# Car4Pac: Last Mile Parcel Delivery Through Intelligent Car Trip Sharing

Fangxin Wang, *Student Member, IEEE*, Yifei Zhu, *Student Member, IEEE*, Feng Wang, *Senior Member, IEEE*, Jiangchuan Liu, *Fellow, IEEE*, Xiaoqiang Ma, *Member, IEEE*, and Xiaoyi Fan, *Member, IEEE*

***Abstract*— The explosion of online shopping brings great challenges to traditional logistics industry, where the massive parcels and tight delivery deadline impose a large cost on the delivery process, in particular the last mile parcel delivery. On the other hand, modern cities never lack transportation resources such as the private car trips. Motivated by these observations, we propose a novel and effective last mile parcel delivery mechanism through car trip sharing, to leverage the available private car trips to incidentally deliver parcels during their original trips. To achieve this, the major challenges lie in how to accurately estimate the parcel delivery trip cost and assign proper tasks to suitable car trips to maximize the overall performance. To this end, we develop *Car4Pac*, an intelligent last mile parcel delivery system to address these challenges. Leveraging the real-world massive car trip trajectories, we first build up a *3D* (time-dependent, driver-dependent and vehicle-dependent) landmark graph that accurately predicts the travel time and fuel consumption of each road segment. Our prediction method considers not only traffic conditions of different times, but also driving skills of different people and fuel efficiencies of different vehicles. We then develop a two-stage solution towards the parcel delivery task assignment, which is optimal for one-to-one assignment and yields high-quality results for many-to-one assignment. Our extensive real-world trace driven evaluations further demonstrate the superiority of our Car4Pac solution.**

***Index Terms*— Intelligent transportation system, trajectory data mining, route planning, travel cost prediction.**

## I. Introduction

RECENT years have witness the rapid development of e-commerce, which also brings unprecedented challenges to the traditional logistics industry. US e-commerce sales are estimated to be 409.2 billion dollars in 2017 and increase to 561.5 billion dollars by 2020 [1], which accordingly generates massive parcels for delivery. The parcels generated in China's 2017 online Singles' Day are estimated to achieve 1.5 billion and have to be delivered in a few days [2]. Such a massive parcel delivery demand obviously surpasses the regular logistical capability, delaying the eventual parcel delivery time for days or even weeks [3]. Considering the entire logistics chain, *last mile* delivery [4] accounts for the most expensive component, ranging from 13 percent to even 75 percent of the entire delivery cost [5]. This is mostly due to the diverse delivery destinations where large-scale parcel consolidation via train or planes no longer applies, leading to expensive human cost. Amazon's annual shipping cost achieved 7.2 billion in 2016 [6], and it proposed AmazonFlex[1] that hires part-time drivers for parcel delivery. Post service providers in some countries (e.g., Canada Post [7]) even plan to stop the home delivery service to reduce this last mile cost. On the other hand, McKinsey, an international famous consult company confirms again that the majority of consumers prefer the to-door delivery and the same day delivery [8].

The key problem lies in the mismatch between the limited logistics capability and the ever increasing parcel delivery demand. Improving logistics capacity to catch the delivery demand no doubt is costly. Furthermore, it is usually the burst of demand in holiday periods that greatly challenges the delivery services, leading to accumulative delivery delay for customers. It is also not cost-efficient to over-provision the delivery capacity to meet such transient demand. We therefore turn to other approaches to dynamically fill in the gap between the limited logistics capability and the increasing delivery demand. As a matter of fact, modern cities are full of transportation resources. We further examine a citywide private car trajectory dataset and observe that everyday car trip is rich enough and covers all the citywide main roads.

Motivated by these observations, in this paper we propose a novel and efficient parcel delivery mechanism through car trip sharing, to address the mismatched logistics capacity and ever-increasing parcel delivery demand. Different from existing car sharing solutions (such as ZipCar[2] and Car2Go[3])

Manuscript received December 1, 2017; revised March 13, 2018, September 19, 2018, and April 5, 2019; accepted September 6, 2019. Date of publication October 7, 2019; date of current version October 2, 2020. This work was supported by the Canada Technology Demonstration Program (TDP) Grant and the Canada NSERC Discovery Grant. The work of X. Ma was supported in part by the National Natural Science Foundation of China under Grant 61702204. The work of X. Fan was supported in part by the NSERC Postdoctoral Fellowship. The Associate Editor for this article was Y. Song. *(Corresponding author: Jiangchuan Liu.)*

F. Wang, Y. Zhu, and J. Liu are with the School of Computing Science, Simon Fraser University, Burnaby, BC V5A 1S6, Canada (e-mail: fangxinw@sfu.ca; yza323@sfu.ca; jcliu@sfu.ca).

F. Wang is with the Department of Computer and Information Science, The University of Mississippi, Oxford, MS 38677 USA (e-mail: fwang@cs.olemiss.edu).

X. Ma is with the School of Electronic Information and Communications, Huazhong University of Science and Technology, Wuhan 430074, China (e-mail: mxqhust@gmail.com).

X. Fan is with the Department of Electrical and Computer Engineering, The University of British Columbia, Vancouver, BC V6T 1Z4, Canada (e-mail: xiaoyif@ece.ubc.ca).

Digital Object Identifier 10.1109/TITS.2019.2944134

[1]https://flex.amazon.com/
[2]http://www.zipcar.com/
[3]https://www.car2go.com/US/en/

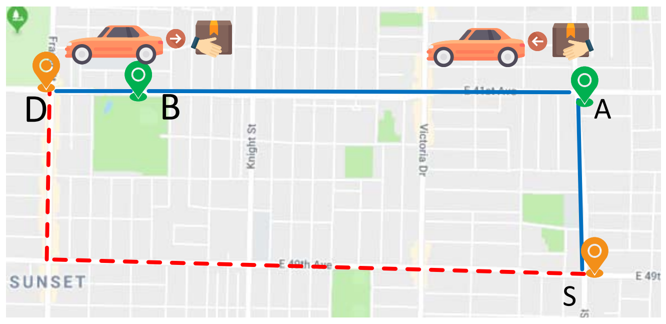


Fig. 1. A toy example of parcel delivery through car trip sharing. A driver is planning to travel from location S to D through the dashed red route. Assume a parcel delivery is scheduled from A to B. Then the driver can deliver this parcel incidentally during the trip through the new solid blue route.

that treat cars as sharing resources, we treat *car trips* as sharing resources, where the parcel delivery can be completed through such a resource sharing. Fig. 1 illustrates an example of parcel delivery through car trip sharing. Assume that a driver is planning to have a trip from the source $S$ to the destination $D$. A parcel delivery is scheduled from $A$ to $B$, where $A$ is very close to $S$ and $B$ is close to $D$. Note that $A$ and $B$ can be any user-designated locations, including home addresses or a storage station. If we can ask the driver to deliver this parcel on the trip with a reasonable amount of reward for compensation, then the parcel delivery can be completed more promptly and economically compared to the traditional logistics service.

In general, we view the delivery of each parcel as a task with a monetary delivery reward upon completion by the given arrival deadline. Given the relatively lower cost on the delivery reward compared to the cost of sending couriers for delivery and the extra income for each driver, parcel delivery through car trip sharing is a win-win solution for all parties.

Although desirable, it is challenging to turn this idea into a practical system. First, since each parcel delivery has an arrival deadline, we need to guarantee that a driver can deliver a parcel before its deadline. Given the complex and time-dependent traffic conditions, it is difficult to accurately estimate when a delivery can be completed with a car trip. Second, once taking a task, the driver may have to pay an extra cost (e.g., extra time cost and fuel consumption) as the parcel delivery task may inevitably change the driver's trip route. A driver is unlikely to take a task with a higher extra cost than reward. Given that multiple car trips may be available to deliver a particular parcel, how to select the best choice remains a problem. Last but not least, such a schedule system requires a fast allocation process, considering the number of parcel delivery tasks and potential car trips can be massive.

To this end, we develop *Car4Pac*, a novel citywide last mile parcel delivery system through intelligent car trip sharing, and closely examine the opportunities and challenges therein. We first construct a 3D[4] (time-dependent, driver-dependent and vehicle-dependent) landmark graph to accurately estimate the travel time cost and fuel consumption cost of each road segment. Mining from more than 25,978 car trips in a citywide road network from our dataset, we learn the travel time and fuel consumption characteristics of each road segment in each time period. Given the different driving skills of drivers (e.g., skilled driver may drive faster and cause less trip time) and different fuel efficiencies of vehicles, we design a mechanism to calibrate their impacts and estimate the costs on each road segment at a particular time using a Gaussian mixture model (GMM) [9]. With the constructed fine-grained 3D landmark graph, we can further calculate the 3D travel cost for each potential car trip.

We then formulate the parcel delivery task assignment as a task-trip matching problem that aims to maximize the social welfare, i.e., finding a solution to assign tasks to proper car trips so that the sum utilities of the platform and all drivers taking delivery tasks are maximized. As the general task assignment problem can be very complex and challenging to solve, we first consider a simpler case, where one-to-one assignment takes place. We then proceed to discuss the general many-to-one assignment situation. Specifically, we formulate the one-to-one task-trip assignment as a maximum weighted matching problem and solve it optimally in polynomial time. For the many-to-one case, we design a heuristic algorithm that conducts assignment iteratively based on the one-to-one case. To our best knowledge, Car4Pac is the first to explore the opportunities of having parcels hitchhiking private car trips and propose comprehensive models and solutions.

We conduct extensive evaluations to verify the effectiveness of Car4Pac based on a citywide real-world massive vehicle trajectory data in Vancouver, Canada. Our dataset collects more than 8,634,000 vehicle driving records from 09 June 2016 to 30 June 2016, each record detailing the GPS information, remaining fuel, car type, driver identification and so on. The result shows that our 3D trip cost prediction can reduce the prediction error by up to 60 percent when compared to the approach without considering different driving skills and fuel efficiencies. In a typical car trip sharing context, Car4Pac can complete more than 90 percent delivery tasks and achieve 40 percent higher social welfare than the state-of-the-art method.

## II. Related Work

In this section, we first introduce some related research fields about the last mile parcel delivery. According to their own focus, we divide these related works into three categories, i.e., car trip sharing, last mile parcel delivery, and route planning and recommendation.

### A. Car Trip Sharing

Car trip sharing or carpooling has attracted a lot of researches in recent years due to its unique advantages in reducing the traffic congestion, improving resource utilization and satisfying the ever increasing travel and delivery demands. The main focus of car trip sharing is trying to find an optimal car trip scheduling to satisfy a proposed target such as minimizing the distance and maximize the utility. For instance, Guidotti *et al.* [10] conducted a real data-driven analysis and constructed the network of potential carpooling to minimize the single occupancy vehicles. Berlingerio *et al.* [11]

[4]In this paper, "3D" always indicates "time-dependent, driver-dependent and vehicle-dependent".

