Vehicle-oriented ridesharing package delivery in blockchain system

Xuefei Zhang, Junjie Liu, Yijing Li, Qimei Cui, Xiaofeng Tao, Ren Ping Liu, Wenzheng Li

PII: S2352-8648(22)00276-0

DOI: https://doi.org/10.1016/j.dcan.2022.12.008

Reference: DCAN 581

To appear in: *Digital Communications and Networks*

Received Date: 6 March 2021

Revised Date: 8 September 2022

Accepted Date: 9 December 2022

Journal Logo

00 (2022) 1–13

Vehicle-oriented Ridesharing Package Delivery in Blockchain System

Xuefei Zhang*[a], Junjie Liu[a], Yijing Li[a],
Qimei Cui[a], Xiaofeng Tao[a], Ren Ping Liu[b], Wenzheng Li[c]

[a]Department of Information and Communication Engineering, Beijing University of Posts and Telecommunications, Beijing, 100876, China
[b]Department of Electrical and Data Engineering, University of Technology Sydney, Sydney, Australia
[c]Department of Information Technology, Beijing University of Technology, Beijing, 100876, China

Abstract

Package delivery via ridesharing provides appealing benefits of lower delivery cost and efficient vehicle usage. Most existing ridesharing systems operate the matching of ridesharing in a centralized manner, which may result in the single point of failure once the controller breaks down or is under attack. To tackle such problems, our goal in this paper is to develop a blockchain-based package delivery ridesharing system, where decentralization is adopted to remove intermediaries and direct transactions between the providers and the requestors are allowed. To complete the matching process under decentralized structure, an Event-Triggered Distributed Deep Reinforcement Learning (ETDDRL) algorithm is proposed to generate/update the real-time ridesharing orders for the new coming ridesharing requests from a local view. Simulation results reveal the vast potential of the ETDDRL matching algorithm under the blockchain framework for the promotion of the ridesharing profits. Finally, we develop an application for Android-based terminals to verify the ETDDRL matching algorithm.

Keywords:
Blockchain, Dynamic matching, Ridesharing package delivery.

1. Introduction

The success of the ridesharing pattern encourages a wide extension into other fields, e.g., package delivery. The ridesharing package delivery is defined as the package delivery by non-freight truck (i.e., private car, taxi, etc.) to the original destination of the driver or the intermediate destinations on one ride [1]. Compared to the traditional package delivery, the ridesharing delivery can fully utilize the existing private cars or taxis to provide same-day delivery service city-wide [2]. Besides, by allowing the driver and the packages with similar itineraries to share one vehicle, the delivery cost can be reduced significantly [3, 4].

As one of the critical issues in ridesharing to promote the ridesharing efficiency, the vehicle dispatching has been extensively studied [5]. Inherently, the vehicle dispatching in ridesharing, regardless of the properties of the requestor (e.g., package or passenger), can be formulated as a requestor-provider matching problem. The goal of the matching problem operated in a central node (management platform) with a global view over the whole system, is commonly to minimize the cost for requestors or maximize the benefits for providers. Literature [6–9] adopt graph theory or convex optimization to solve the matching problem under the assumption that all requests are known in advance. In addition, some model-free learning algorithm is utilized in a central node to dynamically dispatch vehicles to accept new coming requests [10, 11].

However, nearly all existing ridesharing systems are managed by a third party, which means the customer identities, ridesharing matching, and payment information are recorded or controlled by an intermediary.

*Xuefei Zhang (Corresponding author, zhangxuefei@bupt.edu.cn)
[1]Junjie Liu, Yijing Li, Qimei Cui, and Xiaofeng Tao (e-mail: junjieliu@bupt.edu.cn, liyijing@bupt.edu.cn, cuiqimei@bupt.edu.cn, taoxf@bupt.edu.cn).
[2]Ren Ping Liu (e-mail: renping.liu@uts.edu.au).
[3]Wenzheng Li (e-mail: liwww@bjut.edu.cn).

In this centralized way, data leakage or data tampering might happen once the controller breaks down or is attacked [12], which brings a great threat to the security of customer data. In addition, the system efficiency is another intractable problem in the centralized system because the complexity of global matching scales increases exponentially with the growth of participators (e.g, private cars and packages). Distributed ridesharing system seems to be a solution to the inherent defects of the centralized system. In this way, each vehicle picks the requests individually without coordination with others, and the information is stored in multiple scattered nodes [13].

Currently, some literatures have started the exploration for the decentralized matching [14–16]. For example, a vehicle is free to select the ridesharing packages, but no modifications or additions for the ridesharing order are permitted once the vehicle begins the ridesharing travel [17]. In [14], only requests to similar destinations to the vehicle may be chosen as ridesharing partners. The inefficiency of the strategies in [14] and [16] is evident because they ignore the requests produced during the ridesharing travel and reject the requests for the intermediate places on this ride. In addition, another challenge in the decentralized ridesharing system is to reach a consistent recognition for the vehicle-oriented ridesharing requests between the vehicles that do not trust each other. So, few results can be considered as milestones to the progress of decentralized ridesharing system.

Therefore, in this paper, we fill a gap left by previous work on a decentralized ridesharing system, where vehicles are permitted to add the ridesharing requests during an ongoing ridesharing trip, and the recognition and traceability of the local ridesharing requests are guaranteed as well. More specifically, the main contributions of this paper are:

- An Event-Triggered Distributed Deep Reinforcement Learning (ETDDRL) algorithm that runs in local vehicles is proposed to match the new coming ridesharing requests dynamically during an ongoing ride without coordinating with others. We formulate event-triggered state transitions to avoid the problem of dimensionality in Deep Reinforcement Learning (DRL) and consider the origin and the destination of a new request as a whole to semi-decouple the dynamic request selection and the delivery path planning in order to speed up the dynamic matching.

- We provide a blockchain-based framework for the vehicle-oriented ridesharing delivery system to enable the locally produced matching orders to gain consistent recognition between the vehicles in a tampering proof and transparent way.

- We evaluate the average profit and the average profit per unit distance of the proposed distributed system using a Global Positioning System (GPS) trajectory dataset generated by over 33,000 taxis during a period of 3 months as the provider positions, and simulate the package delivery requests by the passengers' requests from this dataset. The results show that the proposed system improves the average profit of a single trip by 89.17% and the average profit per unit distance by 86.6%, compared to the classic matching strategy.

- We develop an application for Android-based terminals to run the proposed distributed vehicle-oriented ridesharing package delivery system. Except for the real-time location monitoring and auto-payment, this application supports the requestors to submit the delivery request and the providers to dynamically ridesharing matching.

The remainder of this paper is arranged as follows. Section 2 provides the related works about vehicle dispatching. Section 3 describes the system model and the assumptions used throughout the paper. A dynamic matching strategy for vehicle-oriented ridesharing package delivery is proposed and discussed in Section 4. In Section 5, we presents the Blockchain based package delivery ridesharing system. Simulation results are provided to evaluate the performance achieved by our proposed methods in Section 6. Finally, Section 7 draws the conclusion.

2. Related works

Currently, ridesharing has been considered as an icon of green life, providing a convenient and economical means of transportation. Companies or apps providing ridesharing services spring up, like Uber, Didi, Hitch and Roadie, etc. Therein, Hitch and Roadie are two ridesharing package delivery applications. Distinct from the passenger delivery, the requirement for security and delivery time in package delivery is no longer strict, which is an attractive opportunity to permit more ridesharing packages on one ride.

