A City-Wide Crowdsourcing Delivery System with Reinforcement Learning

YI DING, Alibaba Group, China and University of Minnesota, United States
BAOSHEN GUO, Southeast University, China
LIN ZHENG, Central South University, China
MINGMING LU, Central South University, China
DESHENG ZHANG, Rutgers University, United States
SHUAI WANG, Southeast University, China
SANG HYUK SON, DGIST, Korea
TIAN HE, Alibaba Group, China and University of Minnesota, United States

The revolution of online shopping in recent years demands corresponding evolution in delivery services in urban areas. To cater to this trend, delivery by the crowd has become an alternative to the traditional delivery services thanks to the advances in ubiquitous computing. Notably, some studies use public transportation for crowdsourcing delivery, given its low-cost delivery network with millions of passengers as potential couriers. However, multiple practical impact factors are not considered in existing public-transport-based crowdsourcing delivery studies due to a lack of data and limited ubiquitous computing infrastructures in the past. In this work, we design a crowdsourcing delivery system based on public transport, considering the practical factors of time constraints, multi-hop delivery, and profits. To incorporate the impact factors, we build a reinforcement learning model to learn the optimal order dispatching strategies from massive passenger data and package data. The order dispatching problem is formulated as a sequential decision making problem for the packages routing, i.e., select the next station for the package. A delivery time estimation module is designed to accelerate the training process and provide statistical delivery time guarantee. Three months of real-world public transportation data and one month of package delivery data from an on-demand delivery platform in Shenzhen are used in the evaluation. Compared with existing crowdsourcing delivery algorithms and widely used baselines, we achieve a 40% increase in profit rates and a 29% increase in delivery rates. Comparison with other reinforcement learning algorithms shows that we can improve the profit rate and the delivery rate by 9% and 8% by using time estimation in action filtering. *We share the data used in the project to the community for other researchers to validate our results and conduct further research.*[1] [1].

CCS Concepts: • **Human-centered computing → Ubiquitous and mobile computing design and evaluation methods**.

Additional Key Words and Phrases: Crowdsourcing, Sharing Economy, Crowdsourced Labor, Reinforcement Learning

[1] https://tianchi.aliyun.com/dataset/dataDetail?dataId=106807

Authors' addresses: Yi Ding, dingx447@umn.edu, Alibaba Group, Shanghai, China, University of Minnesota, Minneapolis, United States; Baoshen Guo, guobaoshen@seu.edu.cn, Southeast University, Nanjing, China; Lin Zheng, 144611124@csu.edu.cn, Central South University, Changsha, China; Mingming Lu, mingminglu@csu.edu.cn, Central South University, Changsha, China; Desheng Zhang, desheng.zhang@cs.rutgers.edu, Rutgers University, United States; Shuai Wang, shuaiwang@seu.edu.cn, Southeast University, Nanjing, China; Sang Hyuk Son, son@dgist.ac.kr, DGIST, Korea; Tian He, tianhe@umn.edu, Alibaba Group, Shanghai, China, University of Minnesota, Minneapolis, United States.

2474-9567/2021/9-ART97 $15.00
https://doi.org/10.1145/3478117

ACM Reference Format:
Yi Ding, Baoshen Guo, Lin Zheng, Mingming Lu, Desheng Zhang, Shuai Wang, Sang Hyuk Son, and Tian He. 2021. A City-Wide Crowdsourcing Delivery System with Reinforcement Learning. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 5, 3, Article 97 (September 2021), 22 pages. https://doi.org/10.1145/3478117

1 INTRODUCTION

With the advances in e-commerce, we are moving into a new era where online and offline has never been connected so tightly. It is estimated that retail e-commerce sales will increase from 1,336 billion dollars in 2014 to 6,542 billion dollar in 2023 [48]. As the essential component of e-commerce [24, 46, 47], a city-wide delivery service is of special importance and interests in both industry and research communities. With the rise of gig economy [58], companies such as Uber Eats, Postmate [22], Instacart [23], and Ele.me [42]) start hiring the crowd to deliver city-wide packages. The crowdsourcing delivery business model is attracting more and more gig workers because the work flow is simple and the working hours is flexible.

A lot of research works have been proposed in recent years to solve the problem raised in crowdsourcing delivery [6, 10, 36, 54]. Particularly, some studies utilize public transport for delivery given its low-cost delivery network with millions of passengers as potential couriers. In some metropolitan areas, public transport is preferred compared to private cars and taxis because it's faster and brings less congestion and pollution. Researchers have designed several approaches to deliver packages with public transportation [12, 37]. However, most existing works solve the order dispatching problem in an empirical manner, where an optimization problem is formulated and a heuristic algorithm is designed to minimize the cost. There are two limitations of the empirical solutions, (1) some practical factors are not considered like the time constraints, multiple hops, profit, and price setting; (2) given that multiple hops are needed for delivering some orders, independent dispatching at different timestamps fails to consider the relations in subsequent dispatching decisions.

The nature of "hitchhiking" in public-transport-based crowdsourcing delivery enables us to learn the optimal dispatching decision from big data while taking the practical factors and subsequent decisions into consideration. Unlike most existing crowdsourcing delivery that requires detours, public-transport-based crowdsourcing delivery does NOT alter vehicles' or passengers' behavior, which allows us to learn the optimal dispatching decisions from historical data. With massive public transport data and package order data collected from ubiquitous mobile devices and infrastructures, we can build an emulator to mimic the public-transport-based crowdsourcing delivery environment and train a data-driven model. Specifically, we formulate the package dispatching as a package routing problem with sequential station selection for the next hop of the package. Then the problem is naturally formulated as a Markov decision process.

Reinforcement learning (RL) has been used to solve order dispatching in city-wide express [33, 34] and on-demand ride-hailing [25, 51, 60, 66] in recent years. However, these methods cannot resolve the conflicting objectives (earnings for the platform and delivery time for the customers) and time constraints in the *multi-hops* setting of public-transport-based crowdsourcing delivery system. Routing the packages via multiple hops may save time to meet the time constraints but brings additional challenges to optimize the profit and estimate total delivery time.

To address the challenges, we make the following design in applying RL to package routing in this paper: (1) We design a profit model with consideration of earnings (from customer payment), cost (payments to participating passengers), and timeout compensation. The profit model guides the design of the reward function in RL for each action. (2) We design an action filter based on the estimated time of arrival (ETA) to eliminate the invalid actions to improve the package routing performance and provide a statistical delivery time guarantee. Following the ideas, we design ETA-Aware RL-Dispatch with the ultimate goal of profit-maximizing for the platform while providing time-guaranteed delivery service. Specifically, our contributions in this paper are as follows:

• To the best of our knowledge, we conduct the first data-driven study on crowdsourcing delivery based on public transport with practical considerations (i.e., profit, passenger participation willingness, package transfers, and time constraints). We believe the massive transport data and delivery data collected from ubiquitous mobile devices and infrastructures can enable more applications besides crowdsourcing delivery.
• We design a public-transport-based crowdsourcing delivery scheme and formulate the order dispatching problem as an optimization problem to maximize the profit for the platform with guaranteed delivery time. We design ETA-Aware RL-Dispatch, a reinforcement learning algorithm with a carefully designed reward function to dispatch the packages with the ultimate goal of profit maximization. ETA-Aware RL-Dispatch decides the next hop for each package based on the observed and predicted packages' and passengers' states. An ETA module is designed to predict the delivery time between stations with parametric Gaussian. The ETA information is used to filter the actions to speed up training and improve order dispatching and provide statistical guarantee on the delivery time.
• We evaluate the system and the dispatching algorithms with 3-month public transportation data from Shenzhen including 135 million subway swipes, and one month of package delivery data from [anonymous platform] including 10 million packages. Based on the large-scale data, we build an emulator for training the RL model, and test the performance of the model under different settings. We also test the practicability of our system with real world package and passenger data.
• Our research efforts lead to several insights and lessons learned, including (i) Compared with state-of-the-art crowdsourcing delivery baselines, we improve the profit rate and successful deliver rate by 40% and 29%, respectively, indicating the strength of RL in learning optimal routes in multi-hop setting with time constraints; (ii) Compared with RL baselines without ETA information, we improve the profit rate and deliver rate by 9% and 8%, respectively, indicating the advantage of statistical delivery time guarantee provided by Gaussian-based ETA; (iii) Based on a practical passenger participation setting (1%-10% from in-field survey), the public-transport-based crowdsourcing delivery system shows preferable performance; (iv) ETA-Aware RL-Dispatch outperforms baselines at various participate rates, station distances, and time constraints.

2 HITCHHIKING DELIVERY SYSTEM

Although some literature have studied one or two of these topics, (e.g., multiple hops for one package is considered in [12]), we argue that all factors (i.e., time constraints, multiple hops, profit, and price setting) should be considered comprehensively to improve the applicability and the performance of the crowdsourcing delivery via public transport. In the following, we first introduce the practical factors, then design a public-transport-based crowdsourcing delivery system in the practical setting.

2.1 Practical Factors in Crowdsourcing Delivery

Time Constraints. Nowadays, most companies provide time guaranteed delivery services at different levels. For example in Amazon, four types of shipping services are provided [2]. On Amazon Prime Now, an extra of $7.99 is needed if customers want the commodity to be delivered within one-hour [39]. Some delivery companies in China provide "late delivery insurance". Customers who bought the insurance will be compensated if the package is not delivered within the time constraint. Given the facts, we argue that the time constraints should be considered in the package routing process.
Multiple Hops. With pervasive potential transfer stations in public transport, multi-hop delivery becomes possible. Here a ***hop*** means a trip of a single passenger. A counter-intuitive example is that a multi-hop route may save time compared with a single-hop route. Based on Shenzhen subway data on a single day (2018/11/01), for any origin-destination station pair (OD pair), we find the minimum travel time of one-hop routes $T_{\text{min}}^{\text{one-hop}}$ and two-hop routes $T_{\text{min}}^{\text{two-hop}}$ between them. In one-hop routes, the trip is completed by a single passenger from O to D. In

two-hop routes, the route has two trips completed by two passengers, one from O to T, and the other from T to D, where T is a transfer station. We find that for a total of 12,687 OD pairs between 118 stations, $T_{\text{min}}^{\text{one-hop}} < T_{\text{min}}^{\text{two-hop}}$ in 7,641 OD pairs (60%), and $T_{\text{min}}^{\text{two-hop}} < T_{\text{min}}^{\text{one-hop}}$ in 5,046 OD pairs (40%). Admittedly, the median or mean travel time can better reflect the travel time distribution among stations, the minimum travel time can better illustrate the potential of using multi-hop routes, since the dispatching algorithm not only consider the mean travel time but the whole distribution. Therefore we use the comparison of minimum travel time of one-hop and two-hop routes to show the importance of multi-hop routes. For example, if a package routing algorithm only considers one-hop routes, it may lead to a longer delivery time for 40% OD pairs. Particularly, we color the OD pairs where $T_{\text{min}}^{\text{two-hop}} < T_{\text{min}}^{\text{one-hop}}$ in Fig.1, and the darkness of the color represents the difference between $T_{\text{min}}^{\text{two-hop}}$ and $T_{\text{min}}^{\text{one-hop}}$ (i.e., $T_{\text{min}}^{\text{one-hop}} - T_{\text{min}}^{\text{two-hop}}$). Note that for OD pairs where $T_{\text{min}}^{\text{one-hop}} < T_{\text{min}}^{\text{two-hop}}$, we set the value to zero instead of using a negative value for the readability of the plot.