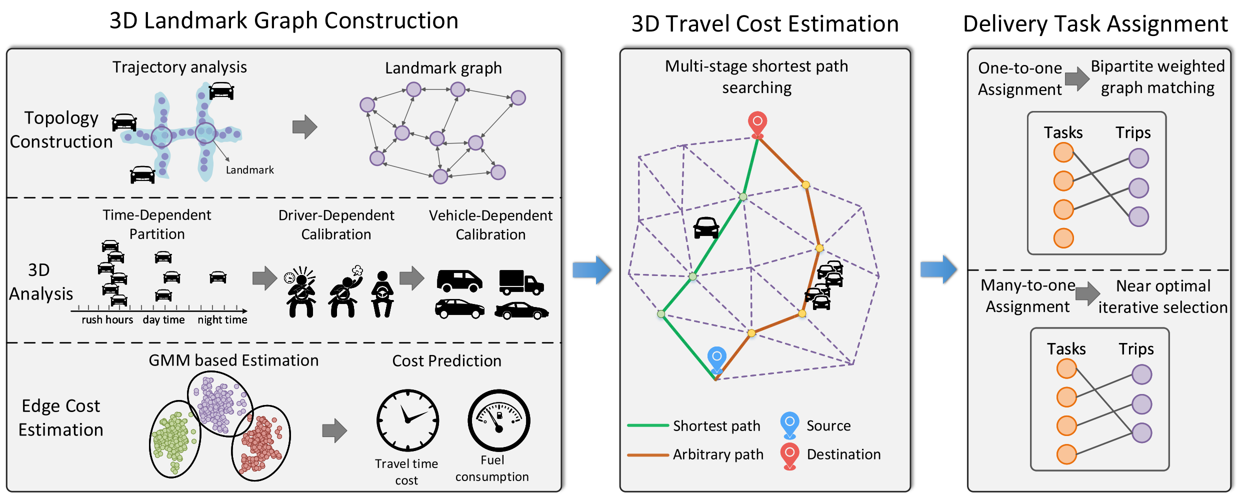


Fig. 2. The framework of Car4Pac.

proposed a carpooling system that not only minimized the number of cars needed at the city level, but also maximized the enjoyability of people sharing a trip. Ferrari *et al.* [12] mainly designed a greedy algorithm for trip matching to reduce the computational complexity, while they only considered the metrics of distance. Ferreira *et al.* [13] proposed a systematical business model for collaborative carpooling. Yet they mainly focused on the module design, credit mechanism and payment strategy rather than the optimal matching. Some related works [14]–[17] also designed model to find optimal carpool routing for urban ridesharing. These works above mostly focused on finding suitable matching between trips and passengers with specific travel plans, which is not dedicated to parcel delivery.

## B. Last Mile Parcel Delivery

Last mile parcel delivery has also become a hot topic in recent years and many works are proposed to improve this service. Punakivi *et al.* [18] analyzed the cost saving between using reception box and delivery box through extensive simulation, and discussed the suitability under different consideration. Boyer *et al.* [19] mainly investigated the impact of customer density and delivery window size on the delivery costs. Some existing works particularly focused on using car trip sharing to improve the last mile delivery. Balcik *et al.* [20] proposed a mixed integer programming model for vehicle schedule between local distribution centers and target locations with practical considerations, while it is not a general point-to-point delivery mechanism and can be inefficient when the scale is large. Wang *et al.* [21] modeled the last mile delivery model as a network min-cost flow problem with some pruning techniques. Yet it still simplified many constraints and mainly considered the delivery distance. Sadilek *et al.* [22] proposed a crowdsourced parcel delivery mechanism that one user delivers the assigned parcel to another user nearby. Each package is passed from person to person according to the overlaps in time and space until it is delivered to the destination. Wang *et al.* [23] proposed a ridesharing based package delivery approach that mainly exploited the online package and trip matching. Chen *et al.* [24] exploited the relays of taxis with passengers to deliver parcels along with the passenger transportation process. As to such hop-by-hop solutions, the platform needs to build many interchange stations to store the parcels when the former delivery has finished yet the latter trip has not begun, which can introduce huge extra cost. In our Car4Pac system, each parcel delivery task only relies one trip, even though one driver during a trip is allowed to take multiple parcels.

## C. Route Planning and Recommendation

The route planning and recommendation through trajectory data mining are the foundations of such trip sharing and play important roles in our Car4Pac system. Many recent works have mined massive trajectory based data [25]–[30] and proposed intelligent route planning model towards different targets. Ding *et al.* [31] considered the dynamic road network with the cost varying from time to time and studied the problem of finding the best departure time to minimize the total travel time from the source to the destination. Winter [32] particularly considered the costs of turns in route planning and reduced this problem as a pseudo-dual graph. Similarly, Szczerba *et al.* [33] also considered multiple mission scenarios such as minimum route length and maximum turning angle. However, these works are not involved with complex real-time traffic conditions.

Baum *et al.* [34] focused on the key problem of electric vehicle route planning, i.e., finding a routing path that minimizes the energy consumption, and developed a practical algorithm achieving fast computation. Besides previous routing plans that focused on a single target (e.g., minimizing the travel time or the fuel consumption), many works also exploited more diversified navigation models. Fan *et al.* [35] claimed that local drivers had a better knowledge of last mile road information and chose the most preferred last mile route as the recommendation to drivers. Some works [28], [29] further exploited more comprehensive routing targets considering distance, fuel cost, time cost, weather, etc., to achieve personalized routing recommendation based on the individual preference of drivers. Different from these aforementioned works, we take a holistic approach towards the last mile parcel delivery based on effective route planning.

## III. Overview of Car4Pac

In this paper, we mainly consider how to fully utilize the car trip resources for parcel delivery so as to achieve maximized social welfare.[5] The Car4Pac system consists of three modules, including *3D Landmark Graph Construction*, *3D Travel Cost Estimation*, and *Delivery Task Assignment*, as illustrated in Fig. 2. Given the collected historical trajectory data, the parcel delivery tasks and the car trip plans, Car4Pac calculates the tasks assignments.

### A. 3D Landmark Graph Construction

Rather than simply using the static road network information from open source maps, we build up a 3D (time-dependent, driver-dependent and vehicle-dependent) landmark graph based on the collected massive car trip trajectory data, where each intersection is represented as a vertex and each road connecting two intersections is viewed as an edge. Different from existing works [24] that only considered time-varying traffic conditions, we comprehensively analyze the impact of different time, different drivers and different vehicles on the cost of each edge. We further estimate the edge cost employing a Gaussian mixture model (GMM) based algorithm and construct a fine-grained landmark graph considering all these factors.

### B. 3D Travel Cost Estimation

When deciding whether to assign a task to a trip, we need to estimate the arrival time of each parcel and the travel cost of each target trip. To do so, we first need to calculate the optimal travel route for a trip. Given that the cost of a trip is affected by different departure time, different drivers' driving skills and different vehicle types, we construct a time-expanded graph to solve this dynamic trip routing problem efficiently and calibrate the cost of each trip considering individual driving skill and vehicle type.

### C. Task Assignment via Car Trip Sharing

We develop a two-stage algorithm towards the parcel delivery task assignment. For the first stage, we consider the one-to-one allocation and convert it into a bipartite weighted matching problem. We can solve it optimally in polynomial time. For the second stage, we generalize the problem to the many-to-one case and design a heuristic algorithm to update the assignment in an iterative manner.

## IV. 3D Landmark Graph Construction

### A. Data Driven Graph Topology Construction

We start from constructing a routable graph from the collected massive historical car trip trajectories to represent the complex citywide road network. We calibrate the GPS locations of each trajectory using Google SnapToRoads API[6] and get uniform GPS samples for each trajectory given that raw GPS records of trajectories are usually skewed and not perfectly mapped to roads. After this processing, we are able to determine whether two trajectories have an intersection. When two trajectories intersect, we identify the point of intersection as a landmark, indicating a vertex in our graph. Note that we use the trajectory intersections rather than the physical road intersections to construct the landmark graph for the following reasons. First, it does not rely on specific road information, making our model adaptive to different cities. Second, it enables the further edge cost calibration (in the following subsections) since each edge have historical trajectory data. Moreover, the graph derived from the trajectories can be more practical than that from physical road information as it can naturally avoid roads that are under maintenance, hard to drive and too remote, etc. Given the massive trajectories cover all the citywide main roads, we can construct such a landmark graph with the following formal definition:

*Definition 1 (Landmark Graph):* A landmark graph is defined as a directed graph $\mathbb{G} = (\mathbb{V}, \mathbb{E})$, where $\mathbb{V}$ and $\mathbb{E}$ are a vertex set and an edge set, respectively. A vertex $v_i \in \mathbb{V}$ is a landmark representing the intersection of at least two roads. A directed edge $e_k = (v_i, v_j) \in \mathbb{E}$ denotes a directed road segment, indicating there is a road from $v_i$ to $v_j$.

With the calibrated GPS records of each trajectory, a car trip can be mapped into the landmark graph and represented by a sequence of consecutive edges,[7] which is defined as:

*Definition 2 (Trip):* A car trip $\mathcal{T}$ describes a trip plan from one place to another, defined as $< \mathcal{T}.s, \mathcal{T}.e, \mathcal{T}.t, \mathcal{T}.dr, \mathcal{T}.ve >$, where $\mathcal{T}.s$ is the trip start location, $\mathcal{T}.e$ is the trip end location, $\mathcal{T}.t$ is the trip start time, $\mathcal{T}.dr$ is the driver identification and $\mathcal{T}.ve$ is the vehicle type of this trip. A trip $\mathcal{T}$ includes a consecutive edge sequence $< e_1, e_2, ..., e_A >$, where $e_j \in \mathbb{E}$.

When assigning a parcel delivery task to a trip, a key consideration is whether this task is worth taking. Thus we need to estimate the cost of a trip, which is defined as follows:

*Definition 3 (Trip Cost):* The cost of a trip is defined by a trip cost function. We consider two main factors, i.e., the time cost and the fuel cost. Given a particular trip with the edge sequence in *definition 2*, we can calculate trip cost as $\mathcal{C}(\mathcal{T}) : (\mathcal{C}_T(\mathcal{T}), \mathcal{C}_F(\mathcal{T})) \to \mathbb{R}$, where $\mathcal{C}_T$ and $\mathcal{C}_F$ represent the travel time cost and fuel consumption cost, respectively.

*Definition 4 (Parcel Delivery Task):* A parcel delivery task $\mathcal{P}$ includes five properties: a pick up location $\mathcal{P}.p$, a drop off location $\mathcal{P}.d$, a task submission time $\mathcal{P}.t$ a required arrival deadline $\mathcal{P}.ddl$, and a reward $\mathcal{P}.rew$ for completing this task. A parcel delivery task should be completed by its deadline.

Since a trip route contains a sequence of edges, to calculate the trip cost, we need to know the travel cost (including the travel time cost and fuel consumption cost) of each edge at

[5]The price design is beyond the scope of this work and is left for future work.