One of the significant issues faced by the ridesharing companies or applications is the vehicle dispatching, which has been extensively studied for ridesharing efficiency promotion. Inherently, the vehicle dispatching in ridesharing can be formulated as a requestor-provider matching problem, which is commonly classified as the static matching and the dynamic matching [17].

The static matching means that the ridesharing route is generated before the departure, and no additional request will be accepted once the trip begins [18]. The preliminary static matching mainly focuses on one-pick ridesharing [19], [20], [21], which means vehicles only choose the requests with a similar origin or destination. For example, Walmart invites its in-store customers to deliver packages to its online customers on their way home [19]. Xu et al. proposed a learning-based path planning method to improve long-term platform efficiency for large scale applications [21]. However, such a one-pick ridesharing system is inefficient in the citywide package delivery since it is hard to pair the exactly similar itineraries from origin to destination, resulting in low vehicle occupancy.

Therefore, multi-pick package delivery has been proposed to improve the ridesharing efficiency. Multi-pick refers to the scenario where a set of geographically dispersed positions (origins or destinations of ridesharing requestors) are visited by the provider [22]. For instance, [23] and [24] adopt a graph-theoretic framework to resent the shareability between various requests and optimize the vehicle dispatching strategy in light of total travel distance. [1] proposed a non-stop package allocation strategy by assigning packages to the vehicle with the same or similar pre-planned route, which can greatly reduce the detour distance and improve the profit of drivers.

As a consequence of neglecting the new coming requests during the ridesharing trip, the ridesharing efficiency of the previously mentioned static matching is low. In order to promote the ridesharing efficiency, the dynamic ridesharing matching is proposed. In this way, the new generated requests during a ridesharing trip may be accepted [25]. For example, [26] introduced a spatio-temporal searching method to find candidate vehicles for every newly generated request, in which the vehicle with the shortest detour is selected as the provider. [27] proposed a kinetic tree algorithm to adjust the routes on-the-fly according to the dynamic requests. Especially, considering the exponential explosion of the searching size for the tree algorithm, it further designs a hotspot-based clustering algorithm to reduce the searching zone. Similarly, a distributed vehicle dispatching strategy is proposed to restrain the search area for each vehicle by filtering out the requests that violate the service quality [28].

In previous literature, both static and dynamic matching were based on a centralized architecture, that is, a central controller is in charge of the vehicle dispatching and the transaction recording. Once the controller breaks down or is attacked, it is likely to happen data leakage or data tampering [29, 30]. In order to tackle this problem, an alternative solution is the decentralized ridesharing system. For example, literature [14] applied distributed Deep Q-Network (DQN) into the vehicle dispatching, in which every vehicle decides the ridesharing partner by itself. But this method relies on the sharing of Q-value to avoid matching conflicts in the distributed scenario, which is unfeasible in practical situations for the privacy concern.

To sum up, most of the previous literature mainly concentrated on the centralized matching algorithm. Besides, most current matching algorithms often assume that the available request has a similar origin or destination to the vehicle. However, the distributed structure makes the centralized algorithm not applicable. Thus, this paper proposed an ETDDRL algorithm that runs in local vehicles. Specifically, to further improve the ridesharing efficiency, this paper semi-decoupled the origin and destination of the request, which enables the vehicle to choose the request without constrain of origin and destination.

Another hard issue in the decentralized framework is how to achieve consistent recognition between the vehicles that do not trust each other without the assistance of a credible third party. Currently, it is widely accepted that blockchain is a technology to enable the distributed nodes to achieve consistent recognition [31, 32]. By this means, kinds of literature have conducted deep research in terms of security [33], performance [34], and application problems [35–37]. For instance, [33] analyzed the forking probability in wireless blockchain networks. [34] compared the performance of Proof of Work (PoW), Proof of Stack (PoS), and Directed Acyclic Graph (DAG) based blockchains. [35] utilized blockchain to conduct the unified access authentication for mobile users. [36] proposed a blockchain-based information sharing system on the Internet of Vehicles (IoV). The authors of [37] designed a blockchain system for the decentralized medical data sharing system.

In our previous work [38], we specified the overall operating process of the blockchian-based package delivery system under the assumption that the vehicle can select their desirable request by themselves. On this basis, we elaborate the distributed dynamic ridesharing matching algorithm to further complete the delivery system in this paper.

3. System model

In this paper, we consider a vehicle-oriented dynamic ridesharing delivery system, as is shown in Fig.1. The city-wide service area is divided into several non-overlapped equal small regions. In this way, the travel path of a vehicle is represented by a sequences of region numbers, and the travel distance is calculated by the Euclidean distances between the centers of the regions that the vehicle passes by. This system involves four types of nodes: requestor (i.e., package), provider (i.e., vehicle), Road Side Unit (RSU) and edge server. We list important acronyms in Table 1.

1) Road side unit and edge server: RSUs are distributed along the road, and each RSU is connected to an edge server via highly reliable cable.The main job of RSUs is to support the information exchange between the associated edge server to other terminals (e.g., providers and requestors). Edge servers maintain a blockchain to record the ridesharing transactions and auto-payment by implementing consensus, smart contract, etc. Specifically, RSUs are controlled by different operators, and they can charge a certain fee from

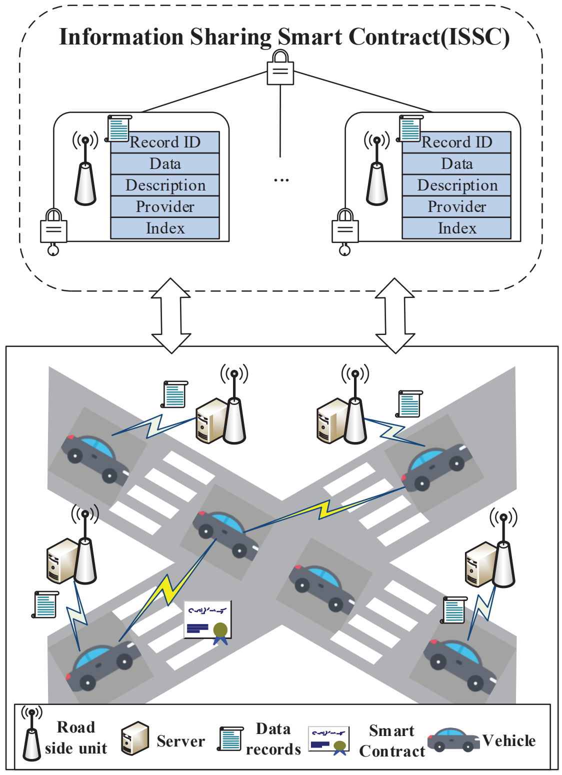


Figure 1: System model of vehicle-oriented dynamic ridesharing delivery system.

the provider for their efforts of helping the global confirmation of the package delivery transaction.

2) Provider: A provider is a vehicle that provides the package delivery ridesharing service. It hears the surrounding package delivery requests and meanwhile conducts the dynamic ridesharing matching in light of the delivery profits, and the route and time restriction. It is worth noting that the matching is locally performed in an individual provider without coordinating with others.