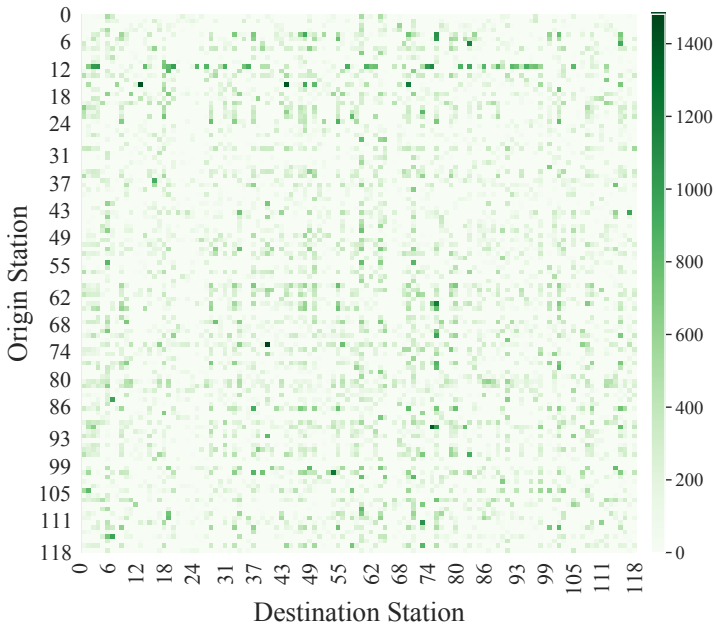


Fig. 1. $(T_{\text{min}}^{\text{one-hop}} - T_{\text{min}}^{\text{two-hop}})$ for OD pairs that $(T_{\text{min}}^{\text{two-hop}} < T_{\text{min}}^{\text{one-hop}})$.

Profit and Price Setting. Existing works either minimize the number of hops [10] or the time needed [12] for a package, while no study was conducted to combine the two factors in a practical view. In our vision, the earnings for the delivery platform are the customers' payment for the package delivery, and the costs include infrastructure (mailboxes, servers) and payment to the passengers who deliver the packages. For package routing, a profit model is needed to trade-off between delivery time and number of hops to minimize cost while satisfying the time constraint.

2.2 Public-transport-based Crowdsourcing Delivery

Based on the factors we discussed above, we envision a practical public-transport-based hitchhiking delivery as follows. The hitchhiking delivery system consists of four parts as shown in Fig. 2: a cloud server, a mailbox, a customer client and a passenger client. The mailbox connects with cloud server via Wi-Fi or cellular to update package information. Off-the-shelf box products have been developed and deployed as smart lockers in public area [4]. Customer and courier client are implemented as individual APPs so that customers and couriers can claim routes and packages. The system works as follows:

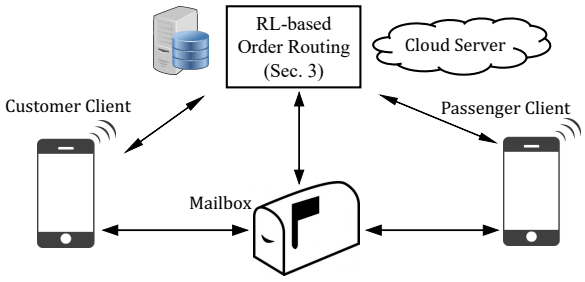


Fig. 2. Crowdsourcing delivery system overview

(1) A customer (sender) places the package in the mailbox of a mailbox array at his/her original station, and then input the destination of the package in the customer client. The customer can select a time-guarantee service like two hours, half day (6 hours), or same day at different pricing.
(2) Based on the timely information of all the packages and the environment (e.g., time, weather, supply/demand ratio), the server decides the next hop of all the packages across all the stations.
(3) When a participating passenger comes to a mailbox array, she can report her destination on her smartphone. Based on the passenger's input and the scheduled routes of the existing packages in the mailbox array, the system decides which package(s) to be carried by the passenger. Multiple packages will be dispatched to the passenger if they share the same next hop, and it's not beyond the passenger's capacity. The details on how we consider the passenger capacity is discussed in Section 3.3.
(4) When the passenger arrives at her/his destination (not necessarily the destination of the package), she/he places the package into the assigned mailbox and claims the completion of the task. Another passenger will come and repeat the above procedures until the package arrives at its destination. Passengers only need to pick up and place the packages at the source and the destination of their own hops, and there is no detour needed from the passenger. Therefore the delivery is purely ***hitchhiking***, and the passengers' routes or behavior are not altered.
(5) When the package arrives at its destination, the customer (receiver) gets notified and opens the mailbox to fetch the package. If the delivery exceeds the time constraint, the platform will compensate the customers, i.e., the customer will get his/her payment back if the package delivered exceeds the time guarantee.

3 PROFIT-ORIENTED DISPATCHING

In this paper, we focus on the order dispatching problem on the cloud server to maximize the platform's profit regards to the packages' time constraints. In this section, we first introduce a profit model with practical factor; then we show the details of the ETA module to model the estimated the delivery time between stations; lastly, we design a reinforcement learning algorithm to solve the routing problem of the packagers.

3.1 Profit Model

To calculate and optimize the profit, we need to estimate the delivery cost, in which we mainly consider the payment to the passengers who deliver the packages. For each passenger completing one hop in the delivery, he/she will get a payment. Ignoring infrastructure costs (e.g., mailbox installment), we have formalized the profit problem as follows:

$$\max_{h_i, M} \text{Profit} = \sum_{i=1}^{N} \text{CustPay}_i - \sum_{i=1}^{N} \sum_{j=1}^{h_i} \text{HopCost}_i^j - \sum_{k=1}^{M} \text{CustPay}_k \tag{1}$$

Here N is the number of all packages in consideration, M is the number of overdue packages, i.e., packages that do not meet time constraints. $CustPay_i$ is the payment from the customer with corresponding service type for package i, h_i is the total hops used to deliver package i, $HopCost_i^j$ is the payment to the passenger for the jth hop of package i,

In the optimization problem (1), the first part indicates the platform's earning from the customers' payment; the second part indicates the platform's expenses on payments to the hitchhiking passengers; the last part indicates the platform's expenses on payments to the customers due to package overdue.

The optimization indicates that, in order to maximize the profit for the platform, we need higher CustPay, lower HopCost, less hops for each package, and less overdue packages. The number of hops needed in the delivery is the key in the optimization. Less hops can save cost but may lead to longer delivery time and order overdue (as we discussed in Sec.2.1). That is why we need a ETA model to learn the travel time among different station pairs, and a RL model to choose a optimized route. The impact of CustPay and HopCost will be evaluated and

discussed in Sec.4.4. In the package dispatching, we focus on how to decrease the hops needed for each package while make more packages delivered within the time constraints.

3.2 ETA Module

Estimated time of arrival (ETA), a key concept used in transportation and logistics, is the time when a vehicle or person is expected to arrive at a certain place [57]. In hitchhiking delivery, ETA is the estimated delivery of the package between any two stations. Although the delivery time information can be learned by the RL algorithm from the simulator with historical transport data and package data, the huge action space will lead to a slow convergence process and degraded routing performance. Therefore, we use a ETA module to explicitly aggregate and model the delivery time information to shrink the action space and improve the learning performance.

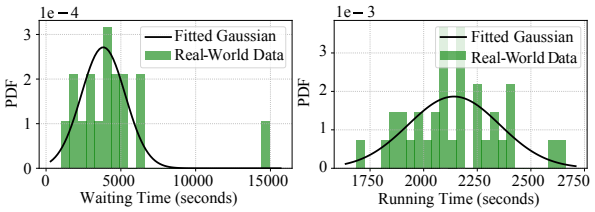
The delivery time for a package is divided into two parts: the *waiting time* and the *running time*. Specifically, waiting time is the time that the package is in the mailbox at the source or transfer station, running time is the time that the package is carried by the passenger on the move.

In the presence of multiple hops, both waiting and running time are composed of several segments at/between different stations. If there is a transfer station in the route, the duration between the package is placed and taken by the next passenger at the transfer station is also counted as part of the waiting time. Therefore the total delivery time T_{D} is computed as $T_{\mathrm{D}} = \sum_{i=1}^{n} T_{\mathrm{W}}^{i} + \sum_{i=1}^{n} T_{\mathrm{R}}^{i}$ where T_{W}^{i} is the waiting time at the ith transfer station, T_{R}^{i} is the running time from the ith transfer station in the route, and n is the total number of total hops.

Motivation of Parametric Gaussian. Many works have been proposed for ETA in the industry and research community. Learning methods are used in [31, 43, 67] to predict the ETA under a complex environment. However, these methods only output the estimated travel between an origin-destination (OD) pair without the variance information of the time distribution. In this paper, we use a parametric Gaussian ($N(\mu, \sigma)$) to model the waiting time and running time because Gaussian not only tells us the mean (μ) but also the variance (σ) of the distribution. We use the variance information to provide a statistical guarantee on the total delivery time [44]. Specifically, we use the quantile function of Gaussian, $\mu + \sigma\sqrt{2}\mathrm{erf}^{-1}(2p-1)$, to guarantee that the probability of the real-time being less than or equal to the estimated time is p (we call it on-time-rate parameter). Note that Poisson distribution may be more accurate for modeling the waiting time, but in the ETA-Aware RL-Dispatch, the purpose of time estimation is to have more information about the time distribution. Therefore, we use Gaussian since the sum of Gaussian is still Gaussian, which helps us estimate the total time as a parametric Gaussian and provide the statistical guarantee (Details in "ETA-Aware Action Filtering" paragraph in Section 3.3).