[6]https://developers.google.com/maps/documentation/roads/snap

[7]In reality, a vehicle can start or stop in the middle of a road instead of the intersection. For ease of exposition, we merge the trip source and destination to the nearest landmark, which only introduces marginal errors as demonstrated by our data analysis.

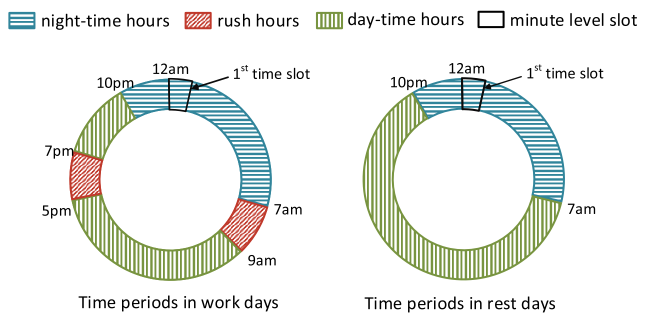


Fig. 3. The distribution of time periods in work days and rest days.

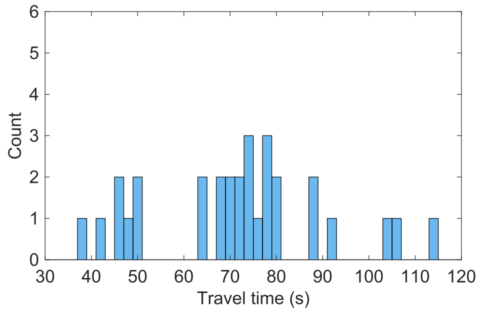
a particular time. Car4Pac obtains the edge costs from the collected trace data and learns the features therein to estimate the future edge cost. We next analyze different factors that affect the cost estimation and introduce the 3D landmark graph construction mechanism.

## B. Time-Dependent Partition

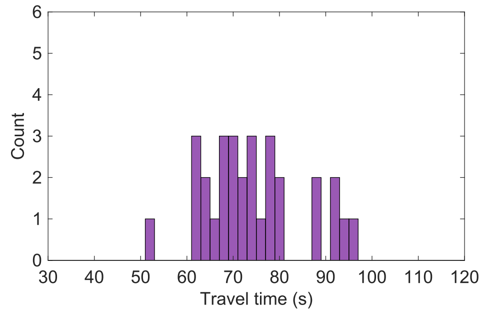
The time and fuel costs of traveling an edge usually highly depend on the traffic conditions, which are closely related to different departure time. For example, roads are usually much more crowded during rush hours and one can spend more time and fuel driving to the same destination than usual.

Previous works [24], [25] usually adopted coarse-grained time slot division, e.g., a day is divided into several equal segments, and predict specific features for each segment. This method is however not accurate enough in this context. Car4Pac adopts a fine-grained two-level partition to accurately capture the time-dependent features for travel time cost and fuel cost on each edge, as illustrated in Fig. 3. The first level is *day level* partition. We divide a week into *work days* $W$ and *rest days* $R$, where we also include holidays and weekends in the rest days. For the work days, we divide a day into three time period: rush hours $W_r$, day-time hours $W_d$ and night-time hours $W_n$; and for the rest days, we can roughly divide a day into two time period: day time $R_d$ and night time $R_n$. Thus, the first level partition has five time period types. The second level is *minute level* partition, dividing a day into $L = [\frac{24*60}{\alpha}]$ time slots, where $\alpha$ indicates the interval minutes between two consecutive slots, e.g., 15 minutes.

Given an edge $e_k$ and a time slot $l$, we select the car trips whose trajectory passes through this edge during this particular time slot as $S^l_{e_k}$. Since traffic flow usually exhibits periodic features, we consider extending the time slot in one day to the corresponding time slots in a *time cycle* (e.g., a week or a month) with the same day level type (i.e., work days or rest days). For example, car trips of time slot $l$ of work days on edge $e_k$ can be calculated as $S^l_{e_k}(Mon) + \ldots + S^l_{e_k}(Fri)$ if the time cycle is a week. In the rest of the paper, all time slots mean the extended time slots unless otherwise specified. Based on $S^l_{e_k}$, we can obtain the cost-count set $C^l_{T,e_k}$ and $C^l_{F,e_k}$ as $\{(cost, count)\}$, where *cost* means the travel time or fuel consumption, and *count* means the number of such trips that are observed.



(a) w/o driver-dependent calibration



(b) w driver-dependent calibration

Fig. 4. The count of different travel time through an edge w and w/o driver-dependent calibration.

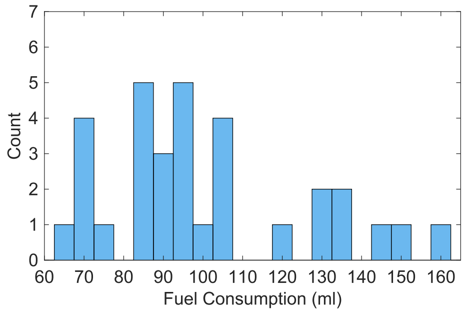
## C. Driver-Dependent Calibration

Besides the time-dependent traffic conditions, people's different driving skills also impose an obvious impact on the cost of travel time for a particular edge. Even departure at the same time, people with different driving skills can go through a road using different time. For example, skilled or senior drivers may prefer to change lanes and overtake other cars frequently so as to drive faster. As a contrast, those unskilled or junior drivers usually tend to drive strictly following the speed limit and only change lane when necessary (e.g., turning left or right in the next intersection), leading to a longer travel time on the same road. Such different behaviors can accumulate to a large variance in $C^l_{T,e_k}$. Fig. 4(a) illustrates the count of different travel time through an edge. We can observe that the travel time distribution is relatively dispersed and the fastest trip uses only one-third time compared to the slowest trip, which is possibly due to the large diversity in driving skills. Thus, the actual benchmark travel cost of each edge is quite driver-dependent and we need to calibrate it for accurate cost estimation.

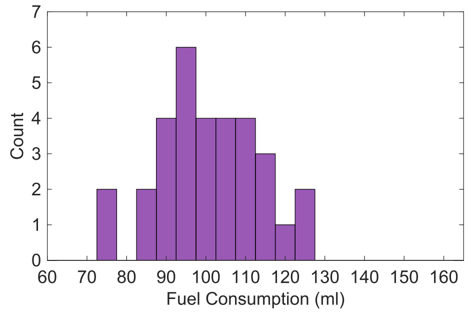
We thus try to identify the drivers for each trip and calibrate the costs based on their individual driving skills. We define the *driving skill index* based on their average trip time, as follows:

*Definition 5 (Driving Skill Index):* Assume that each car in our dataset is used by the same driver and the driver keeps his/her driving skill consistent. Then the *driving skill index* $DS_u$ related to a particular driver $u$ is represented as the mean ratio between his/her travel time through every road and other drivers' average travel time through the same road.

$DS$ is calculated for each driver as the following steps. For a particular driver $u$, all the edges he/she has driven in all time slots are first extracted. Since each car has a unique ID



(a) w/o vehicle-dependent calibration



(b) w vehicle-dependent calibration

Fig. 5. The count of different fuel consumption through an edge w and w/o vehicle-dependent calibration.

in our dataset, we can easily retrieve all the car trips as well as the related edges belonging to a specific driver. Given a particular $C^l_{T,e_k}$, we then represent $DS^l_{u,e_k}$ as the ratio of the travel time cost of driver $u$ and the average travel time cost of all other drivers at edge $e_k$ during time slot $l$. Thus, $DS_u$ is then calculated as the average value of all $DS^l_{u,e_k}$ through our dataset. Repeating this process, we can obtain the driving skill index for every driver. Finally, we calibrate the travel time cost for a road at a given time period by dividing each cost in $C^l_{T,e_k}$ by the corresponding $DS_u$. Fig. 4(b) shows the calibrated benchmark travel time of the same road as in Fig. 4(a). We can find that the cost distribution is more concentrated compared to the original costs, where most people use between 60 seconds to 80 seconds to pass through this road segment.

## *D. Vehicle-Dependent Calibration*

Car trips with different vehicle types usually have different fuel efficiency (represented by fuel consumption of per 100 kilometers in city road), causing diverse fuel consumption collection in $C^l_{F,e_k}$. For example, a truck can consume twice the fuel as that of a small sedan on the same road. Fig. 5(a) shows the count of fuel consumption levels on the same edge. We can find that the fuel consumption distribution is quite dispersed without a centralized region. This is possibly due to the neglect of vehicle type.

To accurately evaluate the benchmark fuel cost of an edge, however, we need a unified indicator. We first choose a particular vehicle type $v_0$ and specify the fuel efficiency of this vehicle type $f_{v_0}$ as the baseline fuel efficiency. Note that the fuel efficiency data can be obtained from the official vehicle manuals given the detailed car types. Similarly, we define a *fuel efficiency index* $FE$ as follows:

*Definition 6 (Fuel Efficiency Index):* Given a baseline vehicle type $v_0$ and the corresponding fuel efficiency $f_{v_0}$. The fuel efficiency index of vehicle type $v$ is defined as $FE_v = \frac{f_v}{f_{v_0}}$, indicating the fuel efficiency ratio of car type $v$ and the baseline car type $v_0$.

We can next calibrate the fuel consumption cost by dividing each cost in $C^l_{F,e_k}$ by the corresponding fuel efficiency index $FE_v$ based on this index. Fig. 5(b) show the count of benchmark fuel consumption after calibration. Compared to raw data, the cost distribution after calibration are obviously more centralized and show better statistical characteristics. These calibrated metrics further enable us to learn the actual characteristics of travel cost and achieve an accurate 3D cost estimation.

## *E. GMM Based Edge Cost Estimation*

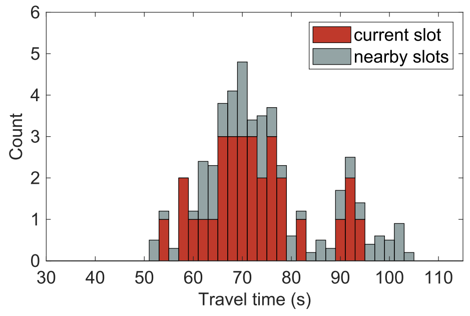
Next, we view the costs of each time slot as random distributions and estimate random variables $RV^l_{T,e_k}$ and $RV^l_{F,e_k}$ for each cost. Intuitively, we can consider $C^l_{T,e_k}$ and $C^l_{F,e_k}$ as the travel time costs and fuel consumption costs of time slot $l$. Yet such an estimation can be inaccurate and not robust for two reasons. First, since traffic changes gradually, this *hard* time slot partition cuts off the continuity of time, which can lead to an inaccurate traffic features capture. Second, given our fine-grained time slot partition, the number of costs in each time slot can be limited, or even empty. For example, it is likely that no car passes through a road in midnight so that we are unable to estimate its random variable.