3) Requestor: A requestor is a package that asks for a ridesharing delivery. It sends a logistics request $r_{id}(t, o, d)$ to nearby providers, where *id* represents the identifier of a request, t represents the submission time of the request, o represents the origin of the package, and d represents the destination of the package.

Acronym	Definition
DQN	deep Q-network
DRL	deep reinforcement learning
ETDDRL	event-triggered distributed deep reinforcement learning
EDM	ETDDRL-based dynamic matching
GHGs	green house gases
GPS	global positioning system
IoV	Internet of Vehicles
PSD	pre-scheduling delivery
RSU	road side unit
RSD	random selection delivery
SACA	sequential ant colony algorithm

Tab. 1: Summary of important acronyms.

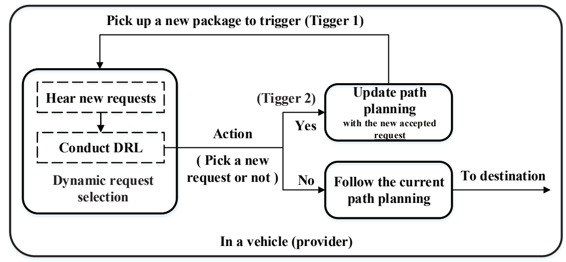


Figure 2: The procedure of ETDDRL-based dynamic matching strategy.

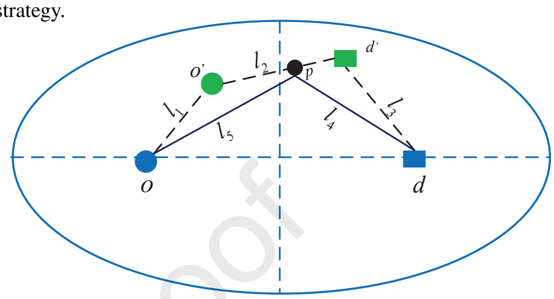


Figure 3: The alternative request searching zone determined by the origin and destination of the vehicle.

4. ETDDRL-based dynamic matching strategy

In this section, we provide an ETDDRL-based dynamic matching strategy. The idea of the proposed strategy is to let providers dynamically select the ridesharing requests during their travel and update the path planning to the vehicle's destination accordingly before completing all the ridesharing package delivery. Considering the nature of decentralization, each vehicle conducts the above issues individually without coordination.

The whole procedure of the strategy is shown in Fig.2, where the provider conducts distributed DRL for the dynamic request selection and on this basis updates the following path planning. Both the two parts are event-triggered. First, picking up a package triggers the vehicle to begin a new round ridesharing matching. Second, the path planning update is triggered by the condition that the vehicle decides to accept a new request.

The contributions of the strategy are three-fold. First, we propose a detour ratio-based searching zone reduction strategy to accelerate the matching process. Second, we formulate the dynamic matching as an event-triggered distributed deep reinforcement learning problem to select a desirable package request. A unique principle of this strategy is the event-triggered state transitions instead of the fixed interval ones, i.e., the vehicle conducts DRL only after the new package pickup, to solve the curse of dimensionality in DRL. Third, we consider the origin and the destination of a new request as a whole to semi-decouple the dynamic request selection and the delivery path planning, which can speed up the package delivery and meanwhile keep the effects of the delivery path on dynamic request selection.

Thus, we propose an ETDDRL from vehicle views to dynamically select the ridesharing requests without coordinating with others, where a detour ratio based searching zone reduction strategy is utilized to accelerate this process. Then, we devise an ant colony algorithm to update the path planning that incorporates the origin and destination of the new request in sequence. The detailed descriptions of the three parts are provided in the following.

4.1. ETDDRL-based dynamic request selection

The formulation of an ETDDRL problem in this paper is present in a tuple $< S, A, P, R >$, where S is the state space, A is the action space, P represents the state transitions, and R is the reward function. Note that our DDRL are given from the event-triggered perspective, that is, the state transitions are triggered by accepting a new request, which is different from the traditional slot-based structure of matching model with fixed equal decision intervals. Following is the description of the four components in the ETDDRL problem.

1) State space: The state space is defined as $S = G \times N \times \Phi \times \Omega$ considering the current position and ridesharing request status of the vehicle, where G is the set of the region index, N is the set of the maximum number of the remaining requests that can be accepted by the vehicle, Φ is the set of the undelivered requests that have been accepted by the vehicle, and Ω is the set of the maximum remaining distance for the vehicle to its destination. The state variable is defined as a four-dimensional vector for $g \in G$, $n \in N$, $r \in \Phi$, and $p \in \Omega$.

2) Action space: The action is to go to the origin of the new selected request denoted by a or reject all new requests given by $a = 0$. Hence, the time-varying action space consists of the region index of the new requests and the region index of the vehicle's destination, denoted by G' for $G' \subseteq G$.

By this means, the provider needs to search the action space G' to match the desirable requests. The recurrent matching attempt will consume a lot of computation resources, which is unbearable for resource-limited vehicles. Therefore, based on [28], we proposed a new action space reduction strategy to decline the attempts in the matching process.

As shown in Fig.3, o and d are the origin and the destination of a provider, o' and d' are the origin and the destination of a package delivery request, the dotted line shows the traveling route if the provider accepts the request, p is a random point on the path, and l_* represents the Euclidean distance of these line segments.

We can easily obtain the relationship between the distances in Fig.3

$$l_4 + l_5 \leq l_1 + l_2 + l_3 \tag{1}$$

To guarantee the QoS of providers, the maximum traveling distance of the vehicle cannot exceed $(1 + \Delta)\, l$, where Δ is the detour ratio and l is the original travel distance (without accepting any ridesharing request) of the vehicle. By this means, we can obtain the constrain that

$$(1 + \Delta)\, l \geq l_1 + l_2 + l_3 \tag{2}$$

Combining with (1) and (2), we can deduce the following equation

$$l_4 + l_5 \leq (1 + \Delta)\, l \tag{3}$$

which is the definition of the elliptical boundary C. This inequality indicates that any point on the path cannot exceed the boundary C determined by o and d.

In this way, the action space in current state is sharply reduced from the whole observation area to the elliptical area $G' \subseteq C$, thus saving the computation consumption in the matching process.

3) Reward function: The reward function represents the profit/penalty to the vehicle if it takes action a under the current state $\mathbf{s}$, which is given by

$$\mathcal{R}(\mathbf{s}, a) = \begin{cases} \kappa D(o', d') & n \geq 1, p > 0, a \neq end \\ 0 & n \geq 0, p > 0, a = end \\ \eta & else \end{cases} \tag{4}$$

where κ is the coefficient to describe the profit, $D(o', d')$ is the Euclid distance of the new request from its origin o' to its destination d', end is the region index of the vehicle's destination, and η is the penalty for exceeding the distance or receiving time restriction.

4) Q deep network: In this section, we utilize the deep Q-networks to update the state and action. This technique is widely used in modern decision-making due to its adaptability to dynamic environments. The optimal action-value function for a vehicle is defined as the maximum expected achievable reward. Thus, for any policy π we have

$$Q^*(s, a) = \max_{\pi} Q^{\pi}(s, a) = \mathbb{E}_{s'}\left[r + \lambda \max_{a'} Q^*(s', a') \,|\, s, a \right] \tag{5}$$

where $0 < \gamma < 1$ is the discount factor for the future.

