012. Mining large-scale, sparse GPS traces for map inference: Comparison of approaches. In *Proceedings of the International Conference on Knowledge Discovery and Data Mining (KDD'12)*.

Lyft. 2015. *Announcing Lyft Line*. Lyft. https://www.lyft.com/line.

Shuo Ma and Ouri Wolfson. 2013. Analysis and evaluation of the slugging form of ridesharing. In *Proceedings of the 21st ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems (SIGSPATIAL'13)*. ACM, New York, NY, 64–73. DOI: http://dx.doi.org/10.1145/2525314.2525365

Shuo Ma, Yu Zheng, and Ouri Wolfson. 2013. T-share: A large-scale dynamic taxi ridesharing service. In *IEEE International Conference on Data Engineering (ICDE'13)*. IEEE. http://research.microsoft.com/apps/pubs/default.aspx?id=174865. The Best Paper Runner-Up Award.

Johanna Meurer, Martin Stein, David Randall, Markus Rohde, and Volker Wulf. 2014. Social dependency and mobile autonomy: Supporting older adults' mobility with ridesharing ict. In *Proceedings of the 32nd Annual ACM Conference on Human Factors in Computing Systems (CHI'14)*. ACM, New York, NY, 1923–1932. DOI: http://dx.doi.org/10.1145/2556288.2557300

Moni Naor. 2005. On fairness in the carpool problem. *Journal of Algorithms*. 55 (2005), 93–98.

Nationmaster. 2011. Energy statistics. http://www.nationmaster.com/graph/ene_oil_con-energy-oil-consumption.

Paul Oberlin, Sivakumar Rathinam, and Swaroop Darbha. 2009. A transformation for a multiple depot, multiple traveling salesman problem. In *Proceedings of the Conference on American Control Conference (ACC'09)*.

Junhyuk Park and Byung-In Kim. 2010. The school bus routing problem: A review. *European Journal of Operations Research* 202 (2010), 311–319.

Matt Richtel. 2014. Distracted driving and the risks of ride-hailing services like Uber. The New York Times. http://bits.blogs.nytimes.com/2014/12/21/distracted-driving-and-the-risks-of-ride-hailing-services-like-uber/?_r=1.

Schaller Consulting. 2006. The new York city taxicab fact book. http://www.schallerconsult.com/taxi/taxifb.pdf.

NYC Taxi and Limousine Commission. 2011. Taxi of tomorrow survey results. http://www.nyc.gov/html/tlc/downloads/pdf/tot_survey_results_02_10_11.pdf.

"New York Times". 2010. Limited share-a-cab test to begin soon. www.nytimes.com/2010/02/22/nyregion/22ataxis.

Paolo Toth and Daniele Vigo. 2001. The vehicle routing problem. In *Society for Industrial and Applied Mathematics (SIAM'01)*.

Uber 2015. *Announcing UberPool*. Uber. http://newsroom.uber.com/announcing-uberpool/.

Ling-Yin Wei, Yu Zheng, and Wen-Chih Peng. 2012. Constructing popular routes from uncertain trajectories. In *Proceedings of the International Conference on Knowledge Discovery and Data Mining (KDD'12)*.

Wikipedia. 2016. Emissions trading or cap and trade. http://en.wikipedia.org/wiki/Emissionstrading.

Wei Wu, Wee Siong Ng, Shonali Krishnaswamy, and Abhijat Sinha. 2012. To taxi or not to taxi? - enabling personalised and real-time transportation decisions for mobile users. In *Proceedings of the 2012 IEEE 13th International Conference on Mobile Data Management (MDM'12)*.

Jing Yuan, Yu Zheng, and Xing Xie. 2012. Discovering regions of different functions in a city using human mobility and POIs. In *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'12)*.

Jing Yuan, Yu Zheng, Xing Xie, and Guangzhong Sun. 2011a. Driving with knowledge from the physical world. In *Proceedings of the International Conference on Knowledge Discovery and Data Mining (KDD'11)*.

Jing Yuan, Yu Zheng, Chengyang Zhang, Wenlei Xie, Xing Xie, Guangzhong Sun, and Yan Huang. 2010. T-drive: Driving directions based on taxi trajectories. In *Proceedings of the 18th SIGSPATIAL International Conference on Advances in Geographic Information Systems (GIS'10)*.

Jing Yuan, Yu Zheng, Liuhang Zhang, XIng Xie, and Guangzhong Sun. 2011b. Where to find my next passenger. In *Proceedings of the 13th International Conference on Ubiquitous Computing (UbiComp'11)*.

Daqing Zhang, Nan Li, Zhi-Hua Zhou, Chao Chen, Lin Sun, and Shijian Li. 2011. iBAT: Detecting anomalous taxi trajectories from GPS traces. In *13th Conference on Ubiquitous Computing (UbiComp'11)*.

Desheng Zhang, Ye Li, Fan Zhang, Mingming Lu, Yunhuai Liu, and Tian He. 2013. coRide: Carpool service with a win-win fare model for large-scale taxicab networks. In *SenSys*. 9.

Wangsheng Zhang, Shijian Li, and Gang Pan. 2012. Mining the semantics of origin-destination flows using taxi traces. In *Proceedings of the 2012 ACM Conference on Ubiquitous Computing (UbiComp'12)*.

Yu Zheng, Yanchi Liu, Jing Yuan, and Xing Xie. 2011. Urban computing with taxicabs. In *Proceedings of the 13th International Conference on Ubiquitous Computing (UbiComp'11)*.

Received July 2015; revised February 2016; accepted February 2016