We therefore utilize a *cost overlay* mechanism to address this issue. When estimating the random variable in time slot $l$, we not only consider the costs in $l$ (i.e., $C^l_{T,e_k}$ and $C^l_{F,e_k}$), but also consider the costs of the time slots nearby. That is, the costs of nearby time slots are multiplied by a weight and then adds to the costs in $l$. Obviously, a farther time slot should have a smaller weight. We employ the widely-used exponential decaying function $Decay(t) = e^{-\frac{t}{\lambda}}$, where $\lambda$ indicates the mean lifetime. The contribution of time slot $x$ regarding the target time slot $l$ is represented as follows:

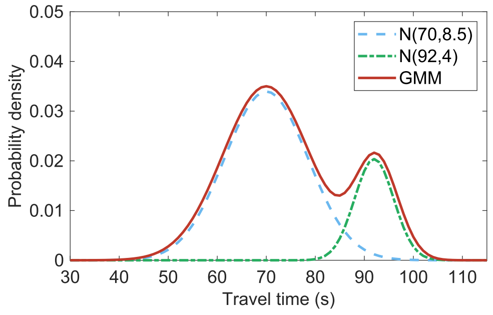
$$Decay(x,l) = e^{-\frac{1}{\lambda(x,l)}\cdot min\{|x-l|,|x+D-l|,|x-D-l|\}} \tag{1}$$

$$\lambda(x,l) = \begin{cases} \tau_{la}, & \text{if } x \text{ and } l \text{ are in the same} \\ & \text{time period type} \\ \tau_{sm}, & \text{otherwise} \end{cases} \tag{2}$$

where $D$ is the number of time slots in a day equal to $[\frac{24*60}{\alpha}]$, $\tau_{la}$ is a larger value and $\tau_{sm}$ is a smaller value. We set $\lambda(x,l)$ as different values according to whether $x$ and $l$ have the same time period type (refer to §IV-B), different from [29] using a constant mean lifetime parameter. This is intuitive to understand: If $x$ and $l$ belong to the same time period type (e.g., the rush hour $W_r$), the cost features of $x$ is more similar to $i$ so that $\lambda(x,l)$ uses a large mean lifetime parameter $\tau_{la}$ to achieve a slow decaying. Otherwise, $\lambda(x,l)$ takes a small value $\tau_{sm}$ for fast decaying if $x$ and $l$ fall in different time periods. In this way, we can calculate the final $C^l_{T,e_k}$ and $C^l_{F,e_k}$ as $\{(cost, \sum_x count(x)\cdot Decay(x,l))\}$. Fig. 6(a) illustrates a simple case of the cost overlay mechanism for a particular



(a) cost overlay



(b) GMM based estimation

Fig. 6. An example of using cost overlay and GMM based estimation.

time slot based on the travel time cost of a road. We can observe that the counts of nearby slots complement that of the current slot, e.g., there is no data record from travel time 55s to 60s, while with the cost overlay mechanism we now have corresponding data records. This mechanism makes the law of data distribution more clear, especially when some roads do not have very dense data samples.

After obtaining the final benchmark time-dependent travel time and fuel consumption cost for each time slots on every road, we next consider learning a random variable and its probability function. Given the possibly diverse data distribution, we need a model that can generally fit all kinds of distributions well. As such, we consider using Gaussian mixture model (GMM) to represent these random variables. As a linear combination of multiple Gaussian distributions, GMM is able to approximate any probability distribution [9], which fits well in our context. Fig. 6(b) shows an example of using two-component GMM to estimate the travel time distribution. We can find that GMM is able to fit the distribution precisely. Finally, we use the expectation of random variables $RV^l_{T,e_k}$ and $RV^l_{F,e_k}$ as the estimated travel cost (including travel time cost $E(RV^l_{T,e_k})$ and fuel consumption cost $E(RV^l_{F,e_k})$) for time slot $l$ on edge $e_k$, and the 3D landmark graph is thereby constructed. The entire 3D landmark graph construction algorithm is presented in Algorithm 1.

It is worth noting that our prediction approach applies well in the real world scenario where the trip data is changing over time (e.g., the travel cost of an edge may suddenly increase due to the road maintenance). Recall that we maintain a *time cycle* (a week or a month) when calculating the costs of each time slots. We update the estimated cost-count set by incorporating the new data of a time slot into the cycle and removing the last one from the cycle. We then recalculate the new edge cost

**Algorithm 1** 3D Landmark Graph Construction

```
Input: Vehicle trajectories from dataset.
Output: The landmark graph $\mathbb{G}$.
1  Build up the vertices and edges based on the historical trajectories;
2  Retrieve all the car trips from these trajectories;
3  Calculate the $DS_u$ for each driver $u$;
4  Calculate the $FE_v$ for each car type $v$;
5  foreach $e_k$ in $\mathbb{E}$ do
6      foreach $l$ in TimeSlots $L$ do
7          Calculate initial $C^l_{T,e_k}$ and $C^l_{F,e_k}$ ;
8          Calibrate $C^l_{T,e_k}$ and $C^l_{F,e_k}$ with $DI_u$ and $FE_v$;
9          Use cost overlay mechanism and $Decay()$ to obtain new $C^l_{T,e_k}$ and $C^l_{F,e_k}$;
10         Estimate random variable $RV^l_{T,e_k}$ and $RV^l_{F,e_k}$ using GMM;
11         Mark $E(RV^l_{T,e_k})$ and $E(RV^l_{F,e_k})$ as the estimated cost;
12 return landmark graph $\mathbb{G}$;
```

through the 3D analysis. In this way, the estimated edge cost of each road can be updated in real time.

## V. 3D Travel Cost Estimation

Given that the travel cost is dependent on different departure time, driving skills and vehicle types, we next consider how to select the optimal routing path for a particular trip and achieve an accurate travel cost estimation. Comparing with traditional static shortest path problem, the travel time and the corresponding cost of each path in our problem are uncertain given the cost of each edge in the landmark graph is time-dependent, driver-dependent and vehicle-dependent. Thus this problem is equivalent to finding the shortest path in a dynamic weighted graph. We consider finding an optimal path to minimize the travel time for a given trip.

According to the travel distance, the problem actually can be divided into two situations. First, a travel trip has quite short distance and the entire time period is covered by only one time slot. Since the landmark graph is static in this situation, we can directly use the traditional shortest path finding algorithm such as Dijkstra algorithm [36] to find the travel path with the least travel time cost. Second, for a long trip that the travel time falls in multiple time slots, the weight of each time slot for each edge is different in the landmark graph. Then the weights of edges change during the path finding in this situation. We assume that all the car trip satisfy the first-in-first-out (FIFO) property,[8] which is exhibited in many networks, in particular transportation networks [37]. Then this dynamic shortest path problem can be solved within multiple stages, where each stage indicates a time slot.

The specific steps are described as follows. We denote the weight of an edge $e_k$ in a particular time slot $l$ as $w(e_k, l)$.

[8]The FIFO property stipulates that vehicles exit from an arc in the same order as they entered, so that delaying one's departure along any path never results in an earlier arrival at an intended destination.

Then an edge has $L$ different weights in total, where $L$ is the total time slot number. If a trip departs at a particular time within time slot $l$, then we begin to search the shortest path using the weights $w(e_k, l)$. When the arrival time at any node spans a time slot during the searching process, we then change the cost of every edge as $w(e_k, l+1)$ and continue searching (note that if we reach the last time slot in a day, we next switch to the first time slot of the next day). We repeat this process until we reach the destination node.

Note that for each particular car trip, we calculate the travel time and fuel consumption considering the driver dependent calibration and vehicle dependent calibration, namely, using a *3D* travel cost estimation. Recall that we have obtained the driving skill index for each driver and the fuel efficiency index for each vehicle during the 3D landmark construction (in §IV). Therefore, when calculating the travel cost of a trip related with a particular driver $u$, we calibrate the edge cost of the landmark graph by multiplying the original cost by $DS_u$ to obtain the driver specific cost. Besides, if a trip is related to a vehicle type $v$, the specific fuel consumption cost is the product of corresponding benchmark value and the fuel efficiency index $FE_v$. For those new drivers without historical records or car trips without specifying vehicle type, we set the two indexes both as 1 so that their estimated cost is the same as the benchmark value. In this way, we can finally obtain the parcel delivery time and the corresponding travel cost for each particular trip.

## VI. Task Assignment via Car Trip Sharing

In this section, we consider how to formulate the problem of parcel delivery through car trip sharing and solve it efficiently. We first introduce some basic definitions and formulate the task assignment problem as an integer programming model. Then we develop a two-stage algorithm to solve it efficiently, namely, achieving the optimal solution for the one-to-one case and solving the many-to-one assignment in an iterative manner.

### A. Problem Formulation

People with potential trips can share their future trip plans with specified source, destination, departure time and so on (recall *Definition 2*). A parcel delivery task can be assigned to a car trip, while two basic requirements have to be satisfied. First, the assigned parcel must be delivered before its arrival deadline. With the 3D travel cost estimation method, we can easily obtain the estimated arrival time for a parcel taken by various trips and only choose those satisfied trips as candidates. Second, since different trips can have different sources and destinations, the cost of taking the same task can also be different. For a trip assigned with a task, the reward for task completion must be higher than the extra cost of taking this task compared with the original trip cost, namely, the utility must be positive.

*Definition 7 (Payment for a Parcel):* Each parcel has a cost that the sender pays to the platform for its delivery, denoted as the *payment* $P_i$ for parcel $\mathcal{P}_i$. For example, the payment can be considered as the cost to deliver this task using the traditional approach.

*Definition 8 (Utilities to the Platform):* Given the original trip as $\mathcal{T}$ and the delivered tasks as $(\mathcal{P}_1, ..., \mathcal{P}_i)$, we denote the new trip taking these tasks as $\widetilde{\mathcal{T}} = (\mathcal{T}, (\mathcal{P}_1, ..., \mathcal{P}_i))$. The utility of this trip to the platform is then defined as $U_p(\widetilde{\mathcal{T}}) = \sum_i P_i - \sum_i \mathcal{P}_i.rew$, which can be viewed as the cost saving brought to the platform if a parcel is sent via car trip sharing instead of being sent by traditional approach.

*Definition 9 (Utilities to Drivers):* The utility to a driver for taking a task is defined as the reward of this task minus the extra cost brought to the driver. Namely, $U_d(\widetilde{\mathcal{T}}) = \sum_i \mathcal{P}_i.rew - [\mathcal{C}(\widetilde{\mathcal{T}}) - \mathcal{C}(\mathcal{T})]$.

With the requirements above, we aim to assign delivery tasks to proper candidate trips to maximize the social welfare.[9] Specifically, given a set of parcel delivery tasks $\mathbb{P} = (\mathcal{P}_1, \mathcal{P}_2, ..., \mathcal{P}_n)$ and a set of trip plans $\mathbb{T} = (\mathcal{T}_1, \mathcal{T}_2, ..., \mathcal{T}_m)$, find the task assignment $< (\mathcal{P}_{i_1}, \mathcal{P}_{i_2}, ..., \mathcal{P}_{i_k}) \rightarrow \mathcal{T}_j >$ with objective of maximizing the sum of $U_p(\widetilde{\mathcal{T}_j})$ and $U_d(\widetilde{\mathcal{T}_j})$ for every $\mathcal{T}_j \in \mathbb{T}$.