Here, we consider the origin and the destination of a new request as a whole in ETDDRL to semi-decouple the dynamic request selection and the delivery path planning, which can speed up the package delivery and meanwhile keep the effects of the delivery path on the future dynamic request selection. The whole process of ETDDRL-based dynamic request selection algorithm is shown as Algorithm 1. And in the next subsection, a Sequential Ant Colony Algorithm (SACA)-based path planning is proposed to carry out the dynamic delivery path planning.

4.2. Sequential ant colony algorithm-based path planning

Ant colony algorithm is a promising solution to the NP-hard shortest path problem based on the ant's capability of finding the shortest path from the nest to

Algorithm 1 ETDDRL-based dynamic request selection algorithm

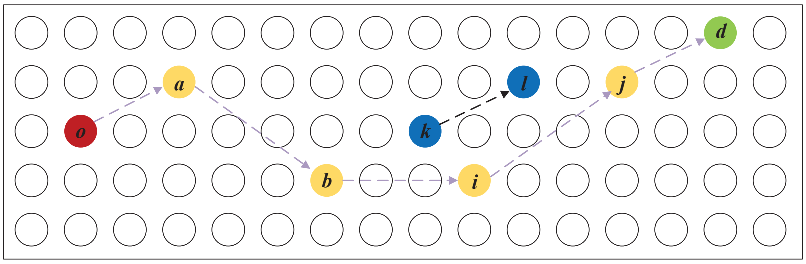
Initialize replay memory $\mathcal{D}$ to capacity N, action-value function Q with random weights θ, current state $s_t = \{G, N, \Phi, \Omega\}$
Initialize the amount of ant M, the origin o and destination d of the vehicle, current region structure $G(V, E)$ and the distance c_{ij} from region i to region j
for episode = 1,T **do**
  Construct the action set A according to the local data set
  Select a random action a_t from a with probability ε
  otherwise select $a_t = \arg\max_a Q(s_t, a; \theta)$
  Execute action a_t and obtain s_{t+1} with the help of $SACA\left(G(V,E), s_t, d, c_{ij}, M, o', d'\right)$(seen in Algorithm 2)
  Store transition (s_t, a_t, r_t, s_{t+1}) in D
  Sample random minibatch of transitions (s_t, a_t, r_t, s_{t+1}) from D
  Set $y_i = \begin{cases} r_j & \text{if } s_{t+1} = d \\ r_j + \gamma \max\limits_{a'} \hat{Q}(s_{t+1}, a'; \theta) & \text{otherwise} \end{cases}$
  Perform a gradient descent step on $\left(y_i - Q\left(s_j, a_j; \theta\right)\right)^2$ with respect to the network parameters θ
end for

a food source. But it cannot satisfy the sequential path planning under the dynamic topologies. A unique challenge in our path planning is dynamically incorporating two nodes (i.e., the origin and the destination of the new accepted request) in order. To tackle this problem, we focus on some important positions in a path, i.e., the starting point, the end point, the necessary points and the insertion points. The definitions of these points in Fig.4 are listed in the following:

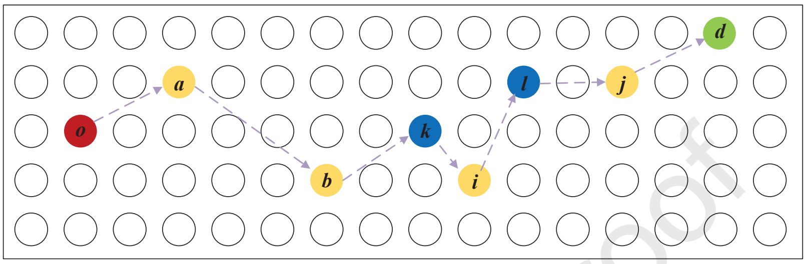
- *Starting point* is the current position of a vehicle(provider), as shown in the red node in Fig.4;
- *End point* is the destination of the vehicle, as shown in the green node in Fig.4;
- *Necessary points* are the origin or destination of the requests that the vehicle has accepted but has not yet delivered, as shown in the yellow nodes in Fig.4;
- *Insertion points* are the origin and the destination of the new accepted request, as shown in the blue nodes in Fig.4(a) and Fig.4(b). Particularly, the vehicle must pass the two nodes in order (given by the black dotted arrow in Fig.4(a)) because the prerequisite to the destination of a new request is that the vehicle has picked up the package. Once the vehicle picks up a new package, the origin

Algorithm 2 The SACA-based path planning algorithm

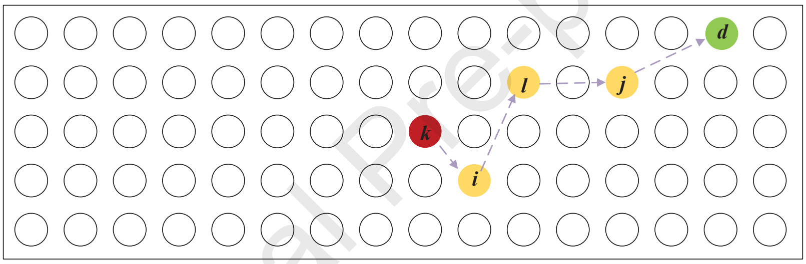
Input: $G(V, E)$, s_t, d, c_{ij}, M, o', d';
/* M is the amount of ant and c_{ij} is the distance from i to j */
Output: *best*: the length of the shortest path; *path*: the shortest path;
procedure SET_INIT_INFORMATION
  for $\forall k \in M$ **do**
    read the current region number v_s and the necessary points set $Point_k$ from s_t;
    let v_s be the starting region for ant k;
    $r_k \leftarrow v_s$;
    /*r_k is the region where ant k located*/
    add o' to the necessary points set $Point_k$;
  end for
end procedure
procedure CONSTRUCT_ROUTES
  for $i \leftarrow 1$ to $|Point_k|$ **do**
    for $\forall k \in M$ **do**
      choose the next region s_k from $Point_k$;
      if $s_k = o'$ **then**
        add d' to the necessary points set $Point_k$;
      end if
      add $edge(r_k, s_k)$ to $Tour_k$;
      $r_k \leftarrow s_k$;
      remove s_k from $Point_k$;
    end for
  end for
  for $\forall k \in M$ **do**
    add $edge(s_k, d)$ to $Tour_k$;
  end for
end procedure
procedure UPADTE_PHEROMONES
  compute $L_k, \forall k \in M$;
  /* L_k is the tour length of ant k */
  update $\tau_{r,s}$ according to the rule;
end procedure
procedure MAIN
  for $\forall edge(r, s) \in E$ **do**
    $\tau_{r,s} \leftarrow \tau_0$;
    /*$\tau_{r,s}$ is the pheromone concentration from r to s*/
    $\eta_{r,s} \leftarrow 1/c_{r,s}$;
    /*$\eta_{r,s}$ is the heuristic function */
  end for
  while Not End_Condition **do**
    Set_int_information;
    Construct_routes;
    Update_pheromones;
  end while
end procedure



(a) The current path planning is $o \to a \to b \to i \to j \to d$, where the vehicle is in the starting point o. After the ETDDRL, the vehicle decides to deliver a new package from k to l based on ETDDRL



(b) After accepting a new ridesharing request, the vehicle updates the path planning $o \to a \to b \to k \to i \to l \to j \to d$ by the sequential SACA



(c) After the vehicle arrives in point k to pick up the new package, it will hear the new requests and adopt ETDDRL to decide whether accept a new ridesharing request. If not, it will follow the current path planning $k \to i \to l \to j \to d$

Figure 4: An example of SACA-based path planning under different conditions.

of the new accepte Now, we evaluate the average profit of a single ride in terms of the detour ratio. Fig.10 shows that the average profit of a single trip of the proposed EDM strategy grows in steps with the ascending detour ratio. In contrast, the impact of the detour ratio on the average profit of a single trip under RSD and PSD is weak. The reason for the growth is the permission to accepting new requests during the ridesharing traveling under the EDM strategy. In addition, an interesting phenomenon is the stepwise growth under EDM strategy, i.e., some intermittent low-growth periods (such as 1.2 ~1.4, 1.6 ~1.8, etc.) exist due to the possibility that the incremental distance brought by the increased detour ratio is not enough to support a new ridesharing delivery request. During this period, the improvement of the profit is slight. A new significant promotion for profit appears once the increased detour ratid request changes into the starting point and the destination of the new accepted request changes into a necessary point.