Waiting Time. Given the passengers' swipe-in and swipe-out timestamp in a transport system, the waiting time of a package to the next station is calculated as the time between the placement of the package and the swipe-in time of the passenger to the next station.

Based on the experiment observation, we find that the waiting time T_{W}^{i} follows Gaussian distribution, i.e., $T_{\mathrm{W}}^{i} \sim N\left(\mu_{\mathrm{W}}^{i}, (\sigma_{\mathrm{W}}^{i})^2\right)$, where μ_{W}^{i} and $(\sigma_{\mathrm{W}}^{i})^2$ are the mean and variance of T_{W}^{i}. Figure 3a shows the distribution of waiting time as well as the corresponding fitted Gaussian at a station.



(a) Waiting time (b) Running time

Fig. 3. Gaussian fitting example at a station pair and a time slot (1 week data assuming 1% passengers willingness)

Since the number of commuters in the public transport system varies dramatically in different time slots of a day, we split a day into 24 slots. Within each slot, μ_{W}^{i} and $(\sigma_{\mathrm{W}}^{i})^2$ are considered as constants. Maximum likelihood is used to estimate μ_{W}^{i} and σ_{W}^{i}. $\hat{\mu}_{\mathrm{W}}^{i} = \frac{\sum^{n} t_{\mathrm{W}}^{i}}{n}$, $(\hat{\sigma}_{\mathrm{W}}^{i})^2 = \frac{\sum^{n} (t_{\mathrm{W}}^{i} - \hat{\mu}_{\mathrm{W}}^{i})^2}{n}$, where $\hat{\mu}_{\mathrm{W}}^{i}$ and $\hat{\sigma}_{\mathrm{W}}^{i}$ are estimation of μ_{W}^{i} and

σ^i_{W}. We use Kolmogorov-Smirnov test (K-S test) [27] to conduct the hypothesis test and decide whether waiting time fits Gaussian distribution. Based on the K-S test result on the real world data with 117 stations, more than 90% OD pairs pass the K-S test for the waiting time at the 5% significance level. Therefore it is reasonable to build a Gaussian distribution model for the waiting time.

Passengers' Willingness. Note that passengers' willingness impacts the waiting time estimation, since not all passengers are willing to take a package. Passengers' willingness is not considered in existing crowdsourcing delivery works, implicitly assuming all passengers as couriers. However, this assumption is not practical as indicated by the example of Uber (not all private car drivers are willing to be Uber drivers). Given that not all passengers are willing to be couriers, how many passengers participate actually impacts the delivery time of a package. In this paper, we use a participate rate indicator ρ to model a global willingness where $\rho = 0.1$ means 10% of passengers are willing to take a package. Note that ρ is a parameter that is impacted by many factors like price setting and time. We evaluate the system's performance with potential ρ values in Fig. 12 and 13. The waiting time is only used to estimate the time cost if we choose to send the package to a specific station, so we use the first arrival passenger between the station pair under the participate rate to estimate the waiting time. Therefore, the estimation of the waiting time is independent of the dispatching policy.

Running Time. The running time is calculated as the time between the passengers' swipe-in and swipe-out for his hop, i.e., $T^i_{\mathrm{R}} = \left(T^k_{\mathrm{out}} - T^k_{\mathrm{in}}\right)$. Here T^k_{out} is the swipe-out time stamp of the kth passenger. Fig. 3b shows the running time distribution and its fitted Gaussian in a station pair. 96% OD pairs pass the K-S test for the running time. Therefore, we build a Gaussian distribution model for the running time, that is $T^i_{\mathrm{R}} \sim N\left(\mu^i_{\mathrm{R}}, (\sigma^i_{\mathrm{R}})^2\right)$.

In our design, the swipe-in/swipe-out time between a OD pair are utilized to estimate the running time. Although it has been studied in [21, 41, 49] that people may choose different routes even for the the same OD pair, which may cause variation in running time. It has been concluded in [49] that passenger's choice tends to be fixed at specific period of time. Specifically, passengers care more about total travel time during peak hours, whereas comfort (e.g., less transfer) is of more concern during off-peaks. When a day is divided into small time slots (e.g., 1 hour), passenger's route choice is almost fixed and its running time follows Gaussian distribution. This is verified by our K-S test and observations in Fig. 3b.

ETA Output. Based on the analysis above, we conclude that under fixed time slot and participate rate, running time and waiting time of a specific OD pair can be fitted with Gaussian, which is verified with K-S test on one-month Shenzhen subway data. The ETA results of a city's public transportation is calculated with different participate rate settings are stored in as a 4-D tensor $[\mu_{\mathrm{W}}, \sigma_{\mathrm{W}}, \mu_{\mathrm{R}}, \sigma_{\mathrm{R}}] = H(\rho, t, o, d)$, where μ_{W} and σ_{W} are the Gaussian parameters of the estimated waiting time, μ_{R} and σ_{R} are the Gaussian parameters of the estimated running time, respectively between station o and d, ρ is the participate rate, and t is the current time slot.

3.3 ETA-Aware RL-Dispatch Algorithm

In this paper, we design a reinforcement learning algorithm to route the packages with ETA awareness. Specifically, we use a deep RL method to learn the optimal route for each package from massive historical transport data and package data under different price setting (CustPay and HopCost) and environment factors (weather, time, supply, and demand). An ETA-aware action filter is designed to eliminate infeasible actions based on ETA to accelerate offline training and improve online routing.

Background. Reinforcement learning (RL) is typically used to model sequential decision-making problems [60]. In RL's setting, an agent takes an action at each timestamp and earns some reward, where the action will change the states of environment and the agent. Then, the environment gives a reward for the decisions back to the agent and turns to the next state. In the learning process, an agent learns how to act in response to the states with the goal to maximize the total rewards, along with this interaction with the environment.

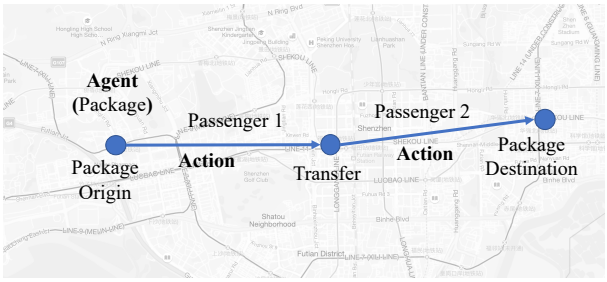


Fig. 4. Dispatching problem in hitchhiking delivery

RL Formulation. With dynamic states of packages and passengers in real world, package routing is naturally a sequential decision-making problem. In our problem setting as shown in Fig. 4, the agents are the packages in the system. The decision center acts as a meta-agent that makes decisions for all packages in a centralized way. The meta-agent makes decisions (i.e., choose the next hop for each package) in consecutive time slots (e.g., 5 minute each) and predicts the return of total revenue (i.e., platform profit) over the next few hours. The environment is the packages and passengers moving in the system, and some context such as weather. The components of the RL are defined as follows.

Agent. We consider each package as a agent, since profit is calculated on package level in optimization problem (1). The decision center acts as a meta-agent and makes decisions for all the agents, so that the routing of all packages between the stations can be learned and shared among all the packages across the stations.

State space $\mathbf{s}_t \in \mathcal{S}$. There are two types of states at each time t, global states and private states. The global state considers the demand, supply and some contextual information. In existing works, transport data (e.g., taxi data [10, 36], bus data [11]) are used to build the dispatching model and evaluate the dispatching performance. The transport data restore the crowdsourcing delivery problem from supplying view, while demanding aspects are seldom studied due to unavailable of package order data. In ETA-Aware RL-Dispatch we use the supply and demand information in as state features to further mimic real-world applications. Specifically,

- *Demand states*: a 1-D distribution of waiting packages at all stations, and a 2-D (a $N_{\text{station}} \times N_{\text{station}}$ matrix) distribution of delivering packages across all station pairs.
- *Supply states*: a 1-D distribution of of entering passengers at all stations, and a 2-D (a $N_{\text{station}} \times N_{\text{station}}$ matrix) distribution of commuting passengers across all station pairs.
- *Contextual states*: the information is represented as weather, date, weekday and time.

The private state of an agent i considers the package's source and the destination, total time constraint, remaining delivery time, hop history, CustPay, and HopCost.

Action $\mathbf{a}_t$. The action of an individual agent specifies the next hop of the package, specifically the next station that the package should go among all the stations.

Reward function. We design a reward function towards the total profit in (1). The reward function is set to 0 before the package reaches its destination; otherwise it is set as

$$r_t^i = \text{CustPay}_i \times \phi(i) - \sum_{j=1}^{h_i} \text{HopCost} \tag{2}$$

where $\phi(\cdot)$ is an indicator function, i.e., $\phi(i) = 1$ if the package i is not overdue and $\phi(i) = 0$ if the package i is overdue. That is, the reward is not counted until the package reaches the destination. The reward function let the algorithm choose the route with minimum hops as long as the time constraint is satisfied.

Action Value Computation. In RL, the future reward, also known as return, $G_t = \sum_{k=0}^{\infty} \gamma^k r_{t+k+1}$, is a total sum of discounted rewards going forward with discount factor $\gamma \in (0, 1]$. The goal of the agent is to maximize the expected return from each state s_t. The action value $Q_\pi(s, a) = \mathbb{E}_\pi[G_t | \mathbf{s}_t = s, \mathbf{a}_t = a]$ is the expected return for selecting action a in state s and following policy π.

In value-based model-free reinforcement learning methods, the action value function is represented using a function approximator, such as a neural network. Let $Q(s, a; \theta)$ be an approximate action-value function with parameters θ. The updates to θ can be derived from a variety of reinforcement learning algorithms. We choose deep Q network (DQN) with experience replay to approximate the action-value function and update the parameters because DQN is capable of handling high-dimensional sensory inputs (states) [38], given the large amount of stations and packages in consideration. In experience replay, all the episode steps $e_t = (\mathbf{s_t}, \mathbf{a_t}, r_t, \mathbf{s_{t+1}})$ are stored in a replay memory $D_t = e_1, \ldots, e_t$. D_t has experience tuples over many episodes. At each iteration, DQN update the parameter using the following loss function:

$$\mathcal{L}(\theta) = \mathbb{E}_{(s,a,r,s') \sim U(D)} \left[\left(r + \gamma \max_{a'} Q(s', a'; \theta^-) - Q(s, a; \theta) \right)^2 \right] \tag{3}$$

where $U(D)$ is a uniform distribution over the replay memory D. θ^- are the network parameters used to compute the target at iteration i. The target network parameters are only updated with θ^- every some C steps and are held fixed between individual updates.