We denote $x_{ij} \in \{0, 1\}$ as a variable indicating whether task $\mathcal{P}_i$ is assigned to trip $\mathcal{T}_j$. Let $ArrT(\widetilde{\mathcal{T}_j}, \mathcal{P}_i)$ denote the arrival time of $\mathcal{P}_i$ in trip $\widetilde{\mathcal{T}_j}$. Then the **problem formulation** is as follows:

$$Max : \sum_{j=1}^{m} \sum_{i=1}^{n} x_{ij} \left[ P_i - \left( \mathcal{C}(\widetilde{\mathcal{T}_j}) - \mathcal{C}(\mathcal{T}_j) \right) \right] \tag{3}$$

$$s.t.$$

$$\sum_j x_{i,j} \leq 1, \quad \forall i \tag{4}$$

$$\widetilde{\mathcal{T}_j} = (\mathcal{T}_j, (x_{ij} \mathcal{P}_i)), \quad \forall i \tag{5}$$

$$\mathcal{P}_i.ddl \geq ArrT(\widetilde{\mathcal{T}_j}, \mathcal{P}_i), \quad \forall i \tag{6}$$

where constraint 4 specifies that any task is taken by at most one trip, namely parcel itself is indivisible; constraint 5 explains the representation of each new trip which takes some parcel delivery tasks; and constraint 6 indicates that the parcel arrival time should be earlier than the required deadline.

### B. One-to-One Assignment

We first consider solving the one-to-one task assignment problem, i.e., each trip is allowed to take at most one delivery task. Then the previous formulation should add another constraint as follows:

$$\textstyle\sum_i x_{i,j} \leq 1, \quad \forall j \tag{7}$$

Assume that a delivery task $\mathcal{P}_i$ is assigned to a trip $\mathcal{T}_j$, the trip route then turns from $\mathcal{T}_j.s \rightarrow \mathcal{T}_j.e$ to $\mathcal{T}_j.s \rightarrow \mathcal{P}_i.p \rightarrow \mathcal{P}_i.d \rightarrow \mathcal{T}_j.e$. Given the certain calculated travel routes in the landmark graph, the arrival time of parcel $\mathcal{P}_i$ and the utility of this trip can be determined. We abstract this task assignment problem as a weighted matching problem in bipartite graph, defined as follows:

*Definition 10 (Maximum Weighted Bipartite Matching):* Given a bipartite graph $G = (V, E)$ with a bipartition $(A, B)$ and weight $w > 0$ between each part, find a matching $M$ so

[9]Note that our focus in this paper is to maximize the total utility of social welfare. How to set the appropriate reward to balance the individual utilities to the platform and drivers will be left as a future work.

that the weight of the matching $w(M) = \sum_{a \in A, b \in B} w(a, b)$ is maximized.

In our context, the bipartition $A$ and $B$ represent the task set and trip set, respectively, and $w$ means the sum of utility to the platform and the related driver. If an assignment $(a, b)$ can be completed before the deadline and the sum of utility for an assignment is positive, then there is an edge with weight $w(a, b)$. To solve this matching problem, we need to conduct a series of conversions. Given the number of tasks and trips can be unequal, we first add some dummy nodes to make this graph a balanced graph, i.e., $|A| = |B|$. For any given pair $(a, b)$, there may not exist an edge in between due to the inability to complete the task by the deadline or the negative utility to the driver. For all such pairs, we then add edges with zero weight between them to make it a completed bipartite graph. With these two steps, we convert the graph into a balanced completed bipartite graph. Next, we negate all the weights and make them positive by adding a large constant value. In this way, we convert this problem into a well-known problem of finding the *minimum weighted bipartite perfect matching*. Then we can solve it by Hungarian algorithm [38], [39], with the computational complexity of $O(|V|^3)$.

With the matching result, we reconvert this graph to the original graph. The matching edges with positive weights are the final one-to-one task assignment result, while the zero weight edges are removed since they are not truly assigned. The one-to-one assignment result is optimal given the optimal Hungarian algorithm. To better exploit the potential of Car4Pac, we next study the many-to-one assignment, where a single trip can take multiple parcels.

### C. Many-to-One Assignment

In the many-to-one assignment situation, one single trip is allowed to take multiple parcels. If the costs of taking individual parcels are independent of each other, we can just add duplicate nodes for each trip in the previous bipartite graph, and still run the selected algorithms to find the maximum weighted matching. However, the cost for taking a selected set of parcels is not the same as the sum of the cost for taking each of the individual parcels, which makes this algorithm not applicable to this case. Since each car has limited capacity for taking extra parcels and each driver has limited tolerance level for picking up and dropping off parcels, we assume that each trip $j$ is limited to take at most $N_j$ parcels. This realistic constraint naturally reduces the candidate parcel combinations to examine for each trip. However, even if each trip is allowed to take 2 parcels in total, the combinations of all available parcel-trip is still too large to be processed in a short time. We therefore reexamine this practical problem again and propose our solutions to solve this problem.

We first formally analyze the computational complexity of this problem. Compared to the one-to-one assignment, the difference of many-to-one assignment is allowing allocating multiple tasks to one trip. We no longer treat each individual delivery task as the assignment unit. Instead, we have a bundle $\mathcal{S}$ as a set of tasks where $\mathcal{S} \in \mathbb{P}$. Each bundle $\mathcal{S}_i$ represents one possible task bundle that a trip is capable of taking, namely, $\mathcal{S}_i = (\mathcal{P}_{i_1}, \mathcal{P}_{i_2}, ..., \mathcal{P}_{i_k})$. The previous introduced variable $x_{ij}$ thus indicates whether task bundle $\mathcal{S}_i$ is assigned to trip $\mathcal{T}_j$.

*Theorem 1: The many-to-one social welfare maximization problem is NP-complete.*

We can prove this by transforming the weighted set packing problem into a special case of our social welfare maximization problem. In weighted set packing problem, each set has a weight. We aim at finding a series of pairwise disjoint sets with the maximum weight in the set universe. In our problem, each set corresponds to a bundle of tasks that a trip can take. The weight of this set corresponds to the social welfare contribution of a trip-ask bundle assignment. Consider a simplified case of our problem, if we only select the task bundle with the largest weight for each trip. Then the problem of maximizing social welfare is equivalent to our mentioned weighted set packing problem. Since the weighted set packing problem is known to be NP-complete, where no optimal algorithms exist to solve this problem in polynomial time in all instances, our social welfare maximization problem naturally also is NP-complete even in this simplified case. □

Given the context of car trip sharing, we propose a car trip aware heuristic algorithm based on the one-to-one assignment to address this problem in an iterative way. The algorithm can be divided into three steps. We use the original bipartite graph $G$ in the one-to-one assignment. First, we try to add an edge between each unassigned task $a_i$ and each assigned trip $b_j$ (note that the trip can have been assigned other tasks). If $a_i$ can be completed before its deadline under the assignment $(a_i, b_j)$ and the extra utility gain is positive, we then add an edge with a weight $w(a_i, b_j)$ as the utility gain. Note that the travel routes can have multiple choices when a trip takes multiple tasks. The route selection can be reduced to the traveling salesman problem (TSP) which is NP-hard. Here we simply search for the next nearest feasible stop from the trip departure node and repeat this process iteratively until the destination node. Second, we each time select an available assignment with the largest utility gain and update the new trip routes. If a trip has taken its maximum allowed tasks, then we remove all edges related this trip. We also remove other candidate edges for the assigned task. Third, we recalculate the weights between the newly updated trip and its connected tasks and update these weights accordingly. We repeat the second step and the third step iteratively until no task can be assigned anymore.

Fig. 7 illustrates this process using a simple example. In the 1st round after the one-to-one assignment, we calculate the weight from each task to each available trips and select the pair $(a_2, b_1)$ for assignment as its weight is largest. The in the 2nd round we update all related weights and then select $(a_4, b_2)$ because $w(a_4, b_1)$ is now smaller than $w(a_4, b_2)$ after the 1st round. Then at the 3rd round, the algorithm finishes when all tasks are assigned.

Note that it is possible that some parcels cannot be effectively delivered due to the unreachable source/destination location or tight delivery time demand, where the platform can send its own couriers to handle these tasks or just refuse them. More details on solutions for such situations are beyond the scope of this paper.

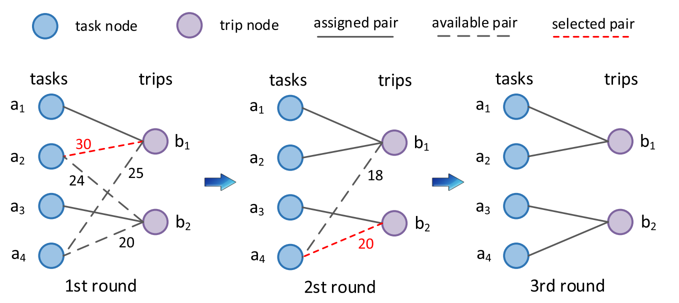


Fig. 7. An example of iterative many-to-one task assignment. In the first round we select $(a_2, b_1)$ and update $(a_4, b_1)$ accordingly. In the second round we select $(a_4, b_2)$. In the last round we achieve the final assignment.

## VII. Evaluation

We conduct extensive real-world trace driven simulations to investigate the performance of Car4Pac. We first evaluate the accuracy of the edge cost estimation from different metrics. Then we compare our task assignment solutions with baseline methods to evaluate its efficiency and effectiveness.

### A. Experimental Setup

We first introduce the data of car trips and parcel delivery tasks we use in our evaluations and the baseline method for comparison as follows:

*1) Car Trip Data:* We use the real-world traces of car trip trajectories for evaluations. We have closely collaborated with Mojio,[10] a leading open platform for connected cars, and collected a three-week trajectory dataset of private cars in Vancouver. This dataset includes more than 8,634,000 data entries and 25,978 car trips, recording the record time, GPS, speed, odometer and fuel levels with dense sampling rate. As such, we use these real car trips in our evaluation. We split the entire dataset into a training set and a testing set. In particular, the data of the first two weeks is used as the training set for the 3D landmark graph construction. The data of the last week is used as the testing set for validating the prediction accuracy of each edge cost in the landmark graph.

*2) Parcel Delivery Tasks:* Since we do not have the real-world parcel delivery data, we empirically generate these data for evaluation. We restrict the deadline for each task in a single day from 9 am to 7 pm. Considering that most parcels are actually delivered in the afternoon and only a small part of them are delivered in the morning, we generate the deadline following Gaussian distribution, i.e., $ddl \sim N(\mu_{ddl}, \sigma^2_{ddl})$, where the mean value $\mu_{ddl}$ is set at 4 pm considering the actual situation. We next randomly generate the source and destination for each task. Those tasks with very short distance (e.g., less than 1 km) are removed out since it is not a big overhead using other delivery methods (e.g., people can fetch their parcels themselves if the distance is very small).