Algorithm 2 shows the detailed steps of the SACA-based path planning method, and Figs.4 (a) to (c) depict an example of this. In Fig.4 (a), the vehicle in the starting point o with the current path planning $o \to a \to b \to i \to j \to d$, decides to provide ridesharing from k to l for a new request. On this basis, the delivery path is updated to $o \to a \to b \to k \to i \to l \to j \to d$, as is shown in Fig.4 (b). Once the vehicle arrives at point k to pick up the new package, the starting point is updated as point k. Then, the vehicle hears new requests and adopts ETDDRL to decide whether accept a new ridesharing request. If not, it will follow the current path planning $k \to i \to l \to j \to d$, as is shown in Fig.4 (c); otherwise, it will update the path planning.

5. Blockchain-based package delivery ridesharing system

Except for the dynamic matching, reaching a consistent recognition for the package delivery transaction information (e.g., origin, destination, receiver, price and auto-payment) between the participators is also a

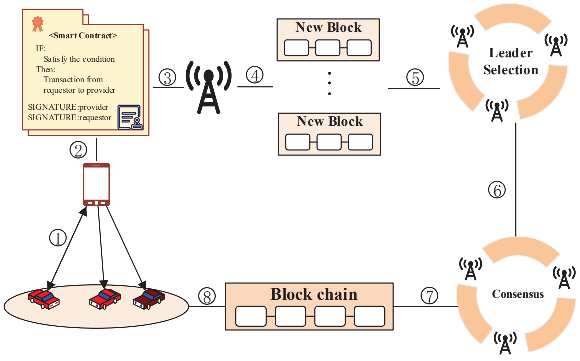


Figure 5: Overall flow of matching results confirmation and transaction record over the ridesharing package delivery blockchain system [38].

vital and intractable issue in a decentralized system, in which the key issue is to find a way to achieve the trusts between the untrusted participators without a privileged third-party. In this section, we utilize blockchain to achieve consistent recognition in an untrusted decentralized system, where the visibility of on-chain information provides public monitoring for authorized users, and Hash pointers realize tamper-proof information on the chain.

In our previous work [38], we developed a blockchain-based ridesharing package delivery system as shown in Fig.5. On this basis, the overall flow of matching results confirmation and transaction record over the blockchain system is summarized as follows:

1) Request broadcast: After generating the delivery request, the requestor broadcasts it to nearby providers within a certain range.

2) Request selecting: The provider performs dynamic ridesharing matching scheme to select a delivery request in terms of maximizing their profit. After that, the provider signs a smart contract with the parameter (e.g., origin, destination, receiver and condition for the payment) generated by the desirable request, and sends the contract to the corresponding requestor.

3) Contract broadcast: The requestor signs a smart contract according to their own criteria and broadcasts it to all edge servers in the system. Specifically, the smart contract not only contains the protocol negotiated by the requestor and provider but also has the service information, e.g, origin, destination, and payment.

4) New block generation: The edge server verifies the signature authenticity of both parties in the contract and stores it in the information pool waiting to be added to the blockchain system.

5) Hash-oriented leader selection: The edge server packs the broadcast information received in the previous period into a new block, namely *newBlock*. Next, each edge server starts the hash calculation based on the transaction data of *newBlock* and the identifier of the edge server. Subsequently, the edge server with the lowest hash value is selected as the leader.

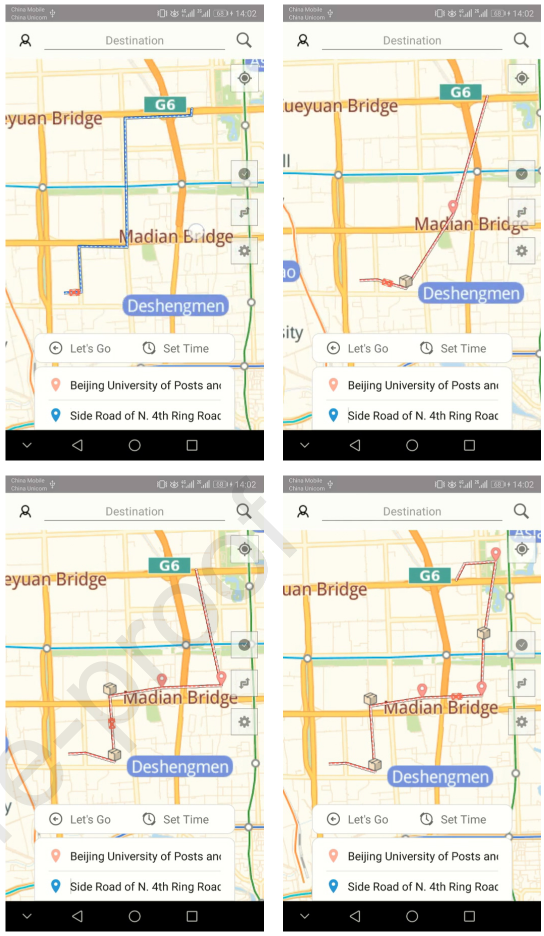


Figure 6: The dynamic matching process in a single trip, and the origin and destination of the package is represented by the box and anchor, respectively. (a) the original path of the vehicle; (b) the renewed path after picking up the first package (shown as the box on the map); (c) the renewed path after picking up the second package; (d) the renewed path after picking up the third package.

6) Information consensus over edge servers: Once the leader has been determined, each edge server verifies the transaction information in the *newBlock* and attaches its digital signature after the verification. Then, the verified replicas are sent to other edge servers. Other edge servers receive the replicas and verify the signature. And when the number of verified replicas exceeds 2/3 of the total number of edge servers, the server sends a confirmation message to the leader.

7) Result information: When the confirmation message received by the leader exceeds 2/3 of the total number of nodes, it indicates that the *newBlock* has been successfully added to the blockchain. The leader returns an acknowledgement to both the requestor and the provider, after which the provider starts the delivering.

8) Payment: When the vehicle arrives at the destination of the request, the payment can be triggered with the real-time coordination of the provider and the digital signature of the receiver.