ETA-Aware Action Filtering. Since the action is the next station choice for the package, the action space is large considering the hundreds of transport stations in a city. The large action space leads to slow convergence in training and also impacts the routing performance. Given the ETA information we collected offline, we use this prior knowledge in an action filter process to accelerate the training and provide a statistical guarantee on the delivery time. In the filtering, the probability of a potential action a_j is given by

$$P_{t,a_j} = \begin{cases} 1, & \text{if } T_{a_j} \leq r_i \\ 0, & \text{otherwise} \end{cases} \tag{4}$$

where T_{a_j} is the estimated delivery time from the current station o_t to the destination d_i, and r_i is the remaining delivery time of the package i. Specifically, T_{a_j} is calculated as

$$T_{a_j} = T_{\mathrm{R}}^{o_t \to s_j} + T_{\mathrm{W}}^{s_j} + T_{\mathrm{R}}^{s_j \to d_i} \tag{5}$$

where s_j is the next station decided by action a_j of package i at time t. That is, the estimated delivery time is the sum of the running time from the current station to the next transfer station, the waiting time at the next transfer station, and the running time from the next transfer station to the destination. If the next station happens to be the destination, we have $T_{a_j} = T^{o_t \to d_i}$.

Based on the ETA results $[\mu_{\mathrm{W}}, \sigma_{\mathrm{W}}, \mu_{\mathrm{R}}, \sigma_{\mathrm{R}}] = H(\rho, t, o, d)$, each term in equation (5) is calculated with the following quantile function (i.e., inverse CDF) of Gaussian, respectively,

$$T = \mu + \sigma\sqrt{2}\mathrm{erf}^{-1}(2p - 1) \tag{6}$$

where erf() is the Error function [59], and $0 < p < 1$ is a probability. We call p the on-time-rate parameter, and use it to guarantee that the probability of the real delivery time being less than or equal to the estimated time is p. For example, if we set $p = 0.9$, we provide a statistical guarantee that 90% of real delivery time is less than the estimated delivery time. We use this mechanism to provide a statistical guarantee on the total delivery time to the time constraints. The choice of the parameter p is a trade-off between delivery rate and profit. A larger p value provides more guarantee on the delivery time but eliminates some potential low-cost routes. If the package $r_i \leq 0$, which means the package is already late, the package will be given priority to dispatch first.

After removing the invalid actions, we select the action with epsilon greedy [50]. Epsilon greedy chooses the action with highest action value but still gives chance to other actions with some low probability (ϵ).

ALGORITHM 1: ETA-Aware RL-Dispatch Algorithm

Input: ETA tensor $H(\rho, t, o, d)$

Initialize the states and the DQN.
for each time slot t **do**
  ***Update states*:** updates the global states and private states for all packages based on the environment changes at t;
  ***Sort packages*:** sort all the un-dispatched packages by the remaining time ascending
  for package i in the un-dispatched packaged list **do**
    ***Calculate Q value*:** computes the actions values $Q(s_t, a_j)$ for all the packages of all actions a_j through DQN;
    ***ETA-aware action filtering*:** filter out invalid actions based on ETA results;
    ***Action selection*:** select the action based on epsilon greedy for the package;
    ***Dispatch packages*:** dispatches the packages with the actions chosen.
  end
end

Based on the components, the ETA-Aware RL-Dispatch algorithm is designed as shown in Fig. 5 and summarized in Algorithm 1. The main idea is that, in each time slot, ETA-Aware RL-Dispatch aims to maximize the expected gain of the platform.

We use a sequential dispatch scheme as indicated in (3) because we want a more "urgent" package can be dispatched first if the number of dispatched packages exceeds a passenger's capacity. Note that when we ignore the passenger's capacity, this sequential scheme is the same as a parallel scheme that consider all the packages at the same time.

In the model training process, we build a simulator that can restore the environment evolution based on the passenger data, package data, and the context data. The details of the simulator is discussed in Section 4.2.

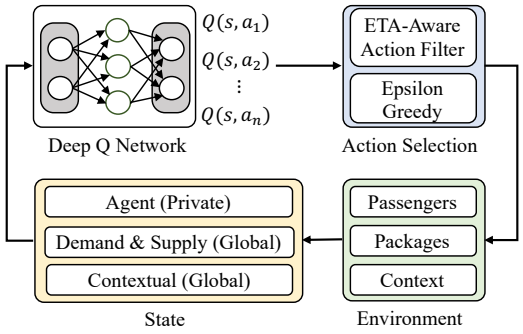


Fig. 5. ETA-Aware RL-Dispatch algorithm

4 EVALUATION

In this section, we first give a description of the passenger data and the package data we used; then we explain how we use the data to build a simulator for training the ETA-Aware RL-Dispatch algorithm; after that we describe the evaluation methodology we used (i.e., metrics and baselines); lastly we show the evaluation results under different practical factor settings.

4.1 Data-set Description

Two data-sets are used in our evaluation: A subway swipes data-set and a package delivery data-set. We share the data used in the project to the community for other researchers to validate our results and conduct further research.[2][1].

[2] https://tianchi.aliyun.com/dataset/dataDetail?dataId=106807

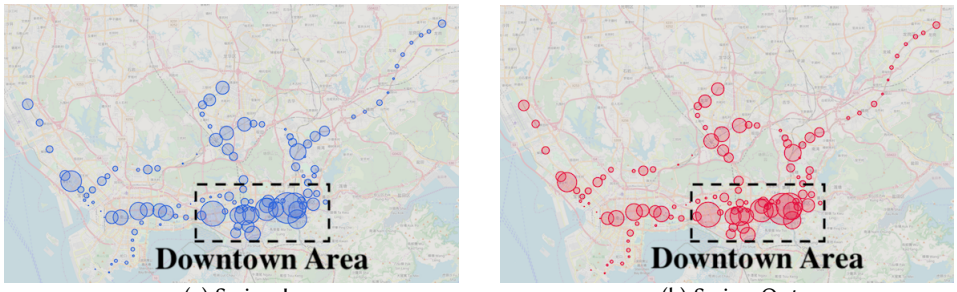
Table 1. Subway data-set details

Subway Data-set Summary	
Collection Period	09/01/18-11/30/18
Number of Swipes	135,092,970
Data Size	35 GB
Number of Cards	6,282,235
Format	
Swipe-in time	Swipe-in station
Swipe-out time	Swipe-out station

Table 2. Package data-set details

Package Data-set Summary	
Collection Period	06/01/19-06/30/19
Number of Packages	10,836,543
Data Size	14 GB
Number of Users	2,081,425
Format	
Sender latitude/longitude	Receiver latitude/longitude
Send time	Receive time

The subway data-set contains the swipe-in and swipe-out information from Shenzhen subway system. A detailed description of the data-set can be found in Table 1. In the data-set, the swipe-in and swipe-out record are in pair to indicate a passenger's traveling from one station to another.



(a) Swipe-In (b) Swipe-Out

Fig. 6. Density of swipe-in and swipe-out at each station in Shenzhen subway system

There are 10 lines and 266 stations in Shenzhen subway system. The number of swipes-in and swipe-out at each station is visualized in Fig. 6. The CDF of the trip duration is shown in Fig. 7. We observe that more than 90% of trips are completed within 50 minutes, while longest trips last more than 2.5 hours. In Fig. 8 we plot the histogram of the number of daily swipes. 1.5 million daily swipes occur in the system and this number decreases slightly in Saturday and Sunday.

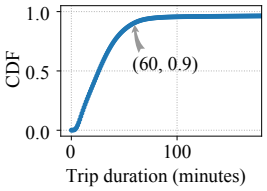


Fig. 7. Trip duration

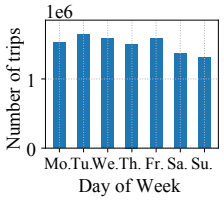


Fig. 8. Daily swipes

The package delivery data-set is collected from Alibaba Group [42], who operates one of the largest city-wide delivery platform in China. A detailed description of the data-set can be found in Table 2. A visualization of the sender and receiver location can be found in Fig. 11. A similar pattern can be found between passengers density distribution (Fig. 6) and packages density distribution (Fig. 11). The CDF of the package delivery duration is in Fig. 9. Since it is a city-wide platform for instant delivery service, most of the deliveries are completed within 80 minutes. The histogram of number of package delivery in a day is

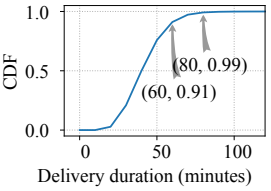


Fig. 9. Delivery time

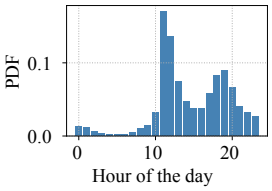
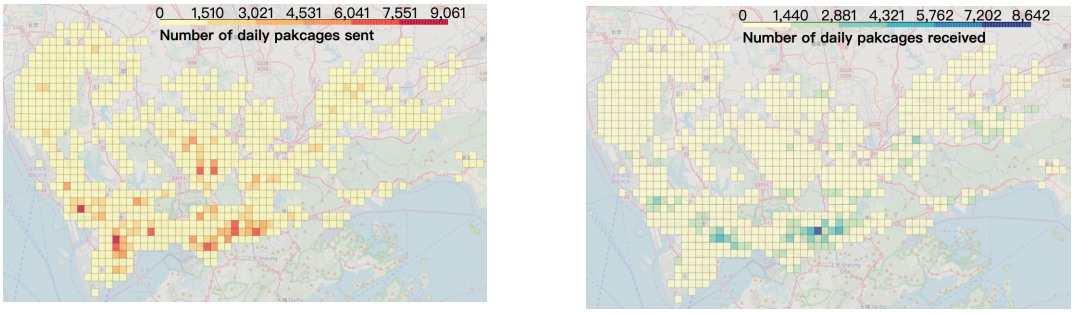


Fig. 10. Hourly PDF

Fig. 10. We observe two peaks around 11:00 AM and 7:00 PM because food delivery composes a major portion on the platform.



(a) Senders' location

(b) Receivers' location

Fig. 11. The packages density in Shenzhen

We also collected weather data and datetime data (time, date, weekday) as the context data.