*3) Baseline Methods:* For comparison, we implement two baseline methods for comparison, namely, *Closest Deadline First* (CDF) and *Shortest Distance First* (SDF). For CDF, all the tasks are sorted by their deadlines and all the trips are sorted by their departure times. For every task in the head of the task queue, we examine every trip in the trip queue until finding one trip that is able to deliver the task before its deadline with a positive profit. For SDF, we every time select the task-trip match with the shortest extra driving distance rather than considering the maximum utility.

[10]https://www.moj.io/

We also compare our Car4Pac system with existing state-of-the-art solution to evaluate its effectiveness. Given that most of the related solution cannot well fit our scenario (e.g., some of them rely on trip relay [22], [24] and some do not focus on trip task matching [18], [19]), we consider the algorithm used in [21] (denoted as *MinCost* in the experiment) for comparison. In [21], the authors reduced the task trip matching problem as a minimum cost flow problem. They considered the simple extra delivery distance as the cost while we fully consider the 3D travel cost in our system. Besides, they assumed that all the delivery tasks are independent and ignored the property of accumulative delivery tasks.

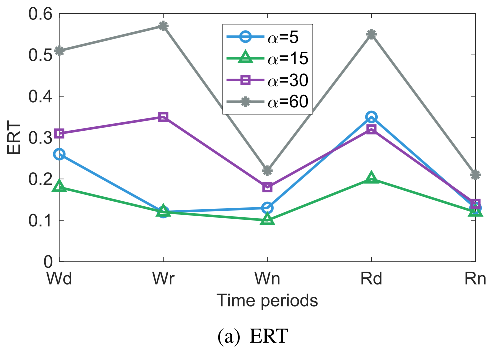
We consider the error ratio of travel time *(ERT)* and that of fuel consumption *(ERF)* as the metrics for edge cost estimation, defined as $ERT = Ave(\frac{|Pred\ time - Real\ time|}{Real\ time})$ and $ERF = Ave(\frac{|Pred\ fuel - Real\ fuel|}{Real\ fuel})$. We also consider the Parcel completion ratio *(PCR)* (defined as $PCR = \frac{|completed\ tasks|}{|total\ tasks|}$), Ratio of completed parcels to trips *(CPTR)* (defined as $CPTR = \frac{|completed\ tasks|}{|total\ trips|}$) and Social Welfare *(SW)* as metrics for task assignment estimation.

### B. Evaluation on Edge Cost Prediction

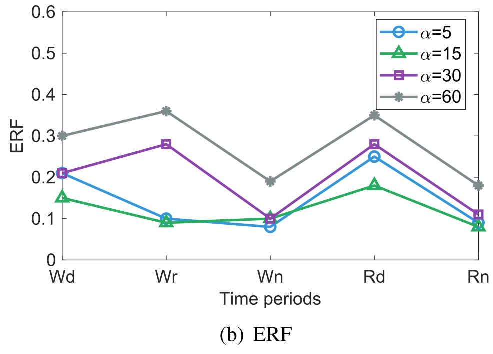
We begin the evaluation from the 3D landmark graph construction in our Car4Pac framework.

*1) Impact of Time Slot Granularities:* We first examine the impact of the time slot granularity $\alpha$ (recall §IV-B) on the cost prediction by setting different time slot intervals, such as 5, 15, 30 and 60 minutes. Fig. 8(a) illustrates the average error ratio of travel time (ERT) of edges in the landmark graph under the different time slot settings. We can find that when $\alpha = 15$, the error ratio of travel time is minimal compared to the real data records. When $\alpha = 5$, the time slot interval is so small that many roads do not have such dense data records. The travel time cost can be less accurate with a small data size. On the other hand, if we use a very large $\alpha$ (e.g., 30 minutes and 60 minutes), the road conditions during a time slot can vary a lot due to the coarse-grained time slot setting, leading to a large variance in travel time prediction. Different time periods also demonstrate different ERT. Compared to other time periods, $W_r$, $W_n$ and $R_n$ have relatively smaller ERT, which is probably due to the highly consistent road conditions (e.g., most roads are crowded in $W_r$ and are pretty clear in night times).

Similarly, we also evaluate the error ratio of the fuel consumption (ERF) with different time slot settings, as illustrated in Fig. 8(b). We can also find that the setting of 15-minute time slot achieves the closest fuel consumption prediction. In rush hours, we can achieve ERF of less than 10 percent with $\alpha = 15$, while the error ratio increases to more than 30 percent if $\alpha = 60$. This is because even in rush hours the traffic congestion level varies according to different times.

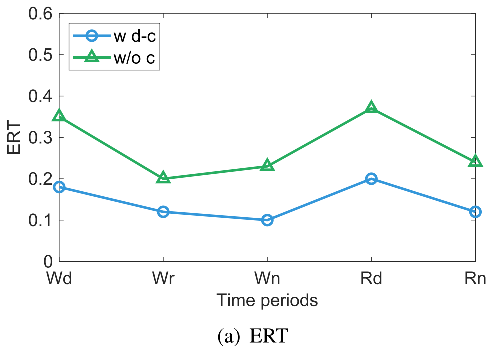


(a) ERT

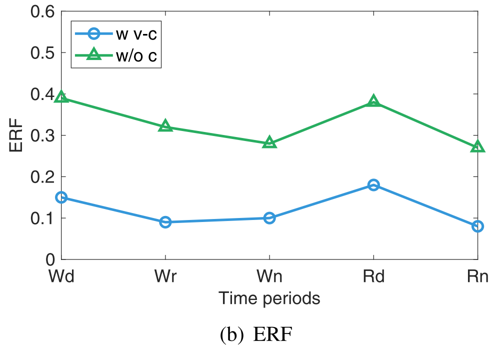


(b) ERF

Fig. 8. Error ratio of travel time and fuel consumption with various time slot settings.



(a) ERT



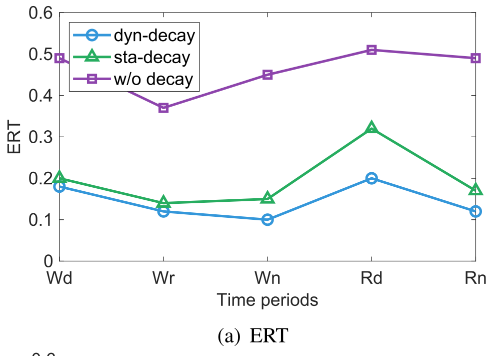
(b) ERF

Fig. 9. Error ratio of travel time and fuel consumption with and without corresponding calibration.

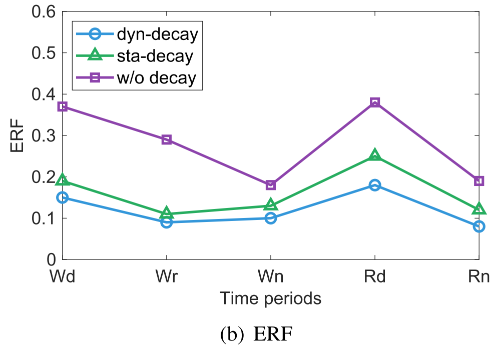
The overly coarse setting, such as 60 minutes, can include many situations with diverse road conditions and make the prediction inaccurate. In this group of evaluation, we find $\alpha = 15$ achieves the best performance among other settings, where our prediction can achieve less than 20 percent error ratio in all time periods for both travel time and fuel consumption.

*2) Impact of Driver and Vehicle Dependent Calibration:* Fig. 9(a) illustrates the comparison between the ERT with driver dependent calibration ($w\ d-c$) and the result without such calibration ($w/o\ c$). We can find that our model achieves much smaller error ratio of travel time in all time periods, where the raw prediction (i.e., without calibration) has more than 35 percent error ratio in $W_d$ and $R_d$. Given that different drivers have different driving skills, this diversity can lead to a large cost variance, even the road conditions are the same. Through the driver dependent calibration, we remove this impact factor when calculating the benchmark travel time cost, and reconsider this factor when predicting the individual trip cost, which enables a driver specific travel time prediction.

The fuel consumption prediction even reveals a bigger difference between ERF with vehicle dependent calibration ($w\ v-c$) and the result without this calibration ($w/o\ c$), as illustrated in Fig 9(b). We can find that without considering the impact of different fuel efficiencies for different vehicle type, the fuel cost prediction can significantly deviate from the actual fuel consumption, achieving an average of 35 percent ERF among all time periods. In contrast, we calibrate the fuel consumption of each car trip considering the baseline fuel efficiency based on their vehicle types and thus is able to achieve a more accurate prediction result (about 12 percent among all time periods).



(a) ERT



(b) ERF

Fig. 10. Error ratio of travel time and fuel consumption with different cost overlay mechanism.

*3) Impact of Cost Overlay Mechanism:* Fig. 10(a) illustrates the ERT of three different mechanisms, namely, using dynamic decaying function ($dyn-decay$), using static decaying function ($sta-decay$) and without using cost overlay mechanism ($w/o\ decay$). Here we set $\tau_{la} = 2$ and $\tau_{sm} = 1$ for the dynamic decaying function and set a constant $\tau = 2$ for the static decaying function. We can find there is a big

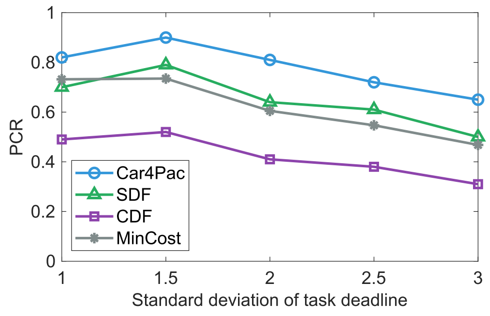


Fig. 11. Task completion ratio of different task deadline generation.

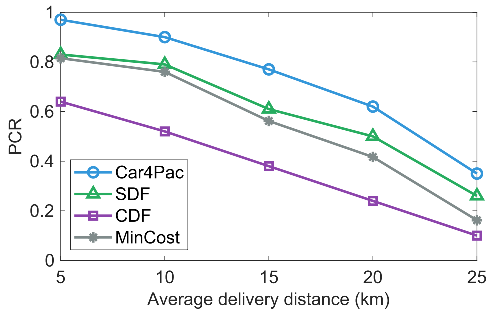


Fig. 12. Task completion ratio of different average parcel delivery distance.

improvement when using at least one cost overlay mechanism. The reason lies in the insufficient trip records when considering all roads in all time slots. Besides, the dynamic decaying function outperforms the static decaying function by an average of 5 percent error ratio in all time periods. Since the traffic condition can transit quickly or slowly depending on different time periods, using a dynamic decaying function can capture the feature more accurately.

Fig. 10(b) shows the similar situation on fuel consumption with different cost overlay mechanism. Our dynamic decaying function achieves the smallest error ratio, which is 10 percent on average. Yet if we do not consider the cost overlay mechanism, we ERF can be as high as 30 percent on average, 20 percent higher than our result. Besides, we observe that the performance difference of these methods in $W_n$ and $R_n$ is relatively smaller than other time periods, which is probably because the road conditions are very clean in these two periods and the fuel consumption does not vary too much.