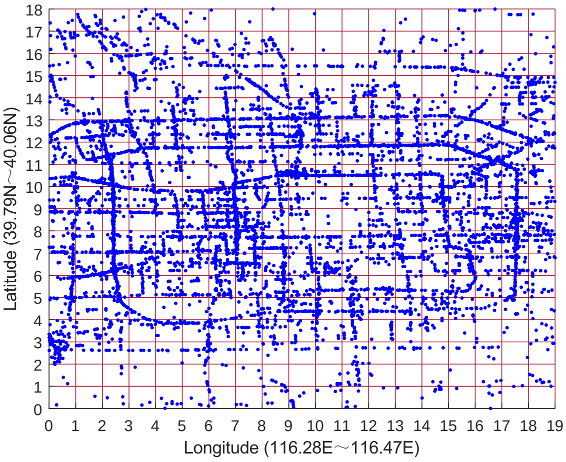


Figure 7: The taxis distribution from 9:00 to 10:00 at 2012-11-27 in the observation area.

6. Simulation result and discussion

In this section, we evaluate the performance of the proposed vehicle-oriented dynamic ridesharing system and develop an android-based application for the requestor (package owner) and the provider (vehicle driver).

6.1. Android-based application for requestor and provider

Fig.6 shows the dynamic matching process of the android-based application. The input box at the bottom of the screen is the origin and destination of the vehicle, and the blue dot-line is the corresponding travel route. The vehicle can hear the surrounding requests and select a desirable one for delivery (the box and anchor represent the origin and destination of the package). After selection, the application renews the red dot-line route with the restriction of request sequence. This matching process will repeat until the vehicle arrives at the destination.

6.2. The real data of the providers' trajectories

The vehicle movement is collected from the real-world datasets of taxis in Beijing (with the longitude from 116.28E to 116.47E and the latitude from 39.79N to 40.06N). The observation area is divided into 342 non-overlapped regions where each region is a rectangle area (length: 0.01E, width: 0.015N), as is shown in Fig.7.

The dataset includes the information on the trajectories and the vacant or loaded status of 12509 taxis for a month (from 2012/11/01 to 2012/11/27). The data format can be shown in Table 2. The sampling frequency is from 1/14Hz to 1Hz.

The explanation of the data in Table 2 is given by

- ID: taxi identifier;
- Timestamp: the upload time of the record, of which format is yyyy-mm-dd-hh-mm-ss;

ID	Timestamp	Longitude	Latitude	Status
164798	20121102001234	116.3108139	40.0201569	2
488249	20121102001511	116.4125519	39.8944168	1
...	...	...	...	...
486276	20121102081332	116.4171829	39.8701134	0

Tab. 2: The data format.

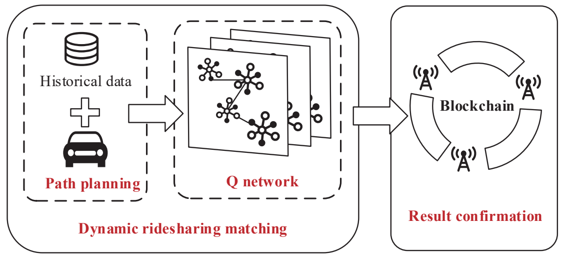


Figure 8: The construction of the vehicle-oriented ridesharing delivery in the blockchain system.

- Location: the longitude and latitude of a taxi;
- Status: 0 for vacant, 1 for loaded, 2 for parking, 3 for offline, and 4 for all other states.

6.3. The generation of the package delivery request

The real data of passenger delivery requests is used to imitate package delivery requests since the two kinds of requests are highly related to the distribution of residence and workplace. Specifically, we deduce the status of providers according to the state changing from vacant (or loaded) to loaded (or vacant), of which positions are considered as the origin and the destination of the request, respectively. In addition, the deadline of the delivery to the destination is written in the request. Thus, the format of the delivery request is given by $\langle RequestID, origin, destinationanddeadline\rangle$.

6.4. The efficiency of the dynamic ridesharing delivery system

In this subsection, we evaluate the efficiency of the proposed ridesharing system. As is shown in Fig.8, the ridesharing delivery system is divided into two stages, in which the Q network for ridesharing matching is trained on the historical data as well as SACA-based path planning along with the new coming request, and the consensus is conducted for the matching results confirmation over the blockchain-based system.

The time on the dynamic ridesharing matching and result confirmation are shown in Table 3. The ridesharing matching process involving the path planning can be complete within 100 ms, and then the matching result is confirmed by blockchain within 21s. Specifically, the concrete process of the confirmation time has been clarified in our previous work [38]. Although the time consumed on the consensus of blockchain overwhelms that of the dynamic ridesahring matching, it is acceptable to the requestors and providers in the realistic condition and is negligible to the whole ridesharing delivery trip.

Dynamic ridesharing matching (Q network update and path planning)	68ms	73ms	79ms	85ms
Result confirmation (blockchain)	11.5s	15s	17.2s	20.5s

Tab. 3: The time on the ridesharing matching and confirmation.

Parameters	Setting Value
Learning rate	0.001
Memory size	100
Reward decay	0.9
e-greedy	0.3
Observation step	100
Batch size	36
Price per units	2
Negative reward	-5
Epochs	7000

Tab. 4: The value of the system parameters.

6.5. *ETDDRL-based dynamic matching strategy*

In this subsection, we evaluate the performance of the proposed ETDDRL-based Dynamic Matching (EDM) strategy on the average profit of a single trip and the average profit per unit distance, compared to the Random Selection Delivery (RSD) and Pre-Scheduling Delivery (PSD). Both of the two compared strategies complete the ridesharing matching and path planning before the departure, i.e., in a centralized matching way, where RSD selects the ridesharing requests that satisfy the limits of the detour and the time in a random way and PSD utilizes ant colony algorithm to maximize the provider's profit. The parameter setting of the ETDDRL-based dyanamic ridesharing matching is shown in Table 4:

Fig.9 shows the loss and reward in each training epoch, where the loss indicates the convergence trending of the Q-value based neural network and the reward reflects the long-term average gains under different states. The loss and the reward of DQN decrease and increase to a stable value along with the training, which indicates that the Q-value-based neural network converges to a global optimum. o enables the distance to support an additional request.

Except for the average profit of a single trip, the av-

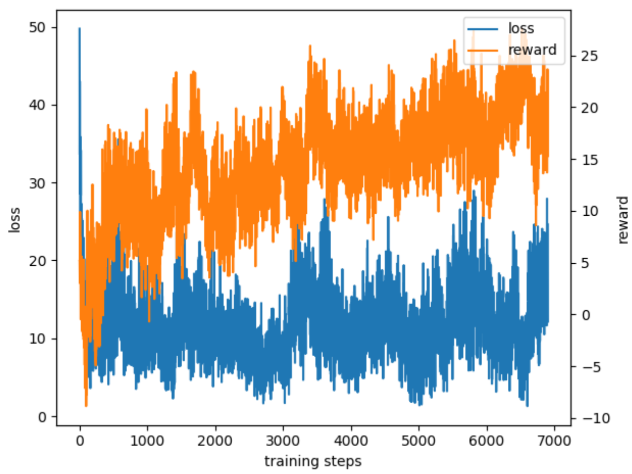


Figure 9: The training result of Q-value based neural network.