4.2 Simulator

Unlike conventional supervised learning problems where the data is stationary, and the algorithm is evaluated with a training-testing paradigm, RL's interactive nature introduces training and evaluation difficulties. Therefore, we build a simulator based on real-world historical data, which can mimic the delivery process of the packages perfectly since we don't need detours of the passengers.

Simulator Implementation We used 75% of subway package delivery data to build the simulator and 25% of data for evaluation. For each package, there are three states: "Waiting", "On the way", and "Delivered". "Wating" means a package is waiting in the mailbox either at the origin station or a transfer station between hops. "On the way" means a package has been picked up by a passenger and is in the delivery progress to a transfer station or the destination. "Delivered" means the package has been placed at its destination by the passenger of the last hop. For example, for a package delivered with two hops, its states sequence should have been "Waiting", "On the way", "Waiting", "On the way", "Delivered". In order to mimic a real-world scenario where only partial passengers are willing to carry packages, we set different participating rate by randomly choosing the corresponding percentage of passengers for package delivery.

For each time interval of the dispatching system ($t = 0$ to T), the simulator works as follows.

(1) *Add new packages*: Check package data-set and add packages emerged in $t - 1$ into the packages set and set their states as "Waiting".

(2) *Update "On the way" packages' states*: For "On the way" packages, check corresponding passengers' data at t to find the packages that arrived at the transfer station or the destination, and set the package' state as "Waiting" or "Delivered".

(3) *Package dispatching*: For all undispatched "Waiting" packages, decide the next hop of the packages with ETA-Aware RL-Dispatch.

4.3 Evaluation Methodology

The time and place of the packages to be delivered are generated following the patterns we found in the package delivery data-set. Specifically, for each day, 1% (around 3,000) of packages extracted from package data-set are to be delivered for evaluation. The nearest subway station of sender and receiver location are assigned as source and destination station. For the choice of price setting in the evaluation, we follow the setting of Amazon Prime

Now [39], and set the CustPay as $5. We set HopCost to as a unit value ($1) to show the potential profit of the system. For the time constraints, we set the time constraints for each package individually from 1 to 8 hours to guarantee same-day delivery, and the time constraints are set proportional to the OD distance. For the on-time-rate parameter p, we set $p = 0.9$, which is comparable to the upper bound performance of that in CrowdExpress [6] and PPtaxi [10].

Table 3. Hyper-parameter settings in RL

Hyper-parameter	Setting	Hyper-parameter	Setting
Learning rate	0.01	Hidden layer size	128, 64, 32
Memory D Size	1e+6	Discounter factor γ	0.9
Mini-batch size	512	epsilon	0.5~0.9
Target θ^- update period C	3000	# of episode	25
State feature vector size	40	Time steps within episode	144

4.3.1 Parameter Settings in RL. The detailed parameter settings for ETA-Aware RL-Dispatch are provided in Table 3. We define each day as an "episode" in the training and testing since we only consider same-day delivery. We split each episode into 144 time steps, where each step is 10 minutes, based on the tradeoff of passengers' mobility, order batching, and computation cost. Other parameters are set based on related RL studies and cross validation.

4.3.2 Metrics. In the evaluation, we mainly focus on three metrics: profit rate, delivery rate, and number of couriers per package. The definition of these three metrics are listed in the following:

$$\text{Profit Rate} = \frac{\text{earning of packages delivered} - \text{cost}}{\text{\# of packages placed}} \tag{7}$$

Profit rate reflects the algorithm's ability to choose the economical route for the package. Packages' delivery cost is defined as the total number of couriers needed. The unit cost is set as $1 per courier per hop, while the payment from the customers is set as $5 per package. The earnings of delivering packages depend on customer payment and whether the package is delivered in time. The impact of payment setting from customers and to couriers will be discussed in the following subsections.

$$\text{Deliver Rate} = \frac{\text{\# of packages delivered}}{\text{\# of packages placed}} \tag{8}$$

Delivery rate indicates how many packages we can successfully deliver within the time constraints. It reflects the algorithm's ability to deliver packages in time, i.e. delivered within time constraints.

$$\text{Average Hop} = \frac{\text{\# of couriers needed}}{\text{\# of packages sent}} \tag{9}$$

Average hop of the package, which is also the number of couriers per package, is the cost of each package since a fixed amount of money is paid to the courier.

Note that a placed package is a package should be delivered (but not necessarily sent due to lack of appropriate couriers), a sent package is a package picked by the courier (but not necessarily delivered due to time constraints), a delivered package is a package sent to the final station within time constraints.

Profit rate is the key indicator to show the algorithm's ability to deliver the package within time constraints. For non-profit organizations whose aim is to serve public but not profit, deliver rate will be more important.

4.3.3 Baseline Approaches. Eight baselines are compared with our ETA-Aware RL-Dispatch algorithm, and we group them into two categories: *system baselines* and *RL baselines*. We use system baselines (i.e., PPtaxi, CPTS, CrowdExpress, Min-cost, Cost-greedy, Distance-greedy) to show the effectiveness of our multi-hop, profit-oriented crowdsourcing delivery system with practical factors considered. We use RL baselins (i.e., A2C, Non-ETA) to show the effectiveness of considering ETA and action space shrinking in our ETA-Aware RL-Dispatch. Moreover, an oracle baseline is implemented to show the performance upper bound on the data-set.

• **PPtaxi:** "PPtaxi" [10] maximizes the delivery rate with time constraints when using taxis for package delivery, and Multivariate Gaussian and Bayesian are used to predict the future passengers.

• **CPTS:** CPTS [12] (i.e., crowdsourced public transportation systems) minimizes the delivery time for the packages by modeling the packages dispatching as an instance of the multicommodity flow problem and proposes a heuristic solution to solve the underlying integer linear programming problem.

• **CrowdExpress:** CrowdExpress [6] builds a package transport network offline using the historical trajectory data and discovers the path with the maximum arriving-on-time probability in scheduling.

• **Min-cost:** "Min-cost" [55] solves a large-scale mobile crowd-tasking problem by minimizing the delivery cost without delivery time consideration.

• **Cost-greedy:** A "cost-greedy" algorithm only assigns the package to the courier that heads to the destination of the package directly.

• **Distance-greedy:** A "distance-greedy" algorithm always dispatches the package to the courier if the courier can take the package closer to the destination.

• **A2C:** To show the effectiveness of the action shrinking design in DQN, we implement a contextual multi-agent actor-critic (A2C) algorithm [35] as an RL variation.

• **Non-ETA:** To show the effectiveness of the ETA module in ETA-Aware RL-Dispatch, we implement a reinforcement learning method without ETA module as a baseline.

• **Oracle:** In "Oracle" algorithm, the system can "foresee" all the passengers in the future and choose the route with minimum hops within the time constraints. Note that the Oracle is not aware of the future package information, since package mobility modeling is out of this paper's scope. The Oracle provides the upper bound of the performance that any algorithm can achieve with perfect time estimation and routing.

4.4 Evaluation Results

Overall Results. By aggregating the evaluation results under different settings, we show the overall results are as follows. The average profit rate is $3.4 for ETA-Aware RL-Dispatch, which improves the state-of-the-art baselines significantly (i.e., $2.7 for CrowdExpress, $2.5 for Cost-Greedy, $2.3 for PPtaxi, $2.2 for Min-cost and CPTs, and $1.2 for Distance-Greedy), and also beats RL baselines (i.e., $3.1 for Non-ETA and A2C). The average delivery rate is 0.89 for ETA-Aware RL-Dispatch, which outperforms both the state-of-the-art baselines (i.e., 0.75 for CrowdExpress, 0.66 for Cost-Greedy, 0.68 for PPtaxi, 0.66 for Min-cost and CPTS, and 0.72 for Distance-Greedy) significantly, and also outperforms RL baselines (i.e., 0.83 for Non-ETA, and 0.82 for A2C). The comparison with the state-of-the-art baselines shows that we can significantly improve both the profit rate and delivery rate by considering multi-hop routes and time constraints in a practical setting. The comparison with the RL baselines shows that generic RL algorithms can achieve fair results, and an ETA module with a statistical guarantee can further improve the profit rate and delivery rate by 9% and 8%. The comparison between cost-greedy and ETA-Aware RL-Dispatch shows that multi-hop routes can save time in some cases.

Impact of Passengers' Willingness. As a crowdsourcing based approach, hitchhiking relies on passengers' participation as couriers. Participate rate measures how many passengers can be viewed as couriers. We aggregate the results by participate rate as shown in Fig. 12 and Fig.13. The participate rate has an impact on both profit rate and delivery rate. The more passengers participate, the more profit made and more packages delivered; but

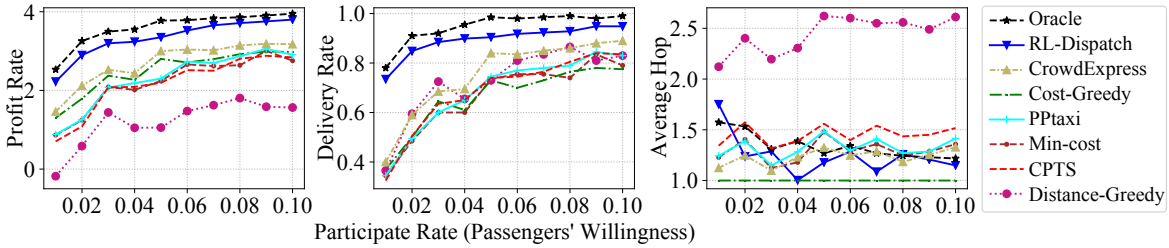


Fig. 12. Impact of participate rate (System baselines).

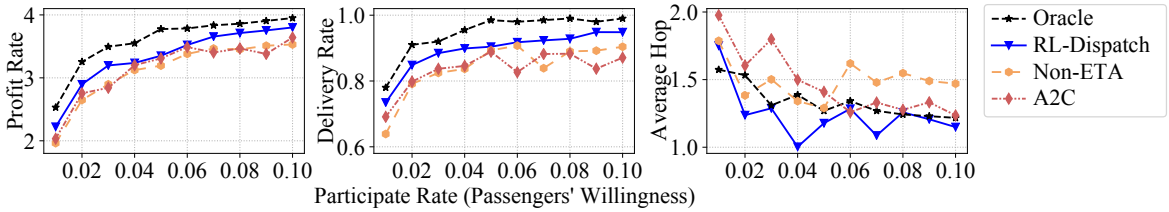


Fig. 13. Impact of participate rate (RL baselines).

the impact of participate rate decreases as it grows. The results also suggest the importance of considering the supply/demand ratio in designing a crowdsourcing delivery system.