## C. Evaluation on Task Assignment

We first explain some additional settings before the evaluation on task assignment.

- *Trip extra cost.* We define the extra cost of a trip for taking the parcel delivery tasks as $\beta * extra\ time + \gamma * extra\ fuel$, where $\beta$ is set as the minimum legal hour rate (e.g., \$12/hour) and $\gamma$ is set as the typical fuel price (e.g., \$1.4/L).
- *Average parcel delivery distance ($AveDis$).* The delivery distance of a parcel is the distance between the pick up location and drop off location, denoted as $dis(\mathcal{P}.p, \mathcal{P}.d)$. If we want to generate parcel sets with a larger $AveDis$, we then remove out the tasks with very small delivery distance and regenerate the same amount of tasks repeatedly until $AveDis$ is achieved, and vice versa.
- *Ratio of parcels to trips (PTR).* PTR indicates the ratio of the total task numbers to the total trips numbers, denoted as $|total\ tasks|/|total\ trips|$.
- *Payment and reward for a parcel.* For simplicity, we set the payment for a citywide parcel delivery service as a constant value $P$ (e.g., \$16 as a typical value for same day delivery). We assume the citywide maximum delivery distance is $D$. Then we set the reward for each parcel as $\mathcal{P}.rew = P_0 + \eta * dis(\mathcal{P}.p, \mathcal{P}.d)$, where $P_0$ is the basic delivery reward, $\eta$ is the coefficient for the delivery distance of the parcel. We assume the maximum reward (namely, the task that $dis(\mathcal{P}.p, \mathcal{P}.d) = D$) is equal to the payment, then $\eta$ can be calculated as $\frac{P-P_0}{D}$ given the certain $P$, $P_0$ and $D$.

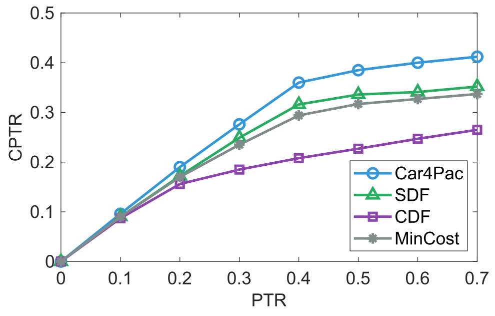


Fig. 13. The value of CPTR when setting different PTR.

The default settings of parameter $\sigma_{ddl}$, $AveDis$, PTR and $\frac{P_0}{P}$ are 1.5, 10km, 0.4 and 0.4, respectively.

We first evaluate the impact of different $\sigma_{ddl}$ on the task completion ratio, as illustrated in Fig. 11. We can observe that when the mean deadline is set at 4 pm and the deviation $\sigma_{ddl}$ is 1.5, the PCR achieves the maximum value for all methods compared with other parameter settings. When the $\sigma_{ddl}$ keeps increasing, the task completion ratio begins to decrease since the deadline of many tasks are so early that they are hard to be completed. Particularly, when $\sigma_{ddl}$ reaches 3, Car4Pac remarkably outperforms other three methods by 15 percent, 34 percent, and 18.2 percent, respectively. Note that the performance result of MinCost is close to that of SDF. This is because these two methods both consider the travel distance as the delivery cost, while Car4Pac fully considers the dynamic cost of travel time and fuel consumption. This comparison result infers that Car4Pac is more capable of processing the task assignment problem even the deadlines of many tasks are very tight.

We further consider the impact of different task delivery distance on the parcel completion ratio, as described in Fig. 12. As the average delivery distance increases, the performance of all the methods gets worse accordingly. This is probably because the excessively far delivery distance may exceed many people's daily car trip distance, where the arrival deadline cannot be satisfied or the extra cost is too large, leading to a low task completion ratio. When the average distance reaches 15 km, the existing state-of-the-art solution MinCost can only achieve a parcel completion ratio of 56.2 percent.

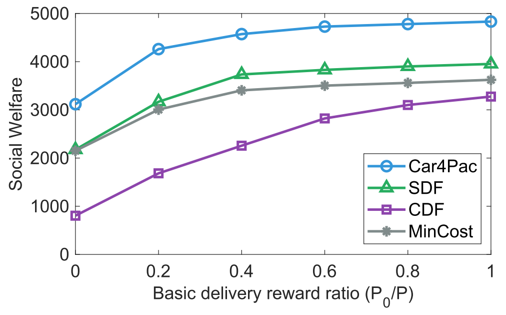


Fig. 14. The social welfare when setting different ratio of basic delivery reward to payment.

Yet our Car4Pac solution has 77 percent PCR, which outperforms MinCost by 20.8 percent, SDF by 16 percent, and CDF by 39 percent.

Besides the previous two factors, the task completion is also affected by the ratio of tasks to car trips. Fig. 13 illustrates the CPTR when we generate tasks according to different PTR values. We can find that when PTR is small (e.g., less than 0.2), most tasks can be completed and the ratio of CPTR to PTR is close to 1. This indicates that when there are abundant car trip resources, all the methods can well handle the delivery tasks. Yet when the PTR increases and reaches 0.4, the CPTR begins to increase very slowly, which infers that the tasks have already exceeded the delivery capacity of car trips. When PTR reaches 0.7, our car4Pac system still has the highest CPTR, which is 8.5 percent higher than MinCost, 7 percent higher than SDF, and 14.7 percent higher than CDF.

We last examine the social welfare when we set different ratio of the basic delivery reward to the payment. Fig. 14 shows the social welfare under different $\frac{P_0}{P}$. In reality, the higher the $\frac{P_0}{P}$ is, it means that the service platform is more likely to offer a high reward for parcel delivery task completion, even for tasks with short delivery distance. We can find that even for a small setting of $\frac{P_0}{P}$ (e.g., 0.2 as a typical value), Car4Pac can achieve a high social welfare of about \$4300, which is 1.4 times higher than MinCost, 1.3 times higher than SDF and 2.5 times higher than CDF. This result indicates that Car4Pac can assign suitable tasks to available car trips more effectively, leading to higher welfare for both drivers and the delivery service platform.

## VIII. Conclusion and Future Work

In this paper, we proposed Car4Pac, a novel and effective last mile parcel delivery system through car trip sharing. Car4Pac leverages the available private car trips to incidentally deliver parcels during their original trips by offering a proper reward to the drivers. To achieve this, we first constructed a 3D (time-dependent, driver-dependent and vehicle-dependent) landmark graph and predicted the travel cost of each road segment accurately. We developed a two-stage solution for the task-trip assignment problem to maximize the social welfare, which can achieve optimal for one-to-one assignment and yield high-quality results for many-to-one assignment. With extensive real-world trace driven experiments, we show that by Car4Pac, the trip cost prediction error can be reduced by up to 60 percent when compared with the approach without driver and vehicle calibration. Car4Pac can effectively complete more than 22.4 percent tasks and achieve 60 percent higher social welfare than the state-of-the-art method in a typical car trip sharing context.

Our system can benefit the real trip sharing system from two aspects. First, the 3D travel cost estimation mechanism helps improve the prediction accuracy of arrival time and travel cost in real trip sharing system such as UberPOOL,[11] where an accurate estimation (in minute level) is necessary. Second, the trip sharing based parcel delivery system can also benefit the post service providers by reducing the overall operational cost without sacrificing the quality of the delivery service (e.g., to better complement Amazon Flex as a more flexible alternative pay-by-trip method).

Our system can be further extended in several aspects, which are worth exploring as our future works. First, the weather (e.g., rainy, snowy and sunny) is likely to have an impact on the trip cost estimation, which is not considered in this paper due to the lack of fine-grained weather information at this time moment. Indeed, our Car4Pac framework can be easily extended from the 3D (time-dependent, driver-dependent and vehicle-dependent) travel cost estimation to 4D, including the weather-dependent travel cost estimation. We are planning to seek help from the local meteorological department, or get the fine-grained weather information of every road through a crowdsourcing way. We expect that our Car4Pac framework can achieve even higher accuracy using the 4D travel cost estimation when more data support on fine-grained weather information becomes available.

Second, the wide deployment of our accurate travel time estimation approach requires the support of large amounts of car trip information, otherwise the landmark graph may not be accurate and complete. Fortunately, some vehicle platforms (e.g., Mojio) and navigation platforms (e.g., GoogleMap) can have sufficient data, which can help our system achieve more accurate prediction and better delivery task assignment. Therefore, another interesting future work is to automatically integrate information from such platforms and gradually improve its performance over time.

Third, our model mainly considers the primary aspects in the cost estimation, i.e., time cost regarding the driving skill and fuel cost regarding the vehicle type. On one hand, such corresponding impacts are much more obvious than the crossover impacts, i.e., time cost determined by different car type and fuel cost caused by different driving skills. On the other hand, it is quite difficult to evaluate such crossover impacts since one driver usually only has one primary car and the diversity is limited. we believe such crossover impacts can be further calibrated given more comprehensive data using the similar calibration methods.

## References

[1] *Retail E-Commerce Sales in the United States*. Accessed: Oct. 1, 2019. [Online]. Available: https://www.statista.com/statistics/272391/us-retail-e-commerce-sales-forecast/

[2] *1.5 Billion Packages to be Shipped During China's Singles*. Accessed: Oct. 1, 2019. [Online]. Available: http://fortune.com/2017/11/11/china-singles-day-alibaba-2/

[11] https://www.uber.com/en-CA/ride/uberpool/

[3] *Argos Customers Have Been Left Furious by Long Delays to Deliveries After Black Friday*. Accessed: Oct. 1, 2019. [Online]. Available: http://www.dailymail.co.uk/news/article-3341898/Argos-customers-fury-delays-Black-Friday-deliveries.html

[4] F. Wang, F. Wang, X. Ma, and J. Liu, "Demystifying the crowd intelligence in last mile parcel delivery for smart cities," *IEEE Netw.*, vol. 33, no. 2, pp. 23–29, Mar./Apr. 2019.

[5] T. Aized and J.-S. Srai, "Hierarchical modelling of Last Mile logistic distribution system," *Int. J. Adv. Manuf. Technol.*, vol. 70, nos. 5–8, pp. 1053–1061, 2014.

[6] *Amazon's Shipping Losses Achieve 7 Billion in 2016*. Accessed: Oct. 1, 2019. [Online]. Available: https://www.geekwire.com/2017/true-cost-convenience-amazons-annual-shipping-losses-top-7b-first-time/

[7] *Canada Post is Ending Its Door-to-Door Delivery*. Accessed: Oct. 1, 2019. [Online]. Available: http://nationalpost.com/news/canada/canada-post-to-stop-door-to-door-delivery-convert-to-community-mailbox-delivery

[8] M. Joerss, J. Schröder, F. Neuhaus, C. Klink, and F. Mann, "Parcel delivery—The future of the last mile," McKinsey & Company, New York, NY, USA, Tech. Rep., 2016.