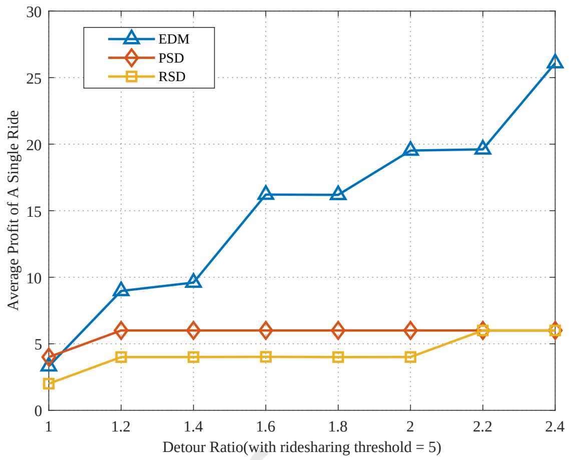


Figure 10: Average profit of a single ride under different detour ratio restriction.

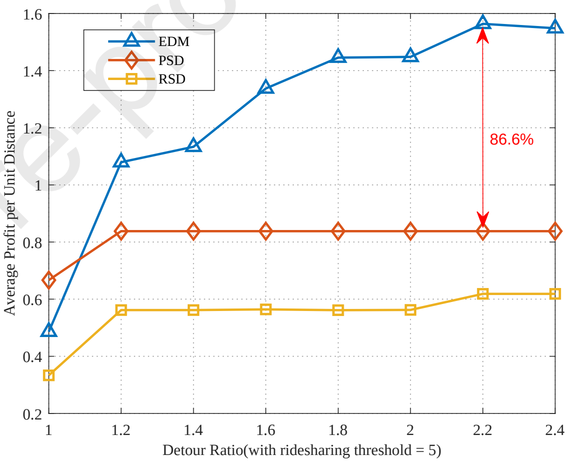


Figure 11: Average profit per unit distance under different detour ratio restriction.

erage profit per unit distance is an essential metric to evaluate the efficiency of the ridesharing delivery strategy. In Fig.11, we find that it has a similar trend to the average profit of a single trip with the ascending detour ratio. The only difference is that the performance of the EDM strategy tends to be stable, instead of a continually growth in Fig.10. The reason is that the gain from the requests selection appears only when the number of ridesharing deliveries has reached the ridesharing threshold under the high detour ratio. The ridesharing threshold refers to the maximum number of ridesharing deliveries for a provider during a single ride.

Fig.12 provides the average profit of a single trip in terms of the different ridesharing thresholds, where ridesharing thresholds refer to the maximum packages that the vehicle can pick up. The results agree with our intuitions that more ridesharing brings more average profits under EDM strategy. However, the gains are greatly restricted under the PSD and NSD strategy due to the fact that they are only permitted to choose the

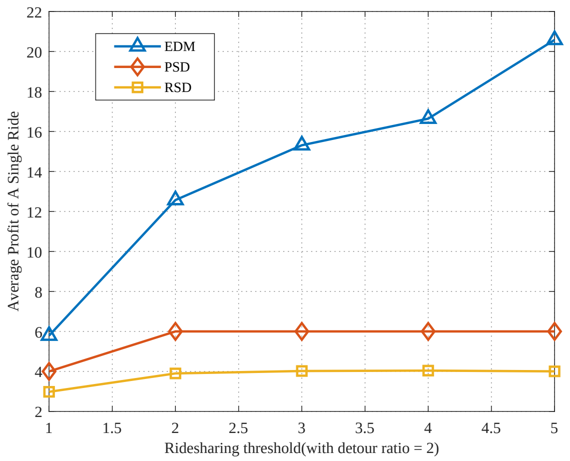


Figure 12: Average profit of a single ride under different ridesharing threshold.

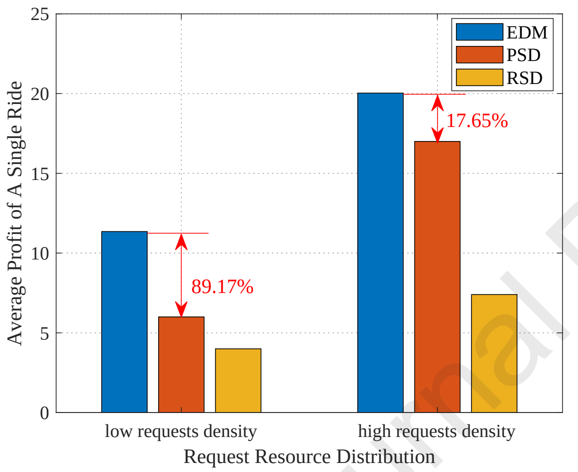


Figure 13: Average profit ratio of a single ride under different requests density(receiving time is set to be 3 and the detour ratio is set to be 1.5).

requests before departure.

Finally, we discuss the impact of the request density, e.g., the high requests density and the low requests density, on the average profit of a single trip. Fig.13 tells us that the profit of EDM strategy is always higher than that of two other strategies no matter for high or low request density. The explanation is that EDM is permitted to accept new requests for the ridesharing travel. Besides, it is easy to understand that the profit of all three strategies are higher under the high request density than under the low request density. A noticeable phenomenon is that the gain (89.17%) of EDM strategy to PSD strategy in the low request density is far larger than that (17.63%) in high request density. The reason is that the gain results from the requests selection are only under the high request density,(in this condition, the number of ridesharing deliveries is always up to the upper threshold) while the gain incurred by both the more ridesharing deliveries and request selection is under the low request density.

7. Conclusion

This paper proposes a vehicle-oriented dynamic ridesharing delivery system, where blockchain is utilized to construct the decentralized structure, and an EDM strategy is proposed to dynamically match the new coming ridesharing requests during an ongoing ride. The simulation results demonstrate the vast potential of the vehicle-oriented ridesharing system under the blockchain framework for the promotion of the ridesharing profits and transaction security. Finally, we develop an application for android-based terminals to run the proposed blockchain-based ridesharing package delivery system.

8. Acknowledgements

This work is supported by National Natural Science Foundation of China (Grant No. 62271073 and 61971066), Beijing Natural Science Foundation (L212003), and the National Youth Top-notch Talent Support Program.

References

[1] Y. Chen, D. Guo, M. Xu, G. Tang, T. Zhou, B. Ren, Pptaxi: Non-stop package delivery via multi-hop ridesharing, IEEE Transactions on Mobile Computing.

[2] F. Wang, Y. Zhu, F. Wang, J. Liu, Ridesharing as a service: Exploring crowdsourced connected vehicle information for intelligent package delivery, in: 2018 IEEE/ACM 26th International Symposium on Quality of Service (IWQoS), 2018, pp. 1–10.

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

[4] J.-F. Rougès, B. Montreuil, Crowdsourcing delivery: New interconnected business models to reinvent delivery, in: 1st international physical internet conference, Vol. 1, 2014, pp. 1–19.

[5] D. J. Fagnant, K. M. Kockelman, Dynamic ride-sharing and fleet sizing for a system of shared autonomous vehicles in austin, texas, Transportation 45 (1) (2018) 143–158.

[6] I. Dumitrescu, S. Ropke, J.-F. Cordeau, G. Laporte, The traveling salesman problem with pickup and delivery: polyhedral results and a branch-and-cut algorithm, Mathematical Programming 121 (2) (2010) 269.

[7] W. Lu, Optimization and mechanism design for ridesharing services, Ph.D. thesis, Doctoral dissertation, Texas A and M University, `http://hdl.handle.net/1969.1/156279.` Accessed March 1, 2021 (2015).