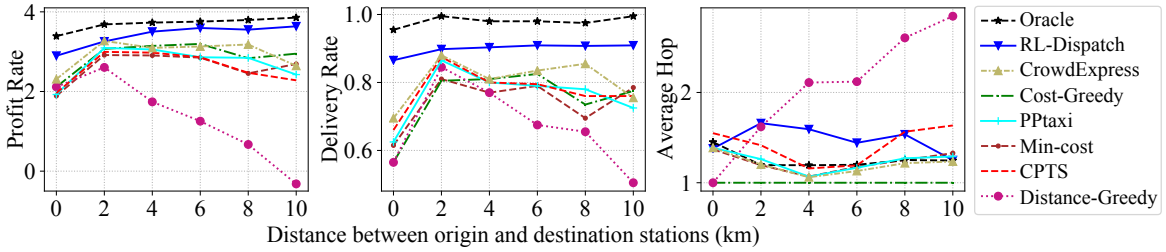


Fig. 14. Impact of OD distance (System baselines).

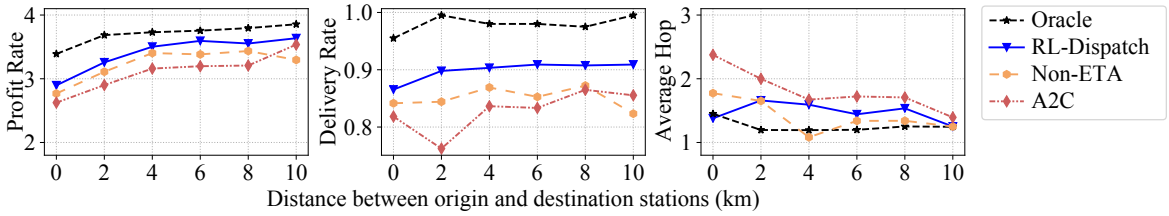


Fig. 15. Impact of OD distance (RL baselines).

Impact of Origin Destination Distance. The impact of distance between origin and destination station is also evaluated by varying this distance from 1 to 10 km. The results are shown in Fig. 14 and Fig. 15. We can see that ETA-Aware RL-Dispatch is not sensitive to the distance. While the performance of distance greedy algorithm degrades as the distance grows. The delivery rates of the system baselines are significantly worse near 1km because ODs with 1km distance are usually neighboring stations, and people prefer other transportation (e.g., walk, bike, bus) instead of the subway for these ODs. RL algorithms can resolve the problem by learning from historical data and choosing multi-hop routes.

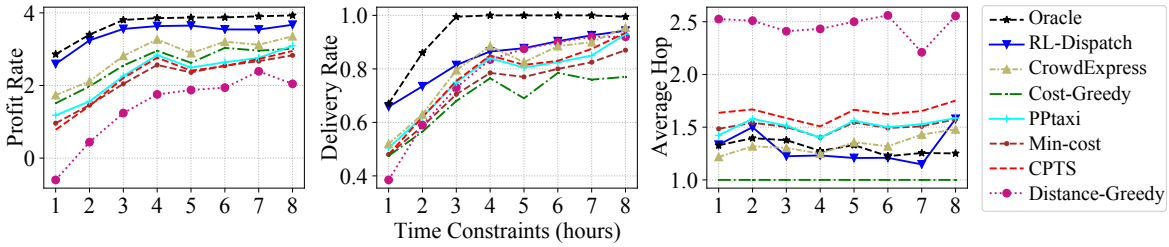


Fig. 16. Impact of delivery time constraints (System baselines).

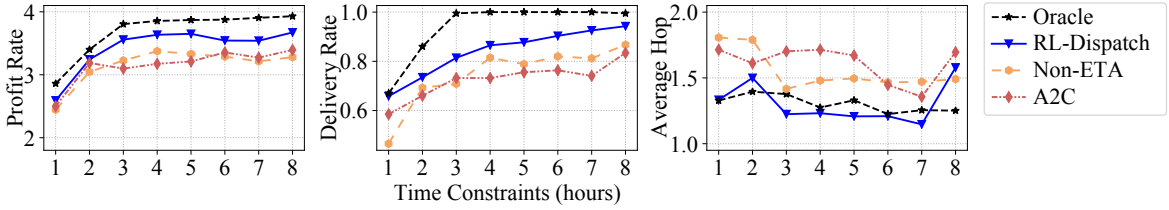


Fig. 17. Impact of delivery time constraints (RL baselines).

Impact of Time Constraints. The performance comparison under different time constraints can be found in Fig. 16 and Fig. 17. Both profit and delivery rates increase as time constraints get looser because more routes can be considered when we have more time.

Impact of Price Setting. We also conduct experiments to evaluate the performance of the hitchhiking system under different price settings. The payment from the customer is tuned from \$2 to \$7 and the result is in Fig. 18. We can see that the profit rate grows stably as the price increases, and the delivery rate does not change much. This shows that our design is consistent under different price settings.

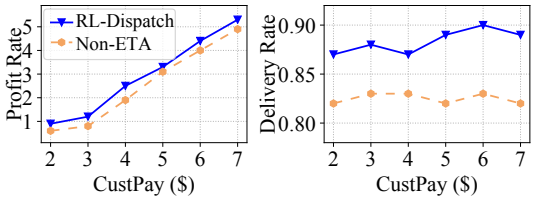


Fig. 18. Impact of Price Setting

Impact of ETA Module on the Training Process. To show the training acceleration effectiveness of adopting the ETA module, we follow the idea in [5] and compare the number of episodes used to reach the convergence in the training of our RL with and without the ETA module under different parameters (i.e., participate rate). As shown in Fig. 19, the mean episodes is 13.7 for RL-Dispatch, which is a 22% reduction compared to that for RL without ETA (17.5).

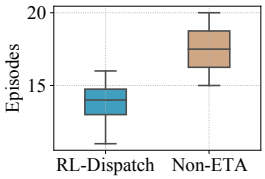


Fig. 19. Training Acceleration

Scalability and Latency. We set the time step as 5 minutes based on the tradeoff of passengers' mobility in the public transportation system. For the RL model, the offline model training takes several hours, which enables a daily or half daily updating of the model; the online model prediction takes several seconds, which enables real-time dispatching decisions. The above statistics are based on the following hardware and software configurations: Linux, Intel Core i7, 16G memory. To enable the application in multiple cities, the scalability of the model can be achieved by parallel training and prediction in different cities.

4.5 In-field Survey

To verify the practicability of our design, we conduct a small-scale in-field survey in two Shenzhen subway stations on passengers' willingness to carry the packages. We randomly choose strange passengers in the station

and ask whether she/he is willing to take a package during the trip if $1 is paid to him/her. Around 150 passengers are asked at each station in two days, and we found that the participate rate varies between 1% and 10% at different hours, which is enough to support the system and make profit according to Fig.12 and 13.

5 DISCUSSION

Lessons Learned: During working on the paper, we learned the following insights:
(1) Crowdsourcing delivery with public transport should consider multi-hop routes (Fig.1). Ignoring the transfers degrades the performance, as shown by the cost-greedy baseline in Fig. 12, 14, and 16.
(2) Crowdsourcing delivery with public transport should consider the passengers' participate rate. Simply considering the distance between stations leads to more hops than necessary, as shown by the distance-greedy baseline in Fig. 12, 14 and 16, because the delivery time between stations also depends on the number of passengers traveling.
(3) Fastest route is not necessarily the profit-optimal route, and a profit model is needed to balance the time and cost. A cost-optimal dispatching algorithm may fail to meet time constraints and leads to less profit, as shown by the min-cost baseline in Fig. 12, 14, and 16.
(4) RL can learn profit-optimal routes from historical passengers' and packages' data in consideration of time constraints and multi-hops, as indicated by the gap between our algorithm and system baselines (i.e., CrowdExpress, PPtaxi, CPTS) in Fig. 12, 14, and 16.
(5) Explicit consideration of ETA information can further improve the RL algorithm's performance, as indicated by the gap between our algorithm and RL baselines (i.e., Non-ETA and A2C) in Fig. 13, 15, and 17.
(6) There is still a gap between our algorithm and oracle as shown in Fig. 12-17. Filling this gap relies on the accurate prediction of passengers' swipe-in and swipe-out time, and destination station on individual level, while much more data is needed to achieve this.

Generalization to other Public Transportation Systems. Although our evaluation focuses on subway systems, the design is generic enough to be applied to other city-wide public transportation systems as long as two core assumptions are fulfilled: (1) Waiting time and running time can be estimated; (2) Package can be placed, stored and picked up at the station. The first assumption can be achieved if historical travel data are collected, which has been achieved in many public transportation systems. The second assumption can be achieved since self-aid pick-up and drop-off box has been developed like Hive Box [4]. Moreover, our design can be applied to an emerging topic: multi-hop ride sharing [15], where transfers are allowed in the ride sharing. It has been proved competitive against other modes of transportation and has the potential to greatly increase ride availability and city connectedness[52]. The application of multi-hop ride sharing requires reliability, which can be provided by our time-constrained strategy.

User Incentive. It has been shown in the survey that $1 is enough to motivate participants and make profit for the platform. In traditional crowdsourcing systems, incentive mechanism is a critical topic since the crowd is actually "hired" [18, 45]. However, in our system, real hitchhiking is achieved since little extra labor is needed to deliver the packages.

Timeout and Undelivered Packages. Although the profit model and the ETA module are designed to reduce the number of timeout and undelivered packages, technical designs cannot avoid the problem entirely. Because in a hitchhiking system, we cannot control passengers as couriers, which is similar to the unavailability of Uber in extreme weather. The solutions of undelivered packages usually need policy-level consideration (e.g., a dedicated team for corner cases), out of this paper's scope. We envision future works with technical improvements such as dynamic pricing that can boost the priority of late packages.

Limitations and Future Works. (1) Price setting is a complex topic while is simplified in this paper. Further studies can be made on price setting for the commercialization of hitchhiking delivery.
(2) Time constraints setting is not discussed in detail in the paper. It should not be either too loose (customers do not want to wait) or too tight (package is impossible to be delivered in time). Further studies can be built on this topic with focus on commercial psychology.
(3) Time estimation. Running time and estimation time is impact by multiple settings like time slot, weather, and weekday. In this paper, we mainly consider the impact of the time slot and assume a fixed value for each time slot. Although it has been proved in the evaluation that the setting is enough to make profit, we believe finer-grained modeling with more features can improve the time estimation results and the system performance.
(4) Advanced RL techniques. Since the essential contribution of the paper is the design of a practical hitchhiking delivery system and and the formulation of a profit-oriented dispatching problem, a straightforward RL solution is used to solve the dispatching problem. Given the nature of sparse and delayed rewards and conflicting objects of hitchhiking dispatching problem, advanced RL techniques such as hierarchical RL [30] and constrained RL [40] can be used.