[9] C. M. Bishop, *Pattern Recognition and Machine Learning*. New York, NY, USA: Springer, 2006.

[10] R. Guidotti, M. Nanni, S. Rinzivillo, D. Pedreschi, and F. Giannotti, "Never drive alone: Boosting carpooling with network analysis," *Inf. Syst.*, vol. 64, pp. 237–257, Mar. 2017.

[11] M. Berlingerio, B. Ghaddar, R. Guidotti, A. Pascale, and A. Sassi, "The GRAAL of carpooling: GReen And sociAL optimization from crowd-sourced data," *Transp. Res. C, Emerg. Technol.*, vol. 80, pp. 20–36, Jul. 2017.

[12] E. Ferrari, R. Manzini, A. Pareschi, A. Persona, and A. Regattieri, "The car pooling problem: Heuristic algorithms based on savings functions," *J. Adv. Transp.*, vol. 37, no. 3, pp. 243–272, 2003.

[13] J. Ferreira, P. Trigo, and P. Filipe, "Collaborative car pooling system," *World Acad. Sci., Eng. Technol.*, vol. 54, pp. 721–725, Jun. 2009.

[14] N. Agatz, A. Erera, M. Savelsbergh, and X. Wang, "Optimization for dynamic ride-sharing: A review," *Eur. J. Oper. Res.*, vol. 223, no. 2, pp. 295–303, 2012.

[15] W. He, K. Hwang, and D. Li, "Intelligent carpool routing for urban ridesharing by mining GPS trajectories," *IEEE Trans. Intell. Transp. Syst.*, vol. 15, no. 5, pp. 2286–2296, Oct. 2014.

[16] S. Yan, C. Chen, and Y. Lin, "A model with a heuristic algorithm for solving the long-term many-to-many car pooling problem," *IEEE Trans. Intell. Transp. Syst.*, vol. 12, no. 4, pp. 1362–1373, Apr. 2011.

[17] E. Kamar and E. Horvitz, "Collaboration and shared plans in the open world: Studies of ridesharing," in *Proc. IJCAI*, vol. 9, 2009, p. 187.

[18] M. Punakivi, H. Yrjölä, and J. Holmström, "Solving the last mile issue: Reception box or delivery box?" *Int. J. Phys. Distrib. Logistics Manage.*, vol. 31, no. 6, pp. 427–439, 2001.

[19] K. K. Boyer, A. M. Prud'homme, and W. Chung, "The last mile challenge: Evaluating the effects of customer density and delivery window patterns," *J. Bus. Logistics*, vol. 30, no. 1, pp. 185–201, 2009.

[20] B. Balcik, B. M. Beamon, and K. Smilowitz, "Last mile distribution in Humanitarian relief," *J. Intell. Transp. Syst.*, vol. 12, no. 2, pp. 51–63, 2008.

[21] Y. Wang, D. Zhang, Q. Liu, F. Shen, and L. H. Lee, "Towards enhancing the last-mile delivery: An effective crowd-tasking model with scalable solutions," *Transp. Res. E, Logistics Transp. Rev.*, vol. 93, pp. 279–293, Sep. 2016.

[22] A. Sadilek, J. Krumm, and E. Horvitz, "Crowdphysics: Planned and opportunistic crowdsourcing for physical tasks," *SEA*, vol. 21, pp. 125–620, Jan. 2013.

[23] F. Wang, Y. Zhu, F. Wang, and J. Liu, "Ridesharing as a service: Exploring crowdsourced connected vehicle information for intelligent package delivery," in *Proc. 26th IEEE/ACM Int. Symp. Qual. Service (IWQoS)*, Jun. 2018, pp. 1–10.

[24] C. Chen *et al.*, "Crowddeliver: Planning city-wide package delivery paths leveraging the crowd of taxis," *IEEE Trans. Intell. Transp. Syst.*, vol. 18, no. 6, pp. 1478–1496, Jun. 2017.

[25] J. Yuan *et al.*, "T-drive: Driving directions based on taxi trajectories," in *Proc. 18th SIGSPATIAL GIS*, 2010, pp. 99–108.

[26] J. Yuan, Y. Zheng, X. Xie, and G. Sun, "Driving with knowledge from the physical world," in *Proc. 17th SIGKDD*, 2011, pp. 316–324.

[27] K. Boriboonsomsin, M. J. Barth, W. Zhu, and A. Vu, "Eco-routing navigation system based on multisource historical and real-time traffic information," *IEEE Trans. Intell. Transp. Syst.*, vol. 13, no. 4, pp. 1694–1704, Dec. 2012.

[28] J. Dai, B. Yang, C. Guo, and Z. Ding, "Personalized route recommendation using big trajectory data," in *Proc. 31st ICDE*, Apr. 2015, pp. 543–554.

[29] B. Yang, C. Guo, Y. Ma, and C. S. Jensen, "Toward personalized, context-aware routing," *VLDB J.*, vol. 24, no. 2, pp. 297–318, 2015.

[30] T. Nuortio, J. Kytöjoki, H. Niska, and O. Bräysy, "Improved route planning and scheduling of waste collection and transport," *Expert Syst. Appl.*, vol. 30, no. 2, pp. 223–232, Feb. 2006.

[31] B. Ding, J. X. Yu, and L. Qin, "Finding time-dependent shortest paths over large graphs," in *Proc. 11th EDBT*, 2008, pp. 205–216.

[32] S. Winter, "Modeling costs of turns in route planning," *GeoInformatica*, vol. 6, no. 4, pp. 345–361, 2002.

[33] R. J. Szczerba, P. Galkowski, I. S. Glicktein, and N. Ternullo, "Robust algorithm for real-time route planning," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 36, no. 3, pp. 869–878, Jul. 2000.

[34] M. Baum, J. Dibbelt, T. Pajor, and D. Wagner, "Energy-optimal routes for electric vehicles," in *Proc. 21st ACM SIGSPATIAL GIS*, 2013, pp. 54–63.

[35] X. Fan, J. Liu, Z. Wang, Y. Jiang, and X. S. Liu, "CrowdNavi: Demystifying last mile navigation with crowdsourced driving information," *IEEE Trans Ind. Informat.*, vol. 13, no. 2, pp. 771–781, Apr. 2017.

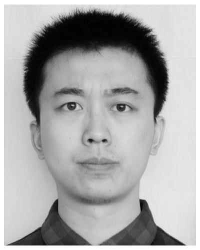
[36] E. W. Dijkstra, "A note on two problems in connexion with graphs," *Numer. Math.*, vol. 1, no. 1, pp. 269–271, Dec. 1959.

[37] B. C. Dean, "Continuous-time dynamics shortest path algorithms," Ph.D. dissertation, Massachusetts Inst. Technol., Cambridge, MA, USA, 1999.

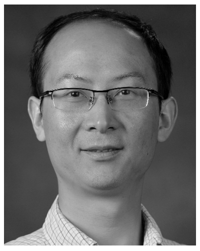
[38] H. W. Kuhn, "The Hungarian method for the assignment problem," *Naval Res. Logistics Quart.*, vol. 2, nos. 1–2, pp. 83–97, Mar. 1955.

[39] J. Munkres, "Algorithms for the assignment and transportation problems," *J. Soc. Ind. Appl. Math.*, vol. 5, no. 1, pp. 32–38, 1957.

**Fangxin Wang** (S'15) received the B.S. and M.S. degrees from the Department of Computer Science and Technology, Beijing University of Posts and Telecommunications, Beijing, China, in 2013 and 2016, respectively. He is currently pursuing the Ph.D. degree with the School of Computing Science, Simon Fraser University, Burnaby, BC, Canada. His research interests include the Internet of Things, wireless networking, multimedia systems, big data analysis, and machine learning.



**Yifei Zhu** (S'15) received the B.E. degree from Xi'an Jiaotong University, Xi'an, China, in 2012, and the M.Phil. degree from The Hong Kong University of Science and Technology, Hong Kong, in 2015. He is currently pursuing the Ph.D. degree with the School of Computing Science, Simon Fraser University, BC, Canada. His areas of interests are cloud computing, multimedia networking, the Internet of Things, and crowdsourcing.



**Feng Wang** (S'07–M'13–SM'18) received the bachelor's and master's degrees in computer science and technology from Tsinghua University, Beijing, China, in 2002 and 2005, respectively, and the Ph.D. degree in computing science from Simon Fraser University, Burnaby, BC, Canada, in 2012. He is currently an Associate Professor with the Department of Computer and Information Science, The University of Mississippi, MS, USA. He is also a Technical Committee Member of the *Computer Communications* (Elsevier). He also serves as a TPC member in various international conferences, such as IEEE INFOCOM, ICPP, IEEE/ACM IWQoS, ACM Multimedia, IEEE ICC, IEEE GLOBECOM, and IEEE ICME. He was a recipient of the IEEE ICME Quality Reviewer Award in 2011. He has served as the Program Vice Chair for the International Conference on the Internet of Vehicles (IOV) in 2014 and the TPC Co-Chair for the IEEE CloudCom 2017 for the Internet of Things and Mobile on Cloud track.

**Jiangchuan Liu** (S'01–M'03–SM'08–F'17) received the B.Eng. degree *(cum laude)* from Tsinghua University, Beijing, China, and the Ph.D. degree from The Hong Kong University of Science and Technology, in 1999 and 2003, respectively, all in computer science.

He is currently a Full Professor (with University Professorship) with the School of Computing Science, Simon Fraser University, BC, Canada. He is also a fellow of the Canadian Academy of Engineering and the NSERC E.W.R. Steacie Memorial Fellow. He is also a Steering Committee Member of the IEEE TRANSACTIONS ON MOBILE COMPUTING. He was a co-recipient of the ACM Multimedia Best Paper Award in 2012, the ACM TOMCCAP Nicolas D. Georganas Best Paper Award in 2013, and the Test of Time Paper Award of the IEEE INFOCOM in 2015. He is also an Associate Editor of the IEEE/ACM TRANSACTIONS ON NETWORKING, the IEEE TRANSACTIONS ON BIG DATA, and the IEEE TRANSACTIONS ON MULTIMEDIA.

**Xiaoqiang Ma** received the B.E. degree from the Huazhong University of Science and Technology, China, in 2010, and the M.Sc. and Ph.D. degrees from Simon Fraser University, Canada, in 2012 and 2015, respectively. He is currently with the faculty of the Huazhong University of Science and Technology as an Assistant Professor. His areas of interests are wireless networks, social networks, and cloud computing.

**Xiaoyi Fan** (S'14–M'18) received the B.E. degree from the Beijing University of Posts and Telecommunications, Beijing, China, in 2013, and the M.Sc. and Ph.D. degrees from Simon Fraser University, BC, Canada, in 2015 and 2018, respectively. He is currently an Honorary Post-Doctoral Research Fellow with The University of British Columbia. His research interests include the Internet of Things, big data, and deep learning.