[8] A. Jauhri, B. Foo, J. Berclaz, C. C. Hu, R. Grzeszczuk, V. Parameswaran, J. P. Shen, Space-time graph modeling of ride requests based on real-world data, arXiv preprint arXiv:1701.06635.

[9] V. M. de Lira, V. C. Times, C. Renso, S. Rinzivillo, Comewithme: An activity-oriented carpooling approach, in: 2015 IEEE 18th International Conference on Intelligent Transportation Systems, 2015, pp. 2574–2579.

[10] T. Oda, C. Joe-Wong, Movi: A model-free approach to dynamic fleet management, in: IEEE INFOCOM 2018-IEEE Conference on Computer Communications, 2018, pp. 2708–2716.

[11] C. Mao, Y. Liu, Z.-J. M. Shen, Dispatch of autonomous vehicles for taxi services: A deep reinforcement learning approach, Transportation Research Part C: Emerging Technologies 115 (2020) 102626.

[12] J. Kang, R. Yu, X. Huang, M. Wu, S. Maharjan, S. Xie, Y. Zhang, Blockchain for secure and efficient data sharing in vehicular edge computing and networks, IEEE Internet of Things Journal 6 (3) (2018) 4660–4670.
[13] K. Bathla, V. Raychoudhury, D. Saxena, A. D. Kshemkalyani, Real-time distributed taxi ride sharing, in: 2018 21st International Conference on Intelligent Transportation Systems (ITSC), 2018, pp. 2044–2051.
[14] A. O. Al-Abbasi, A. Ghosh, V. Aggarwal, Deeppool: Distributed model-free algorithm for ride-sharing using deep reinforcement learning, IEEE Transactions on Intelligent Transportation Systems 20 (12) (2019) 4714–4727.
[15] Y. Semenko, D. Saucez, Distributed privacy preserving platform for ridesharing services, in: International Conference on Security, Privacy and Anonymity in Computation, Communication and Storage, 2019, pp. 1–14.
[16] W. Zhao, Y. Qin, D. Yang, L. Zhang, W. Zhu, Social group architecture based distributed ride-sharing service in vanet, International Journal of Distributed Sensor Networks 10 (3) (2014) 650923.
[17] S. Silwal, M. O. Gani, V. Raychoudhury, A survey of taxi ride sharing system architectures, in: 2019 IEEE International Conference on Smart Computing (SMARTCOMP), 2019, pp. 144–149.
[18] Y. Wang, X. Jiang, L. H. Lee, E. P. Chew, K. C. Tan, Tree based searching approaches for integrated vehicle dispatching and container allocation in a transshipment hub, Expert Systems with Applications 74 (2017) 139–150.
[19] J. W. Alistair Barr, Exclusive: Walmart may get customers to deliver packages to online buyers, forbes, [EB/OL], `https://tinyurl.com/yccje985` Accessed September 7, 2020.
[20] X. Tang, Z. Qin, F. Zhang, Z. Wang, Z. Xu, Y. Ma, H. Zhu, J. Ye, 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, 2019, pp. 1780–1790.
[21] Z. Xu, Z. Li, Q. Guan, D. Zhang, Q. Li, J. Nan, C. Liu, W. Bian, J. Ye, 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, 2018, pp. 905–913.
[22] M. W. Ulmer, J. C. Goodson, D. C. Mattfeld, B. W. Thomas, Modeling dynamic vehicle routing problems: A literature review and framework, in: Working Paper, 2019.
[23] J. Alonso-Mora, S. Samaranayake, A. Wallar, E. Frazzoli, D. Rus, On-demand high-capacity ride-sharing via dynamic trip-vehicle assignment, Proceedings of the National Academy of Sciences 114 (3) (2017) 462–467.
[24] P. Santi, G. Resta, M. Szell, S. Sobolevsky, S. H. Strogatz, C. Ratti, Quantifying the benefits of vehicle pooling with shareability networks, Proceedings of the National Academy of Sciences 111 (37) (2014) 13290–13294.
[25] U. Ritzinger, J. Puchinger, R. F. Hartl, A survey on dynamic and stochastic vehicle routing problems, International Journal of Production Research 54 (1) (2016) 215–231.
[26] S. Ma, Y. Zheng, O. Wolfson, Real-time city-scale taxi ridesharing, IEEE Transactions on Knowledge and Data Engineering 27 (7) (2014) 1782–1795.
[27] Y. Huang, R. Jin, F. Bastani, X. S. Wang, Large scale real-time ridesharing with service guarantee on road networks, arXiv preprint arXiv:1302.6666.
[28] M. Zhu, X.-Y. Liu, X. Wang, An online ride-sharing path-planning strategy for public vehicle systems, IEEE Transactions on Intelligent Transportation Systems 20 (2) (2018) 616–627.
[29] D. Huang, S. Misra, M. Verma, G. Xue, Pacp: An efficient pseudonymous authentication-based conditional privacy protocol for vanets, IEEE Transactions on Intelligent Transportation Systems 12 (3) (2011) 736–746.
[30] Z. Yang, K. Yang, L. Lei, K. Zheng, V. C. Leung, Blockchain-based decentralized trust management in vehicular networks, IEEE Internet of Things Journal 6 (2) (2018) 1495–1505.
[31] X. Fu, H. Wang, P. Shi, A survey of blockchain consensus algorithms: Mechanism, design and applications, Science China Information Sciences 64 (2) (2021) 1–15.
[32] X. Zhang, W. Xia, X. Wang, J. Liu, Q. Cui, X. Tao, R. P. Liu, The block propagation in blockchain-based vehicular networks, IEEE Internet of Things Journal.
[33] Q. Liu, Y. Xu, B. Cao, L. Zhang, M. Peng, Unintentional forking analysis in wireless blockchain networks, Digital Communications and Networks.
[34] B. Cao, Z. Zhang, D. Feng, S. Zhang, L. Zhang, M. Peng, Y. Li, Performance analysis and comparison of pow, pos and dag based blockchains, Digital Communications and Networks 6 (4) (2020) 480–485.
[35] X. Ling, J. Wang, T. Bouchoucha, B. C. Levy, Z. Ding, Blockchain radio access network (b-ran): Towards decentralized secure radio access paradigm, IEEE Access 7 (2019) 9714–9723.
[36] J. Kang, R. Yu, X. Huang, M. Wu, S. Maharjan, S. Xie, Y. Zhang, Blockchain for secure and efficient data sharing in vehicular edge computing and networks, IEEE Internet of Things Journal 6 (3) (2018) 4660–4670.
[37] S. Wang, J. Wang, X. Wang, T. Qiu, Y. Yuan, L. Ouyang, Y. Guo, F.-Y. Wang, Blockchain-powered parallel healthcare systems based on the acp approach, IEEE Transactions on Computational Social Systems 5 (4) (2018) 942–950.
[38] X. Zhang, J. Liu, Y. Li, Q. Cui, X. Tao, R. P. Liu, Blockchain based secure package delivery via ridesharing, in: 2019 11th International Conference on Wireless Communications and Signal Processing (WCSP), 2019, pp. 1–6.

Conflict of interest statement

We declare that we have no financial and personal relationships with other people or organizations that can inappropriately influence our work, there is no professional or other personal interest of any nature or kind in any product, service and/or company that could be construed as influencing the position presented in, or the review of, the manuscript entitled, “Vehicle-oriented Ridesharing Package Delivery in Blockchain System”