Ethics, Privacy and Data Release. The data-sets used in our research are rendered irreversible anonymous in such a way that the individual is no longer identifiable. Hence the research conducted complies with the general data protection regulation (GDPR)[13]. We did not utilize personal information from the couriers, e.g., age, gender, income, to protect the privacy of the passengers. So, IRB is approved. We will release one month of our data we collected for the research community to validate our results and conduct further research. The data-set has been made public on Tianchi [1], the official data release website of Alibaba. There is a strict internal regulatory process to review the data-set before publication. Privacy protection measures are also conducted (details in [1]).

6 RELATED WORKS

6.1 Crowdsourcing Delivery

Studies on crowdsourcing delivery can be categorized based on the transport approach. Most existing studies focus on the taxis possibly due to the data availability [6–8, 29, 55]. TaxiExp [8] is the first work to exploit the taxis for package delivery, in which a two-phase solution is proposed for "hub identification" and "inter-hub routing". In [55], an optimal network min-cost flow algorithm is proposed to dispatch the packages with shortest path. FooDNet [36] proposes the first taxi-based crowdsourcing delivery food delivery. PPtaxi [10] proposes a graph-based method for package delivery with transfers, in which time constraints are evaluated but not tackled in design. Similar idea is used in CrowdExpress [6] to lower the cost and accelerate package deliveries simultaneously with a package transport network.

Private car is another approach for crowdsourcing delivery services. A 3D landmark graph is constructed in Car4Pac [54] for cost estimation and delivery task assignment. Two heuristic methods are used in [9] to enable the package routing and transfer using private cars.

Public transport for crowdsourcing delivery is also studied in recent years. A crowdsourced public transportation system is proposed in [12], in which a NP-hard problem is formulated and a heuristic solution is proposed to minimize the time duration for delivering all packages. In [17], a new transit system is designed to carry both passengers and packages, which requires high cost infrastructure to be built.

Existing works on crowdsourcing delivery fails to consider most practical factor in real world (i.e. willingness, package transfers, time constraints, pricing and cost). Besides, the optimization and heuristic methods used works independently at different timestamps and fails to evaluate the possible impact of current decision in the future, which is critical for sequential decision problem, besides the empirical-based methods cannot capture the environmental factors (weather, time). Time constraints also introduce new challenges since courier routing

and dispatching need to be optimized. A rolling horizon framework is proposed in [3] and an adaptive large neighborhood search heuristic is proposed in [29].

6.2 RL-based Route Planning and Order Dispatching

Reinforcement learning is widely applied for sequential decision problems like traffic light control [56], ride-sharing [26], and re-location of bikes and cars [19, 25, 32, 35, 53]. Particularly, RL has been adopted for route planning [14, 16] and order dispatching [25, 33, 34, 51, 60, 66] in recent years.

For route planning, [16] solves the vehicle routing problem (VRP) problem with a joint approach of RL and graph convolutional network (GCN). The same problem is solved in [14] with a combinatorial action space.

For order dispatching, RL is adopted in express package delivery [33, 34] and on-demand ride-hailing [25, 51, 60, 66]. A contextual cooperative RL model is proposed to guide the couriers to meet the delivery requirements while balancing the workload among couriers [33]. A cooperative multi-agent RL model is proposed in [34] to learn the optimal courier dispatching policy to maximize the total number of completed tasks. The driver-order mapping process is formulated and solved as a sequential decision-making problem in [60] to optimize resource utilization and user experience. Ride-hailing is modeled as a parallel ranking problem in [25], where order dispatching and fleet management are jointly considered to optimize the driver income and order response rate based on hierarchical RL. Ride dispatching is modeled as a Semi Markov Decision Process in [51], and transfer learning is used to generalize the model to multiple cities. [66] solves the dispatching problem in a decentralized fashion with multi-agent RL.

Our work differs from the existing works in proposing a profit model as the reward function and designing an ETA module to accelerate the training process and provide a statistical guarantee on the delivery time.

6.3 Human Mobility Applications

Human mobility pattern is widely studied in the community of smart cities and urban computing [20, 61]. Mining public and taxi data leads to many novel applications in recent years [28, 63]. However, existing works on mining transportation data mainly focus sensing [64], traffic control [28], and urban planning [62, 65]. While package delivery study based on mobility pattern is still an emerging field.

7 CONCLUSION

In this paper, we investigate the crowdsourcing delivery problem and propose a hitchhiking delivery system based on public transport. We build a profit model in consideration of multiple practical factors (e.g., multiple hops, time constraints) to support a practical commercial application. We proposed a reinforcement learning algorithm to learn from massive passenger data and package data to make optimal order dispatching decisions. An delivery time estimation module is also designed to filter the actions and accelerate the training process. Evaluation has been conducted on real-world subway data and delivery data, showing a stable profit increase of our system compared with baselines. Specifically, we achieve a 40% increase in profit rates and a 29% increase in delivery rates compared with existing crowdsourcing delivery algorithms. We can improve the profit rate and the delivery rate by 9% and 8% by using time estimation in action filtering, compared with other reinforcement learning algorithms.

ACKNOWLEDGMENTS

The authors would like to thank anonymous reviewers for their valuable comments and suggestions. Prof. Shuai Wang is partially supported by the National Natural Science Foundation of China under Grant No. 61902066, Natural Science Foundation of Jiangsu Province under Grant No. BK20190336, China National Key R&D Program 2018YFB2100302 and Fundamental Research Funds for the Central Universities under Grant No. 2242021R41068.

REFERENCES

[1] Alibaba Group, Local Service BU. 2021. RL-Dispatch Data-set Link. https://tianchi.aliyun.com/dataset/dataDetail?dataId=106807.

[2] Amazon.com. 2017. *U.S. Shipping Rates & Times.* https://www.amazon.com/gp/help/customer/display.html?nodeId=201118670

[3] Alp Arslan, Niels Agatz, Leo G Kroon, and Rob A Zuidwijk. 2016. Crowdsourced Delivery: A Dynamic Pickup and Delivery Problem with Ad-hoc drivers. (2016).

[4] Hive Box. 2019. *Intelligent service solutions by hive-box.* http://www.fcbox.com/en/pc/index.html#/

[5] Chacha Chen, Hua Wei, Nan Xu, Guanjie Zheng, Ming Yang, Yuanhao Xiong, Kai Xu, and Zhenhui Li. 2020. Toward A thousand lights: Decentralized deep reinforcement learning for large-scale traffic signal control. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 34. 3414–3421.

[6] Chao Chen, Sen Yang, Yasha Wang, Bin Guo, and Daqing Zhang. 2020. CrowdExpress: A Probabilistic Framework for On-Time Crowdsourced Package Deliveries. *IEEE Transactions on Big Data* (2020).

[7] Chao Chen, Daqing Zhang, Xiaojuan Ma, Bin Guo, Leye Wang, Yasha Wang, and Edwin Sha. 2016. crowddeliver: Planning City-Wide Package Delivery Paths Leveraging the Crowd of Taxis. *IEEE Transactions on Intelligent Transportation Systems* (2016).

[8] Chao Chen, Daqing Zhang, Leye Wang, Xiaojuan Ma, Xiao Han, and Edwin Sha. 2014. Taxi Exp: A Novel Framework for City-Wide Package Express Shipping via Taxi Crowd Sourcing. In *Ubiquitous Intelligence and Computing, 2014 IEEE 11th Intl Conf on and IEEE 11th Intl Conf on and Autonomic and Trusted Computing, and IEEE 14th Intl Conf on Scalable Computing and Communications and Its Associated Workshops (UTC-ATC-ScalCom)*. IEEE, 244–251.

[9] Wenyi Chen, Martijn Mes, and Marco Schutten. 2018. Multi-hop driver-parcel matching problem with time windows. *Flexible services and manufacturing journal* 30, 3 (2018), 517–553.

[10] Yueyue Chen, Deke Guo, Ming Xu, Guoming Tang, Tongqing Zhou, and Bangbang Ren. 2019. PPtaxi: Non-stop Package Delivery via Multi-hop Ridesharing. *IEEE Transactions on Mobile Computing* (2019).

[11] Geyao Cheng, Deke Guo, Jianmai Shi, and Yudong Qin. 2018. When packages ride a bus: Towards efficient city-wide package distribution. In *2018 IEEE 24th International Conference on Parallel and Distributed Systems (ICPADS)*. IEEE, 259–266.

[12] Geyao Cheng, Deke Guo, Jianmai Shi, and Yudong Qin. 2019. Smart City-Wide Package Distribution Using Crowdsourced Public Transportation Systems. *IEEE Internet of Things Journal* 6, 5 (2019), 7584–7594.

[13] European Commission. 2019. *EU data protection rules.* https://ec.europa.eu/commission/priorities/justice-and-fundamental-rights/data-protection/2018-reform-eu-data-protection-rules_en

[14] Arthur Delarue, Ross Anderson, and Christian Tjandraatmadja. 2020. Reinforcement Learning with Combinatorial Actions: An Application to Vehicle Routing. *Advances in Neural Information Processing Systems* 33 (2020).

[15] Florian Drews and Dennis Luxen. 2013. Multi-hop ride sharing. In *Sixth annual symposium on combinatorial search.*

[16] Lu Duan, Yang Zhan, Haoyuan Hu, Yu Gong, Jiangwen Wei, Xiaodong Zhang, and Yinghui Xu. 2020. Efficiently Solving the Practical Vehicle Routing Problem: A Novel Joint Learning Approach. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 3054–3063.

[17] Ezzeddine Fatnassi, Jouhaina Chaouachi, and Walid Klibi. 2015. Planning and operating a shared goods and passengers on-demand rapid transit system for sustainable city-logistics. *Transportation Research Part B: Methodological* 81 (2015), 440–460.

[18] Michal Feldman, Kevin Lai, Ion Stoica, and John Chuang. 2004. Robust incentive techniques for peer-to-peer networks. In *Proceedings of the 5th ACM conference on Electronic commerce*. ACM, 102–111.

[19] Suining He and Kang G Shin. 2019. Spatio-temporal capsule-based reinforcement learning for mobility-on-demand network coordination. In *The World Wide Web Conference*. 2806–2813.

[20] Andrea Hess, Karin Anna Hummel, Wilfried N Gansterer, and Günter Haring. 2016. Data-driven human mobility modeling: a survey and engineering guidance for mobile networking. *ACM Computing Surveys (CSUR)* 48, 3 (2016), 38.

[21] Thomas Holleczek, Shanyang Yin, Yunye Jin, Spiros Antonatos, Han Leong Goh, Samantha Low, Amy Shi-Nash, et al. 2015. Traffic measurement and route recommendation system for mass rapid transit (mrt). In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 1859–1868.

[22] Postmates Inc. 2019. *Postmates.* https://postmates.com/

[23] Instacart. 2019. *Instacart.* https://www.instacart.com/

[24] Shenggong Ji, Yu Zheng, Zhaoyuan Wang, and Tianrui Li. 2019. Alleviating Users' Pain of Waiting: Effective Task Grouping for Online-to-Offline Food Delivery Services. *the web conference* (2019), 773–783.

[25] Jiarui Jin, Ming Zhou, Weinan Zhang, Minne Li, Zilong Guo, Zhiwei Qin, Yan Jiao, Xiaocheng Tang, Chenxi Wang, Jun Wang, et al. 2019. Coride: joint order dispatching and fleet management for multi-scale ride-hailing platforms. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*. 1983–1992.

[26] Ishan Jindal, Zhiwei Tony Qin, Xuewen Chen, Matthew Nokleby, and Jieping Ye. 2018. Optimizing taxi carpool policies via reinforcement learning and spatio-temporal mining. In *2018 IEEE International Conference on Big Data (Big Data)*. IEEE, 1417–1426.

[27] Ana Justel, Daniel Peña, and Rubén Zamar. 1997. A multivariate Kolmogorov-Smirnov test of goodness of fit. *Statistics & Probability Letters* 35, 3 (1997), 251–259.
[28] Guohao Lan, Weitao Xu, Sara Khalifa, Mahbub Hassan, and Wen Hu. 2016. Transportation mode detection using kinetic energy harvesting wearables. In *Pervasive Computing and Communication Workshops (PerCom Workshops), 2016 IEEE International Conference on*. IEEE, 1–4.
[29] Baoxiang Li, Dmitry Krushinsky, Tom Van Woensel, and Hajo A Reijers. 2016. An adaptive large neighborhood search heuristic for the share-a-ride problem. *Computers & Operations Research* 66 (2016), 170–180.
[30] Siyuan Li, Rui Wang, Minxue Tang, and Chongjie Zhang. 2019. Hierarchical Reinforcement Learning with Advantage-Based Auxiliary Rewards. In *Advances in Neural Information Processing Systems*. 1409–1419.
[31] Yaguang Li, Kun Fu, Zheng Wang, Cyrus Shahabi, Jieping Ye, and Yan Liu. 2018. Multi-task representation learning for travel time estimation. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 1695–1704.
[32] Yexin Li, Yu Zheng, and Qiang Yang. 2018. Dynamic bike reposition: A spatio-temporal reinforcement learning approach. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 1724–1733.
[33] Yexin Li, Yu Zheng, and Qiang Yang. 2019. Efficient and effective express via contextual cooperative reinforcement learning. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 510–519.
[34] Yexin Li, Yu Zheng, and Qiang Yang. 2020. Cooperative Multi-Agent Reinforcement Learning in Express System. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*. 805–814.
[35] Kaixiang Lin, Renyu Zhao, Zhe Xu, and Jiayu Zhou. 2018. Efficient large-scale fleet management via multi-agent deep reinforcement learning. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 1774–1783.
[36] Yan Liu, Bin Guo, Chao Chen, He Du, Zhiwen Yu, Daqing Zhang, and Huadong Ma. 2018. Foodnet: Toward an optimized food delivery network based on spatial crowdsourcing. *IEEE Transactions on Mobile Computing* 18, 6 (2018), 1288–1301.
[37] Renaud Masson, Anna Trentini, Fabien Lehuédé, Nicolas Malhéné, Olivier Péton, and Houda Tlahig. 2017. Optimization of a city logistics transportation system with mixed passengers and goods. *EURO Journal on Transportation and Logistics* 6, 1 (2017), 81–109. https://doi.org/10.1007/s13676-015-0085-5
[38] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. 2015. Human-level control through deep reinforcement learning. *nature* 518, 7540 (2015), 529–533.
[39] Prime Now. 2021. *About Amazon Prime Now*. https://primenow.amazon.com/helpAndAbout
[40] Santiago Paternain, Luiz Chamon, Miguel Calvo-Fullana, and Alejandro Ribeiro. 2019. Constrained reinforcement learning has zero duality gap. In *Advances in Neural Information Processing Systems*. 7555–7565.
[41] Hasan Poonawala, Vinay Kolar, Sebastien Blandin, Laura Wynter, and Sambit Sahu. 2016. Singapore in motion: Insights on public transport service level through farecard and mobile data analytics. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and data mining*. ACM, 589–598.
[42] Rajax. 2018. *Ele.me*. https://www.ele.me/
[43] Sijie Ruan, Zi Xiong, Cheng Long, Yiheng Chen, Jie Bao, Tianfu He, Ruiyuan Li, Shengnan Wu, Zhongyuan Jiang, and Yu Zheng. 2020. Doing in One Go: Delivery Time Inference Based on Couriers' Trajectories. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 2813–2821.
[44] Gireesh Shrimali, Isaac Keslassy, and Nick McKeown. 2004. Designing packet buffers with statistical guarantees. In *Proceedings. 12th Annual IEEE Symposium on High Performance Interconnects*. IEEE, 54–60.
[45] Adish Singla and Andreas Krause. 2013. Truthful incentives in crowdsourcing tasks using regret minimization mechanisms. *the web conference* (2013), 1167–1178.
[46] Statista. 2020. B2C e-commerce sales worldwide from 2012 to 2018 (in billion U.S. dollars). Webpage.
[47] Statista. 2020. Desktop retail e-commerce sales in the United States from 2002 to 2015 (in billion U.S. dollars). Webpage.
[48] Statista. 2020. Retail e-commerce sales worldwide from 2014 to 2023. Webpage.
[49] Lijun Sun and Jian Gang Jin. 2015. Modeling temporal flow assignment in metro networks using smart card data. In *2015 IEEE 18th International Conference on Intelligent Transportation Systems*. IEEE, 836–841.
[50] Richard S Sutton and Andrew G Barto. 2018. *Reinforcement learning: An introduction*. MIT press.
[51] Xiaocheng Tang, Zhiwei Qin, Fan Zhang, Zhaodong Wang, Zhe Xu, Yintai Ma, Hongtu Zhu, and Jieping Ye. 2019. A deep value-network based approach for multi-driver order dispatching. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*. 1780–1790.
[52] Timm Teubner and Christoph M Flath. 2015. The economics of multi-hop ride sharing. *Business & Information Systems Engineering* 57, 5 (2015), 311–324.
[53] Enshu Wang, Rong Ding, Zhaoxing Yang, Haiming Jin, Chenglin Miao, Lu Su, Fan Zhang, Chunming Qiao, and Xinbing Wang. 2020. Joint Charging and Relocation Recommendation for E-Taxi Drivers via Multi-Agent Mean Field Hierarchical Reinforcement Learning. *IEEE Transactions on Mobile Computing* (2020).

[54] Fangxin Wang, Yifei Zhu, Feng Wang, Jiangchuan Liu, Xiaoqiang Ma, and Xiaoyi Fan. 2019. Car4pac: Last mile parcel delivery through intelligent car trip sharing. *IEEE Transactions on Intelligent Transportation Systems* (2019).

[55] Yuan Wang, Dongxiang Zhang, Qing Liu, Fumin Shen, and Loo Hay Lee. 2016. Towards enhancing the last-mile delivery: An effective crowd-tasking model with scalable solutions. *Transportation Research Part E: Logistics and Transportation Review* 93 (2016), 279–293.

[56] Hua Wei, Guanjie Zheng, Huaxiu Yao, and Zhenhui Li. 2018. Intellilight: A reinforcement learning approach for intelligent traffic light control. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 2496–2505.

[57] Wikipedia. 2020. Estimated time of arrival. Webpage.

[58] Wikipedia. 2020. Gig worker. Webpage.

[59] Wikipedia. 2021. Error function. Webpage.

[60] Zhe Xu, Zhixin Li, Qingwen Guan, Dingshui Zhang, Qiang Li, Junxiao Nan, Chunyang Liu, Wei Bian, and Jieping Ye. 2018. Large-scale order dispatch in on-demand ride-hailing platforms: A learning and planning approach. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 905–913.

[61] Dingqi Yang, Bingqing Qu, Jie Yang, and Philippe Cudremauroux. 2019. Revisiting User Mobility and Social Relationships in LBSNs: A Hypergraph Embedding Approach. *the web conference* (2019), 2147–2157.

[62] Chao Zhang, Keyang Zhang, Quan Yuan, Haoruo Peng, Yu Zheng, Timothy Hanratty, Shaowen Wang, and Jiawei Han. 2017. Regions, Periods, Activities: Uncovering Urban Dynamics via Cross-Modal Representation Learning. *the web conference* (2017), 361–370.

[63] Yan Zhang, Yunhuai Liu, Genjian Li, Yi Ding, Ning Chen, Hao Zhang, Tian He, and Desheng Zhang. 2019. Route prediction for instant delivery. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 3, 3 (2019), 1–25.

[64] Yu Zheng, Furui Liu, and Hsun-ping Hsieh. 2013. U-Air: when urban air quality inference meets big data. *ACM SIGKDD* (2013), 1436–1444. https://doi.org/10.1145/2487575.2488188

[65] Yu Zheng, Yanchi Liu, Jing Yuan, and Xing Xie. 2011. Urban computing with taxicabs. In *Proceedings of the 13th international conference on Ubiquitous computing*. ACM, 89–98.

[66] Ming Zhou, Jiarui Jin, Weinan Zhang, Zhiwei Qin, Yan Jiao, Chenxi Wang, Guobin Wu, Yong Yu, and Jieping Ye. 2019. Multi-agent reinforcement learning for order-dispatching via order-vehicle distribution matching. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*. 2645–2653.

[67] Lin Zhu, Wei Yu, Kairong Zhou, Xing Wang, Wenxing Feng, Pengyu Wang, Ning Chen, and Pei Lee. 2020. Order Fulfillment Cycle Time Estimation for On-Demand Food Delivery. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 2571–2580.